Inferencia Estadística

Solución: EDA2

DSLab

noviembre, 2024

```r
# Ejercicio 1 (tidyr y dplyr)
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
# A partir del siguiente dataframe realizar las siguientes operaciones de limpieza de datos:
set.seed(1)
stocks <- data.frame(
  time = as.Date('2009-01-01') + 0:9,
  Walmart = rnorm(10, 20, 1),
  Target = rnorm(10, 20, 2),
  Walgreens = rnorm(10, 20, 4)
)
#        time  Walmart   Target Walgreens
# 1  2009-01-01 19.37355 23.02356  23.67591
# 2  2009-01-02 20.18364 20.77969  23.12855
# 3  2009-01-03 19.16437 18.75752  20.29826
# 4  2009-01-04 21.59528 15.57060  12.04259
# 5  2009-01-05 20.32951 22.24986  22.47930
# 6  2009-01-06 19.17953 19.91013  19.77549
# 7  2009-01-07 20.48743 19.96762  19.37682
# 8  2009-01-08 20.73832 21.88767  14.11699
# 9  2009-01-09 20.57578 21.64244  18.08740
# 10 2009-01-10 19.69461 21.18780  21.67177
```

```
# Como se puede observar hay un problema de clave-valor en las compañias con sus observaciones.
# Transformar los datos para que tengan una clave stock y el valor sea el precio.
# Por lo tanto se requiere la funcion "gather".

# Opcion 1:
new_stocks <- gather(data = stocks, key = stock, value = price, Walmart, Target, Walgreens)

# Opcion 2:
new_stocks <- gather(data = stocks, key = stock, value = price, Walmart:Walgreens)

# Opcion 3:
new_stocks <- gather(data = stocks, key = stock, value = price, -time)
# El último argumento, -time, significa que todas las columnas excepto el tiempo
# contienen los pares clave-valor.


# Devolver el dataframe al estado original utilizando la funcion "spread".
original_stocks <- spread(data = new_stocks, key = stock, value = price)

# Utilizando el operador tuberia %>% se desea realizar las siguientes operaciones anidadas.
# 1) Transformar los datos para que tengan una clave stock y el valor sea
# el precio mediante la funcion "gather".
# 2) Agrupar los datos por la clave stock mediante la funcion "group_by".
# 3) Obtener el precio minimo y maximo utilizando la funcion "summarise".

stocks %>%
  gather(key = stock, value = price,Walmart:Walgreens)%>%
  group_by(stock) %>%
  summarise(min = min(price), max = max(price))
```

```
## # A tibble: 3 x 3
##   stock       min   max
##   <chr>     <dbl> <dbl>
## 1 Target     15.6  23.0
## 2 Walgreens  12.0  23.7
## 3 Walmart    19.2  21.6
```

```
######################################################################
# Ejercicio 2 (dplyr)

library(dplyr)
library(nycflights13)

# COMPROBACION.
# Observamos los distintos dataframes que nos proporcionan.
# Utilizamos el nombre del paquete y doblemente dos puntos (::) para comprobarlo.
# Tambien se puede utilizar el nombre del dataframe si previamente estamos familiarizados.
```

```
# PRIMERA OBSERVACION.
# Comprobamos las variables de cada uno de los datasets que nos proporcionan
# mediante la instrucción "head".
print(head(flights))
```

```
## # A tibble: 6 x 19
##    year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##   <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
## 1  2013     1     1      517            515         2      830            819
## 2  2013     1     1      533            529         4      850            830
## 3  2013     1     1      542            540         2      923            850
## 4  2013     1     1      544            545        -1     1004           1022
## 5  2013     1     1      554            600        -6      812            837
## 6  2013     1     1      554            558        -4      740            728
## # i 11 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>
```

```
print(head(airports))
```

```
## # A tibble: 6 x 8
##   faa   name                            lat   lon   alt    tz dst   tzone
##   <chr> <chr>                         <dbl> <dbl> <dbl> <dbl> <chr> <chr>
## 1 04G   Lansdowne Airport              41.1 -80.6  1044    -5 A     America/Ne~
## 2 06A   Moton Field Municipal Airport  32.5 -85.7   264    -6 A     America/Ch~
## 3 06C   Schaumburg Regional            42.0 -88.1   801    -6 A     America/Ch~
## 4 06N   Randall Airport                41.4 -74.4   523    -5 A     America/Ne~
## 5 09J   Jekyll Island Airport          31.1 -81.4    11    -5 A     America/Ne~
## 6 0A9   Elizabethton Municipal Airport 36.4 -82.2  1593    -5 A     America/Ne~
```

```
print(head(weather))
```

```
## # A tibble: 6 x 15
##   origin  year month   day  hour  temp  dewp humid wind_dir wind_speed wind_gust
##   <chr>  <int> <int> <int> <int> <dbl> <dbl> <dbl>    <dbl>      <dbl>     <dbl>
## 1 EWR     2013     1     1     1  39.0  26.1  59.4      270       10.4         NA
## 2 EWR     2013     1     1     2  39.0  27.0  61.6      250        8.06        NA
## 3 EWR     2013     1     1     3  39.0  28.0  64.4      240       11.5         NA
## 4 EWR     2013     1     1     4  39.9  28.0  62.2      250       12.7         NA
## 5 EWR     2013     1     1     5  39.0  28.0  64.4      260       12.7         NA
## 6 EWR     2013     1     1     6  37.9  28.0  67.2      240       11.5         NA
## # i 4 more variables: precip <dbl>, pressure <dbl>, visib <dbl>,
## #   time_hour <dttm>
```

```
print(head(airlines))
```

```
## # A tibble: 6 x 2
##   carrier name
##   <chr>   <chr>
## 1 9E      Endeavor Air Inc.
## 2 AA      American Airlines Inc.
```

```
## 3 AS      Alaska Airlines Inc.
## 4 B6      JetBlue Airways
## 5 DL      Delta Air Lines Inc.
## 6 EV      ExpressJet Airlines Inc.
```

```
print(head(planes))
```

```
## # A tibble: 6 x 9
##   tailnum  year type              manufacturer model engines seats speed engine
##   <chr>   <int> <chr>             <chr>        <chr>   <int> <int> <int> <chr>
## 1 N10156   2004 Fixed wing multi ~ EMBRAER      EMB-~       2    55    NA Turbo~
## 2 N102UW   1998 Fixed wing multi ~ AIRBUS INDU~ A320~       2   182    NA Turbo~
## 3 N103US   1999 Fixed wing multi ~ AIRBUS INDU~ A320~       2   182    NA Turbo~
## 4 N104UW   1999 Fixed wing multi ~ AIRBUS INDU~ A320~       2   182    NA Turbo~
## 5 N10575   2002 Fixed wing multi ~ EMBRAER      EMB-~       2    55    NA Turbo~
## 6 N105UW   1999 Fixed wing multi ~ AIRBUS INDU~ A320~       2   182    NA Turbo~
```

```
# Comprobamos las variables de cada uno de los datasets que nos proporcionan
# mediante la instrucción "summary".
print(summary(flights))
```

```
##       year          month             day           dep_time    sched_dep_time
##  Min.   :2013   Min.   : 1.000   Min.   : 1.00   Min.   :   1   Min.   : 106
##  1st Qu.:2013   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 907   1st Qu.: 906
##  Median :2013   Median : 7.000   Median :16.00   Median :1401   Median :1359
##  Mean   :2013   Mean   : 6.549   Mean   :15.71   Mean   :1349   Mean   :1344
##  3rd Qu.:2013   3rd Qu.:10.000   3rd Qu.:23.00   3rd Qu.:1744   3rd Qu.:1729
##  Max.   :2013   Max.   :12.000   Max.   :31.00   Max.   :2400   Max.   :2359
##                                                  NA's   :8255
##    dep_delay          arr_time    sched_arr_time   arr_delay
##  Min.   : -43.00   Min.   :   1   Min.   :   1   Min.   : -86.000
##  1st Qu.:  -5.00   1st Qu.:1104   1st Qu.:1124   1st Qu.: -17.000
##  Median :  -2.00   Median :1535   Median :1556   Median :  -5.000
##  Mean   :  12.64   Mean   :1502   Mean   :1536   Mean   :   6.895
##  3rd Qu.:  11.00   3rd Qu.:1940   3rd Qu.:1945   3rd Qu.:  14.000
##  Max.   :1301.00   Max.   :2400   Max.   :2359   Max.   :1272.000
##  NA's   :8255      NA's   :8713                  NA's   :9430
##    carrier             flight       tailnum             origin
##  Length:336776      Min.   :   1   Length:336776      Length:336776
##  Class :character   1st Qu.: 553   Class :character   Class :character
##  Mode  :character   Median :1496   Mode  :character   Mode  :character
##                     Mean   :1972
##                     3rd Qu.:3465
##                     Max.   :8500
##
##      dest              air_time        distance         hour
##  Length:336776      Min.   : 20.0   Min.   :  17   Min.   : 1.00
##  Class :character   1st Qu.: 82.0   1st Qu.: 502   1st Qu.: 9.00
##  Mode  :character   Median :129.0   Median : 872   Median :13.00
##                     Mean   :150.7   Mean   :1040   Mean   :13.18
##                     3rd Qu.:192.0   3rd Qu.:1389   3rd Qu.:17.00
```

```
##                    Max.   :695.0   Max.   :4983   Max.   :23.00
##                    NA's   :9430
##      minute        time_hour
##  Min.   : 0.00   Min.   :2013-01-01 05:00:00
##  1st Qu.: 8.00   1st Qu.:2013-04-04 13:00:00
##  Median :29.00   Median :2013-07-03 10:00:00
##  Mean   :26.23   Mean   :2013-07-03 05:22:54
##  3rd Qu.:44.00   3rd Qu.:2013-10-01 07:00:00
##  Max.   :59.00   Max.   :2013-12-31 23:00:00
##
```

```
print(summary(airports))
```

```
##      faa                name               lat             lon
##  Length:1458        Length:1458        Min.   :19.72   Min.   :-176.65
##  Class :character   Class :character   1st Qu.:34.26   1st Qu.:-119.19
##  Mode  :character   Mode  :character   Median :40.09   Median : -94.66
##                                        Mean   :41.65   Mean   :-103.39
##                                        3rd Qu.:45.07   3rd Qu.: -82.52
##                                        Max.   :72.27   Max.   : 174.11
##       alt               tz             dst              tzone
##  Min.   : -54.00   Min.   :-10.000   Length:1458        Length:1458
##  1st Qu.:  70.25   1st Qu.: -8.000   Class :character   Class :character
##  Median : 473.00   Median : -6.000   Mode  :character   Mode  :character
##  Mean   :1001.42   Mean   : -6.519
##  3rd Qu.:1062.50   3rd Qu.: -5.000
##  Max.   :9078.00   Max.   :  8.000
```

```
print(summary(weather))
```

```
##     origin               year          month             day
##  Length:26115       Min.   :2013   Min.   : 1.000   Min.   : 1.00
##  Class :character   1st Qu.:2013   1st Qu.: 4.000   1st Qu.: 8.00
##  Mode  :character   Median :2013   Median : 7.000   Median :16.00
##                     Mean   :2013   Mean   : 6.504   Mean   :15.68
##                     3rd Qu.:2013   3rd Qu.: 9.000   3rd Qu.:23.00
##                     Max.   :2013   Max.   :12.000   Max.   :31.00
##
##       hour             temp             dewp            humid
##  Min.   : 0.00   Min.   : 10.94   Min.   :-9.94   Min.   : 12.74
##  1st Qu.: 6.00   1st Qu.: 39.92   1st Qu.:26.06   1st Qu.: 47.05
##  Median :11.00   Median : 55.40   Median :42.08   Median : 61.79
##  Mean   :11.49   Mean   : 55.26   Mean   :41.44   Mean   : 62.53
##  3rd Qu.:17.00   3rd Qu.: 69.98   3rd Qu.:57.92   3rd Qu.: 78.79
##  Max.   :23.00   Max.   :100.04   Max.   :78.08   Max.   :100.00
##                  NA's   :1        NA's   :1       NA's   :1
##     wind_dir       wind_speed       wind_gust         precip
##  Min.   :  0.0   Min.   :  0.000   Min.   :16.11   Min.   :0.000000
##  1st Qu.:120.0   1st Qu.:  6.905   1st Qu.:20.71   1st Qu.:0.000000
##  Median :220.0   Median :  10.357   Median :24.17   Median :0.000000
##  Mean   :199.8   Mean   :  10.518   Mean   :25.49   Mean   :0.004469
```

```
##  3rd Qu.:290.0   3rd Qu.:  13.809   3rd Qu.:28.77   3rd Qu.:0.000000
##  Max.   :360.0   Max.   :1048.361   Max.   :66.75   Max.   :1.210000
##  NA's   :460     NA's   :4          NA's   :20778
##     pressure          visib          time_hour
##  Min.   : 983.8   Min.   : 0.000   Min.   :2013-01-01 01:00:00
##  1st Qu.:1012.9   1st Qu.:10.000   1st Qu.:2013-04-01 21:30:00
##  Median :1017.6   Median :10.000   Median :2013-07-01 14:00:00
##  Mean   :1017.9   Mean   : 9.255   Mean   :2013-07-01 18:26:37
##  3rd Qu.:1023.0   3rd Qu.:10.000   3rd Qu.:2013-09-30 13:00:00
##  Max.   :1042.1   Max.   :10.000   Max.   :2013-12-30 18:00:00
##  NA's   :2729
```

```
print(summary(airlines))
```

```
##    carrier              name
##  Length:16          Length:16
##  Class :character   Class :character
##  Mode  :character   Mode  :character
```

```
print(summary(planes))
```

```
##    tailnum               year          type           manufacturer
##  Length:3322        Min.   :1956   Length:3322        Length:3322
##  Class :character   1st Qu.:1997   Class :character   Class :character
##  Mode  :character   Median :2001   Mode  :character   Mode  :character
##                     Mean   :2000
##                     3rd Qu.:2005
##                     Max.   :2013
##                     NA's   :70
##     model              engines          seats           speed
##  Length:3322        Min.   :1.000   Min.   :  2.0   Min.   : 90.0
##  Class :character   1st Qu.:2.000   1st Qu.:140.0   1st Qu.:107.5
##  Mode  :character   Median :2.000   Median :149.0   Median :162.0
##                     Mean   :1.995   Mean   :154.3   Mean   :236.8
##                     3rd Qu.:2.000   3rd Qu.:182.0   3rd Qu.:432.0
##                     Max.   :4.000   Max.   :450.0   Max.   :432.0
##                                                     NA's   :3299
##     engine
##  Length:3322
##  Class :character
##  Mode  :character
##
##
##
##
```

```
# Simplificar los dataframes originales a 100 observaciones. Renombrarlos
# introduciendo la coletilla "_simple".

flights_simple <- head(flights,100)
airports_simple <- head(airports,100)
```

```
weather_simple <- head(weather,100)
airlines_simple <- head(airlines,100)
planes_simple <- head(planes,100)


# Selecciona los tipos de aerolinea ("carrier") mediante la instruccion "select"
# y el operador "unique" concatenados con el operador tuberia %>%.
airlines_simple %>% unique %>% select(carrier)
```

```
## # A tibble: 16 x 1
##    carrier
##    <chr>
##  1 9E
##  2 AA
##  3 AS
##  4 B6
##  5 DL
##  6 EV
##  7 F9
##  8 FL
##  9 HA
## 10 MQ
## 11 OO
## 12 UA
## 13 US
## 14 VX
## 15 WN
## 16 YV
```

```
# Obtener la media y el maximo de asientos ("seats") que tienen los aviones.
# Utilizar el operador tuberia %>%.
planes_simple %>% summarise(mean = mean(seats),max_engines = max(seats))
```

```
## # A tibble: 1 x 2
##    mean max_engines
##   <dbl>       <int>
## 1  105.         330
```

```
# Ordenar los aviones por numero de motores ("engines") y numero de asientos ("seats").
result1 <- arrange(planes_simple,engines,seats)
print(result1)
```

```
## # A tibble: 100 x 9
##    tailnum  year type              manufacturer model engines seats speed engine
##    <chr>   <int> <chr>             <chr>        <chr>   <int> <int> <int> <chr>
##  1 N10156   2004 Fixed wing multi~ EMBRAER      EMB-~       2    55    NA Turbo~
##  2 N10575   2002 Fixed wing multi~ EMBRAER      EMB-~       2    55    NA Turbo~
##  3 N11106   2002 Fixed wing multi~ EMBRAER      EMB-~       2    55    NA Turbo~
##  4 N11107   2002 Fixed wing multi~ EMBRAER      EMB-~       2    55    NA Turbo~
##  5 N11109   2002 Fixed wing multi~ EMBRAER      EMB-~       2    55    NA Turbo~
##  6 N11113   2002 Fixed wing multi~ EMBRAER      EMB-~       2    55    NA Turbo~
```

```
##  7 N11119   2002 Fixed wing multi~ EMBRAER      EMB-~       2    55    NA Turbo~
##  8 N11121   2003 Fixed wing multi~ EMBRAER      EMB-~       2    55    NA Turbo~
##  9 N11127   2003 Fixed wing multi~ EMBRAER      EMB-~       2    55    NA Turbo~
## 10 N11137   2003 Fixed wing multi~ EMBRAER      EMB-~       2    55    NA Turbo~
## # i 90 more rows
```

```r
# Averigua que numero de cola comparten los dataframes "flights_simple"
# y "planes_simple" que has creado anteriormente.
# Obten su aerolinea ("carrier")
shared <- inner_join(flights_simple,planes_simple,by="tailnum") # -> N14228
shared_carrier <- shared$carrier
print(shared_carrier)
```

```
## [1] "EV"
```

```r
# Cruzar los datos de vuelos ("flights") con los aviones ("planes")
# por el numero de cola ("tailnum") que no coincidan.
# De esos obtener aquellos con 2 o mas motores.
# Finlmente obtener los distintos modelos de avión que satisfacen las premisas anteriores.
fp <- anti_join(planes_simple,flights_simple,by="tailnum")
engines_fp <- filter(fp,engines >= 2)
result2 <- unique(engines_fp$model) # No queremos los repetidos. Por lo tanto usamos "unique".
print(result2)
```

```
## [1] "EMB-145XR" "A320-214"  "EMB-145LR" "737-824"   "767-332"   "757-224"
```

```r
# Crea una nueva variable que calcule el retraso total sumando los
# delays acumulados ("dep_delay") y ("arr_delay").
# Almacena el dataframe resultante en "flights_total".
flights_total <- mutate(flights_simple,total_delay=dep_delay+arr_delay)

# En base a la variable anteriormente obtenida, devuelve los aviones que
# han llegado con antelación a su destino.
filter(flights_total,total_delay < 0)
```

```
## # A tibble: 57 x 20
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      544            545        -1     1004           1022
##  2  2013     1     1      554            600        -6      812            837
##  3  2013     1     1      557            600        -3      709            723
##  4  2013     1     1      557            600        -3      838            846
##  5  2013     1     1      558            600        -2      849            851
##  6  2013     1     1      558            600        -2      853            856
##  7  2013     1     1      558            600        -2      923            937
##  8  2013     1     1      559            559         0      702            706
##  9  2013     1     1      559            600        -1      854            902
## 10  2013     1     1      600            600         0      851            858
## # i 47 more rows
## # i 12 more variables: arr_delay <dbl>, carrier <chr>, flight <int>,
## #   tailnum <chr>, origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>,
## #   hour <dbl>, minute <dbl>, time_hour <dttm>, total_delay <dbl>
```