Tarea - Calculando nuevas variables con dplyr

Jesus Mudarra Luján

2022-11-11

Pregunta 1

El dataset de vuelos tiene dos variables, `dep_time` y `sched_dep_time` muy útiles pero difíciles de usar por cómo vienen dadas al no ser variables continuas. Fíjate que cuando pone 559, se refiere a que el vuelo salió a las 5:59...

Convierte este dato en otro más útil que represente el número de minutos que han transcurrido desde media noche.

```
mutate(flights,
       dep_time = dep_time%/%100*60 + dep_time%%100,
       sched_dep_time = sched_dep_time%/%100*60 + sched_dep_time%%100
       )
```

```
## # A tibble: 336,776 x 19
##     year month   day dep_time sched_de~1 dep_d~2 arr_t~3 sched~4 arr_d~5 carrier
##    <int> <int> <int>    <dbl>      <dbl>   <dbl>   <int>   <int>   <dbl> <chr>
##  1  2013     1     1      317        315       2     830     819      11 UA
##  2  2013     1     1      333        329       4     850     830      20 UA
##  3  2013     1     1      342        340       2     923     850      33 AA
##  4  2013     1     1      344        345      -1    1004    1022     -18 B6
##  5  2013     1     1      354        360      -6     812     837     -25 DL
##  6  2013     1     1      354        358      -4     740     728      12 UA
##  7  2013     1     1      355        360      -5     913     854      19 B6
##  8  2013     1     1      357        360      -3     709     723     -14 EV
##  9  2013     1     1      357        360      -3     838     846      -8 B6
## 10  2013     1     1      358        360      -2     753     745       8 AA
## # ... with 336,766 more rows, 9 more variables: flight <int>, tailnum <chr>,
## #   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
## #   minute <dbl>, time_hour <dttm>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```

Pregunta 2

Compara las variables `air_time` contra `arr_time - dep_time`.

- ¿Qué esperas ver?

  Se espera ver el mismo resultado, ya que ambas opciones presentan el tiempo que se encuentra el avión en el aire.

```
transmute(flights,
          air_time,
          arr_time,
```

```
          dep_time,
          arr_time - dep_time)
```

```
## # A tibble: 336,776 x 4
##    air_time arr_time dep_time `arr_time - dep_time`
##       <dbl>    <int>    <int>                 <int>
##  1      227      830      517                   313
##  2      227      850      533                   317
##  3      160      923      542                   381
##  4      183     1004      544                   460
##  5      116      812      554                   258
##  6      150      740      554                   186
##  7      158      913      555                   358
##  8       53      709      557                   152
##  9      140      838      557                   281
## 10      138      753      558                   195
## # ... with 336,766 more rows
```

- ¿Qué ves realmente?

  Lo que realmente ocurre es que no se encuentran en el mismo formato. Mientras que `air_time` es el tiempo en el aire en minutos, `arr_time - dep_time` es tiempo en el aire expresado en formato (HHMM o HMM)

- ¿Se te ocurre algo para mejorarlo y corregirlo?

```
transmute(flights,
          arr_time = arr_time%/%100*60 + arr_time%%100,
          dep_time = dep_time%/%100*60 + dep_time%%100,
          air_time= arr_time - dep_time)
```

```
## # A tibble: 336,776 x 3
##    arr_time dep_time air_time
##       <dbl>    <dbl>    <dbl>
##  1      510      317      193
##  2      530      333      197
##  3      563      342      221
##  4      604      344      260
##  5      492      354      138
##  6      460      354      106
##  7      553      355      198
##  8      429      357       72
##  9      518      357      161
## 10      473      358      115
## # ... with 336,766 more rows
```

Pregunta 3

Compara los valores de `dep_time`, `sched_dep_time` y `dep_delay`. Cómo deberían relacionarse estos tres números? Compruébalo y haz las correcciones numéricas que necesitas.

```
transmute(flights,
          dep_time = dep_time%/%100*60 + dep_time%%100,
          sched_dep_time = sched_dep_time%/%100*60 + sched_dep_time%%100,
          dep_delay,
          sched_dep_time_with_delay = sched_dep_time + dep_delay)
```

```
## # A tibble: 336,776 x 4
##    dep_time sched_dep_time dep_delay sched_dep_time_with_delay
##       <dbl>          <dbl>     <dbl>                     <dbl>
##  1      317            315         2                       317
##  2      333            329         4                       333
##  3      342            340         2                       342
##  4      344            345        -1                       344
##  5      354            360        -6                       354
##  6      354            358        -4                       354
##  7      355            360        -5                       355
##  8      357            360        -3                       357
##  9      357            360        -3                       357
## 10      358            360        -2                       358
## # ... with 336,766 more rows
```

Pregunta 4

Usa una de las funciones de ranking para quedarte con los 10 vuelos más retrasados de todos.

```
delay.rank<-arrange(mutate(flights,
               m_r=min_rank(desc(dep_delay))
               ),
        m_r
        )
library(knitr)
head(delay.rank, 10)
```

```
## # A tibble: 10 x 20
##     year month   day dep_time sched_de~1 dep_d~2 arr_t~3 sched~4 arr_d~5 carrier
##    <int> <int> <int>    <int>      <int>   <dbl>   <int>   <int>   <dbl> <chr>
##  1  2013     1     9      641        900    1301    1242    1530    1272 HA
##  2  2013     6    15     1432       1935    1137    1607    2120    1127 MQ
##  3  2013     1    10     1121       1635    1126    1239    1810    1109 MQ
##  4  2013     9    20     1139       1845    1014    1457    2210    1007 AA
##  5  2013     7    22      845       1600    1005    1044    1815     989 MQ
##  6  2013     4    10     1100       1900     960    1342    2211     931 DL
##  7  2013     3    17     2321        810     911     135    1020     915 DL
##  8  2013     6    27      959       1900     899    1236    2226     850 DL
##  9  2013     7    22     2257        759     898     121    1026     895 DL
## 10  2013    12     5      756       1700     896    1058    2020     878 AA
## # ... with 10 more variables: flight <int>, tailnum <chr>, origin <chr>,
## #   dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>,
## #   time_hour <dttm>, m_r <int>, and abbreviated variable names
## #   1: sched_dep_time, 2: dep_delay, 3: arr_time, 4: sched_arr_time,
## #   5: arr_delay
```

Pregunta 5

Aunque la ejecución te dé una advertencia, qué resultado te da la operación

1:6 + 1:20

```
1:6 + 1:20
```

```
## Warning in 1:6 + 1:20: longitud de objeto mayor no es múltiplo de la longitud de
## uno menor
```

```
##  [1]  2  4  6  8 10 12  8 10 12 14 16 18 14 16 18 20 22 24 20 22
```

Cuando acaba la secuencia 1:6 vuelve a empezar, es decir, que a partir del séptimo elemento se sumaría 1 + 7, 2 + 8, 3 + 9, . . . , 2 + 20.

Pregunta 6

Además de todas las funciones que hemos dicho, las trigonométricas también son funciones vectoriales que podemos usar para hacer transformaciones con `mutate`. Investiga cuales trae `R` y cual es la sintaxis de cada una de ellas.

```
?Trig
```

```
## starting httpd help server ... done
```

Estas funciones dan las funciones básicas trigonométricas. Calculan respectivamente el coseno, el seno, la tangente, el arcocoseno, el arcoseno, el arcotangente y el arcotangente de dos argumentos.

```
cos(x)
sin(x)
tan(x)

acos(x)
asin(x)
atan(x)
atan2(y, x)

cospi(x)
sinpi(x)
tanpi(x)
```