

Carreras deportivas en España en 2019

Autores: Eugenio Carmona Soriano y Antonio Ruíz Falcó Rojas

Mayo 2019

Contents

Descripción del DataSet.	1
Datos de Interés.	2
Lectura del fichero.	2
Limpieza de los datos.	3
Datos vacíos y otros tratamientos de limpieza.	3
Identificación y tratamiento de datos extremos.	10
Análisis de los datos	11
Selección de los grupos de datos	11
Comprobación de la normalidad y homogeneidad de la varianza	11
Aplicación de pruebas	14
Representación Gráfica	22
Gráficos de tartas	22
Diagramas de barras	25
Diagrama de pruebas por estación y categoría	25
Diagrama de pruebas por tipo y categoría	26
Conclusiones	27
Contribuciones	27

Descripción del DataSet.

El conjunto de datos utilizado en esta actividad es el generado en la primera práctica mediante web scraping, y recoge los eventos de carreras deportivas que se realizan en España el año 2019. Las variables que se recogen en el conjunto de datos son la fecha, el nombre, el lugar, el tipo de carrera, la distancia, la página web de la carrera y si incluye categorías infantiles.

Para cada carrera, se crea un registro en el conjunto de datos que recogen los siguientes campos para todo el año 2019:

- **Fecha:** el día en el que se realiza la carrera en formato dd/mm/aaaa.
- **Carrera:** Nombre de la carrera.
- **Ciudad:** Ciudad en la que se realiza la carrera.
- **Provincia:** Provincia en la que se realiza la carrera.
- **Tipo:** Tipo de carrera. Ruta, Trail, Triatlón, Ciclismo, Duatlón, Obstáculos y Cross/Tierra.
- **Distancia:** La distancia a recorrer en la carrera.
- **Web:** Enlace a la página web de la carrera.
- **Infantil:** Si o No. Según incluya o no categorías infantiles.

Los datos son importantes, pues el conjunto de datos se corresponde con las competiciones deportivas, más concretamente carreras deportivas, que se realizan en España el año 2019. Se incluyen varios tipos de carreras entre los que encuentran: Rutas, Trails, Triatlón, Ciclismo, Duatlón, Obstáculos y Cross/Tierra.

La página web elegida para obtener la información la ofrece a título informativo a todo aquel que quiera conocer las diferentes competiciones que se realizan, el lugar y la fecha. Además de ofrecer el enlace a la página web oficial de cada carrera.

Es evidente que el *footing* o *running* es la actividad física básica. El estudio de los datos permitirá adquirir conocimiento sobre la práctica de deporte popular. Por ejemplo, permitirá estudiar si la actividad física cambia de unas regiones a otras, la distribución de pruebas a lo largo del año, diferenciación entre las competiciones de niños y adultos, etc. En el presente trabajo se realizarán los siguientes estudios:

- Detección de las **reglas de asociación** entre mes de la prueba, provincia de la prueba, tipo de carrera, distancia y si es infantil o no.
- Detección de grupos mediante **Árboles de decisión** para los campos anteriores.
- Estudio de **regresión múltiple**.

Datos de Interés.

En el apartado anterior se ha mostrado que el conjunto de datos tiene ocho atributos, pero no todos ellos son igual de útiles para proporcionar conocimiento:

- **Fecha** muestra la fecha de la celebración de la carrera. El estudio de fechas concretas de celebración de carreras carece de interés, más allá de que si se realizara una búsqueda el 31 de diciembre se encontrarían las clásicas de San Silvestre. Sin embargo, el tratamiento agregado de fechas puede dar información realmente útil. A priori, podrían pensarse en tres tipos de agrupaciones de fechas: agrupación por mes, agrupación por trimestre y agrupación por estación del año. Cualquiera de ellas puede proporcionar información valiosa. Para este estudio utilizaremos la agregación por estación (invierno, primavera, verano, otoño). Como es lógico, la variable resultante será discreta.
- El **Nombre** de la carrera es útil para distinguir una carrera de otra, o como mnemotécnico para referirse a una carrera. Sin embargo, no proporciona información útil para el estudio.
- La **Ciudad** aporta información geográfica, por lo que es útil. No obstante, existe la columna **Provincia**, que no deja de ser agregación de ciudades. Por tanto, se utilizará provincia para el estudio.
- **Tipo** es una variable discreta que aporta información útil sobre la modalidad de la prueba, por lo que debe formar parte del estudio.
- **Distancia**: es la variable fundamental para el estudio.
- **Web**: enlace a la página web de la carrera. No se utilizará para el estudio.
- **Infantil**: variable discreta que representa la categoría de la carrera: infantil o adultos. Participará en el estudio.

Por tanto, de todos los atributos del dataset, los que se utilizarán para el estudio son **Estación** (se convertirá la fecha en la estación del año a la que pertenece), **Provincia**, **Tipo**, **Distancia** e **Infantil**. No se utilizarán el Nombre, Ciudad y Web.

Lectura del fichero.

En primer lugar, hay que leer el fichero `runnings.csv`, que se obtuvo utilizando técnicas de web scraping sobre la página web <https://www.corriendovoy.com/calendario-carreras>. Para la lectura, hay que tener en cuenta que el separador de columnas es “;” (punto y coma) y no “,” (coma).

```
# Se carga el archivo de datos
datos<-read.csv("../csv/runnings.csv", header=T, sep=";", encoding="UTF-8")
attach(datos)
# Se visualizan los datos fundamentales del archivo.
dim(datos)
```

```
## [1] 5550      8
```

```
str(datos)
```

```
## 'data.frame':    5550 obs. of  8 variables:
## $ Fecha      : Factor w/ 192 levels "2019-01-01 00:00:00",...: 1 2 3 3 3 3 3 3 3 ...
## $ Carrera    : Factor w/ 5516 levels "'Ultra Trail Montnegre-Corredor",...: 4492 2961 1018 2113 2048 2...
## $ Ciudad     : Factor w/ 2113 levels "A Coruña","A determinar",...: 654 1391 2100 1674 1376 164 164 60...
## $ Provincia  : Factor w/ 57 levels "A Coruña","Albacete",...: 41 42 39 10 28 33 33 10 35 35 ...
## $ Tipo       : Factor w/ 7 levels "Ciclismo","Cross",...: 5 5 5 5 5 5 5 5 5 ...
## $ Distancia  : Factor w/ 621 levels "0,054 m","0,200 m",...: 377 433 531 431 4 489 329 431 14 431 ...
## $ Web        : Factor w/ 2019 levels " http://www.circuitlacostera.org/",...: 1609 1715 622 1845 229 1...
## $ Infantil   : Factor w/ 2 levels "False","True": 2 1 1 1 1 1 1 2 2 2 ...
```

Como puede apreciarse, el fichero contiene 5550 registros además del registro de cabecera. Todos los atributos son leídos como factor, lo que es lógico dada su naturaleza.

Limpieza de los datos.

Datos vacíos y otros tratamientos de limpieza.

Para la limpieza de datos se utilizarán los siguientes criterios:

- Se analizarán los datos vacíos en todos los casos y se actuará en consecuencia.
- Todas las cadenas de caracteres se normalizarán a mayúsculas.
- En las variables discretas se analizarán los casos existentes, posibles duplicidades por errores tipográficos, etc.
- Se tratarán todos los campos con los criterios mencionados, incluso los que no participarán en el estudio, por si en un momento posterior se decide que participen en el análisis.

A continuación se realiza la limpieza de los ocho campos del dataset.

Atributo Fecha

Fecha no contiene datos vacíos. Ahora bien, es necesario realizar el agrupamiento mencionado en el apartado anterior para agrupar por estación del año.

R ha tomado el campo fecha como un factor. El formato del mismo es “AAAA-MM-DD hh:mm:ss”. Para el tratamiento deseado, debe convertirse el campo a la estación del año correspondiente. Para hacerlo, en vez de convertir la columna fecha (y perderla), se creará una nueva columna. Para realizar la conversión se crea la función `toSeason`, que recibe como entrada el vector de fechas a convertir en la estación correspondiente. El propósito es crear una columna nueva, llamada **Estacion**, que será un factor con cuatro posibilidades: SPRING, SUMMER, FALL y WINTER.

```
# En primer lugar, se define la función toSeason para convertir la fecha en la estación anual
toSeason <- function(dat) {

  scalarCheck <- function(dat) {
    m <- as.POSIXlt(dat)$mon + 1      # correct for 0:11 range
    d <- as.POSIXlt(dat)$mday         # correct for 0:11 range
    if ((m == 3 & d >= 21) | (m == 4) | (m == 5) | (m == 6 & d < 21)) {
      r <- 1
    } else if ((m == 6 & d >= 21) | (m == 7) | (m == 8) | (m == 9 & d < 21)) {
      r <- 2
    } else if ((m == 9 & d >= 21) | (m == 10) | (m == 11) | (m == 12 & d < 21)) {
      r <- 3
    } else {

```

```

    r <- 4
  }
  r
}

res <- sapply(dat, scalarCheck)
res <- ordered(res, labels=c("SPRING", "SUMMER", "FALL", "WINTER"))
invisible(res)
}
# Se llama a la función para crear la nueva columna.
datos$Estacion<-toSeason(datos$Fecha)
# Se muestra el resumen de la tabla para ver el resumen de casos.
summary(datos)

```

```

##          Fecha                      Carrera
## 2019-12-31 00:00:00: 343  Binter Night Run 10k      : 3
## 2019-04-07 00:00:00: 240  Binter Night Run 5k       : 3
## 2019-03-31 00:00:00: 207  Carreras Alimentos de España 10k: 3
## 2019-03-24 00:00:00: 172  Carreras Alimentos de España 5k : 3
## 2019-03-10 00:00:00: 170  100% Tri Platja d'Aro      : 2
## 2019-03-17 00:00:00: 168  Andaina Ría de Noia 25k    : 2
## (Other)      :4250  (Other)                      :5534
##      Ciudad      Provincia      Tipo      Distancia
## Madrid   : 203  Madrid   : 473  Ciclismo : 668  10 Km   : 680
## Valencia : 60  Valencia : 439  Cross   : 339  5 Km    : 608
## Zaragoza : 36  Barcelona: 354  Duatlón : 335  21,097 m: 202
## Barcelona: 34  Alicante : 209  Obstácul.: 103  6 Km    : 190
## Alicante : 32  Murcia   : 191  Running :2328  8 Km    : 135
## Ibiza    : 30  Girona   : 183  Trail   :1432  27,500 m: 127
## (Other)  :5155  (Other) :3701  Triatlón : 345  (Other) :3608
##                               Web      Infantil
## http://www.dipualba.es/carrerasPopulares/#: 65  False:1825
## http://elitechip.net/                        : 62  True :3725
## http://www.triatloncastillayleon.com/        : 52
## http://runcancer.com/                       : 43
## http://runvasport.es/                      : 43
## http://www.circuitobtt.com/cbtt2019/Inicio: 39
## (Other)                                     :5246
##      Estacion
## SPRING:2121
## SUMMER: 474
## FALL   : 529
## WINTER:2426
##
##
##

```

Una vez ejecutada la función `toSeason` para convertir la fecha a la estación correspondiente, se utiliza la función `summary()` para ver el resumen de los datos. Como se aprecia en los resultados, con algo tan simple ya se ha obtenido una información muy significativa, pues se ve claramente que la primavera y el invierno se concentran la mayoría de pruebas (2.121 y 2.426 respectivamente), mientras que llama la atención el reducido número de pruebas que hay en verano (474) y otoño (529).

Atributo Carrera

Este atributo contiene el nombre de la carrera y no se utilizará en el presente estudio. No obstante, se convertirá a mayúsculas para homogeneizar el campo por si se deseara para algún tratamiento posterior.

```
# Se convierte Carrera a mayúsculas
datos$Carrera <- as.factor(toupper(datos$Carrera))
```

Atributo Ciudad

Este atributo contiene el nombre de la ciudad donde se celebra la carrera y no se utilizará en el presente estudio, porque se utilizará el dato agregado de provincia. No obstante, se convertirá a mayúsculas para homogeneizar el campo por si se deseara para algún tratamiento posterior.

```
# Se convierte Ciudad a Mayúsculas
datos$Ciudad <- as.factor(toupper(datos$Ciudad))
```

Atributo Provincia

Este atributo contiene el nombre de la provincia donde se celebra la carrera. Se utilizará en el estudio, y tal como se ha indicado se convertirá a mayúsculas. Una vez hecha la conversión, se llama a la función `summary(datos$Provincia)` para ver los resultados:

```
# Se convierte Provincia a Mayúsculas
datos$Provincia <- as.factor(toupper(datos$Provincia))

# Se muestran los resultados
summary(datos$Provincia)
```

##	A CORUÑA	ALBACETE	ALICANTE
##	116	144	209
##	ALMERÍA	ANDORRA	ARABA/ÁLAVA
##	67	12	31
##	ASTURIAS	ÁVILA	BADAJOS
##	152	43	93
##	BARCELONA	BIZKAIA	BURGOS
##	354	59	84
##	CÁCERES	CÁDIZ	CANTABRIA
##	59	90	89
##	CASTELLÓN	CEUTA	CIUDAD REAL
##	112	10	76
##	CÓRDOBA	CUENCA	FRANCIA
##	88	60	1
##	GIPUZKOA	GIRONA	GRANADA
##	98	183	99
##	GUADALAJARA	HUELVA	HUESCA
##	59	89	62
##	ILLES BALEARS	JAÉN	LA RIOJA
##	172	81	56
##	LAS PALMAS	LEÓN	LLEIDA
##	92	77	83
##	LUGO	MADRID	MÁLAGA
##	46	473	147
##	MARRUECOS	MELILLA	MURCIA
##	2	13	191
##	NAVARRA	OURENSE	PALENCIA

##	103	54	48
##	PONTEVEDRA	PORTUGAL	REST OF THE WORLD
##	63	24	20
##	SALAMANCA SANTA CRUZ	TENERIFE	SEGOVIA
##	58	82	73
##	SEVILLA	SORIA	TARRAGONA
##	93	20	127
##	TERUEL	TOLEDO	VALENCIA
##	74	83	439
##	VALLADOLID	ZAMORA	ZARAGOZA
##	78	29	110

Al ver el `summary(datos$Provincia)` se extraen varias conclusiones. La primera y más importante, es que la conversión se ha realizado de forma correcta. También se observa que en los datos no sólo hay pruebas españolas, pues figuran 20 registros de “REST OF THE WORLD” y 2 registros “MARRUECOS”, 24 de Portugal, 1 de Francia y 12 de Andorra. A continuación, se asignan esos registros como “REST OF THE WORLD”, de forma que el campo provincia sea o bien una provincia española o bien el resto del mundo.

```
# Se convierte "MARRUECOS" a "REST OF THE WORLD"
datos$Provincia <- as.factor(gsub("MARRUECOS", "REST OF THE WORLD", datos$Provincia))
# Se convierte "PORTUGAL" a "REST OF THE WORLD"
datos$Provincia <- as.factor(gsub("PORTUGAL", "REST OF THE WORLD", datos$Provincia))
# Se convierte "ANDORRA" a "REST OF THE WORLD"
datos$Provincia <- as.factor(gsub("ANDORRA", "REST OF THE WORLD", datos$Provincia))
# Se convierte "FRANCIA" a "REST OF THE WORLD"
datos$Provincia <- as.factor(gsub("FRANCIA", "REST OF THE WORLD", datos$Provincia))
# Se muestran los resultados
summary(datos$Provincia)
```

##	A CORUÑA	ALBACETE	ALICANTE
##	116	144	209
##	ALMERÍA	ARABA/ÁLAVA	ASTURIAS
##	67	31	152
##	ÁVILA	BADAJOS	BARCELONA
##	43	93	354
##	BIZKAIA	BURGOS	CÁCERES
##	59	84	59
##	CÁDIZ	CANTABRIA	CASTELLÓN
##	90	89	112
##	CEUTA	CIUDAD REAL	CÓRDOBA
##	10	76	88
##	CUENCA	GIPUZKOA	GIRONA
##	60	98	183
##	GRANADA	GUADALAJARA	HUELVA
##	99	59	89
##	HUESCA	ILLES BALEARS	JAÉN
##	62	172	81
##	LA RIOJA	LAS PALMAS	LEÓN
##	56	92	77
##	LLEIDA	LUGO	MADRID
##	83	46	473
##	MÁLAGA	MELILLA	MURCIA
##	147	13	191
##	NAVARRA	OURENSE	PALENCIA
##	103	54	48

```
##          PONTEVEDRA  REST OF THE WORLD          SALAMANCA
##              63              59              58
## SANTA CRUZ TENERIFE          SEGOVIA          SEVILLA
##              82              73              93
##              SORIA          TARRAGONA          TERUEL
##              20              127              74
##              TOLEDO          VALENCIA          VALLADOLID
##              83              439              78
##              ZAMORA          ZARAGOZA
##              29              110
```

A simple vista, llama la atención que Valencia, con una población inferior a Barcelona y un clima similar, tiene un número de carreras superior. De hecho, el número de pruebas en Valencia (439) se asemeja al de Madrid (473)

Atributo Tipo

Este atributo contiene el tipo de la prueba que se celebrará. Se utilizará en el estudio, y tal como se ha indicado se convertirá a mayúsculas. Una vez hecha la conversión, se llama a la función `summary(datos$Tipo)` para ver los resultados:

```
# Se convierte Tipo a Mayúsculas
datos$Tipo <- as.factor(toupper(datos$Tipo))
# Se muestran los resultados
summary(datos$Tipo)
```

```
## CICLISMO      CROSS    DUATLÓN OBSTÁCUL.  RUNNING      TRAIL  TRIATLÓN
##          668        339        335        103        2328        1432        345
```

En los resultados se muestran los siete tipos posibles de pruebas que hay en el dataset: 668 pruebas de ciclismo, 339 de cross, 335 de duatlón, 103 de obstáculos, 2328 de running, 1432 de trail y 345 triatlón.

Atributo Distancia

No contiene datos vacíos y se utilizará en el estudio. Además, deben homogeneizarse las unidades: la columna es un campo de texto en el que expresa distancia en diferentes formatos y unidades: km, metros. Debe elegirse un formato numérico (por ejemplo, normalizar todas las medidas a metros) y realizar la conversión correspondiente. Una vez que se hayan homogeneizado las distancias a la misma unidad, hay que convertir la columna a enteros.

```
# Se convierte " km" a "000"
datos$Distancia <- as.factor(gsub(" Km", "000", datos$Distancia))
# Se suprime " m"
datos$Distancia <- as.factor(gsub(" m", "", datos$Distancia))
# Se quita la coma que separa las unidades de millar
datos$Distancia <- as.factor(gsub(",", "", datos$Distancia))
# Se muestran los resultados
summary(datos$Distancia)
```

```
##    10000    5000    21097    6000    8000    27500    25750    21000    7000
##      680     608     202     190     135     127     116     113     106
##   12000   40000   11000   4000   15000   14000   30000   13000   25000
##      96      74      68      68      67      64      62      61      60
##  42195    9000    3000   42000   1609   20000   51500   22000   60000
##      60      60      58      55      50      43      43      41      41
##   7500   16000   23000   29000   18000   50000   27000   36000   4500
##      40      39      37      37      35      35      34      34      33
```

```
## 24000 55000 26000 6500 112900 32000 43000 35000 28000
## 30 30 29 29 28 28 28 27 25
## 5500 19000 17000 31000 3500 45000 8500 2000 52000
## 25 23 20 19 19 18 18 17 17
## 80000 100000 13500 70000 7200 48000 12500 25500 5400
## 17 16 15 15 15 14 13 13 13
## 200000 37000 44000 5100 5572 5800 6400 11500 12750
## 12 11 11 11 11 11 11 10 10
## 14500 26500 34000 38000 39000 4400 4800 6300 65000
## 10 10 10 10 10 10 10 10 10
## 8800 9500 102000 13750 28500 5600 56000 10500 2700
## 10 10 9 9 9 9 9 8 8
## 3200 33000 3600 4900 5200 5300 6200 6700 7700
## 8 8 8 8 8 8 8 8 8
## (Other)
## 976
```

En este punto ya están los datos listos para ser convertidos a numéricos, con las unidades homogeneizadas a metros. Se observa que algunos organizadores han utilizado unidades de medida diferentes para pruebas similares. Esto se ve bien claro en algunos ejemplo de Maratón. Los organizadores que han utilizado km han puesto “42km”, mientras que los que han utilizado metros han puesto la cifra exacta de 42.195m. Dado que se trata de la cifra proporcionada por los organizadores y que será tratada numéricamente, se asumen esos pequeños errores.

```
# Se quitan los ceros a la izquierda para la conversión a enteros.
datos$Distancia <- gsub("(^[^0-9])0+", "\\1", datos$Distancia, perl = TRUE)
# Se convierte datos$Distancia a entero
datos$Distancia <- strtoi(datos$Distancia)
# Se muestran los resultados
summary(datos$Distancia)
```

```
## Min. 1st Qu. Median Mean 3rd Qu. Max.
## 54 6552 12000 23568 27000 725000
```

Como se ve en el sumario, la distancia mínima son 54 metros, y la máxima 725 kilómetros. La mediana son 12 kilómetros y la media son algo más de 23km.

Atributo Web

Este atributo contiene la url de la carrera, donde puede obtenerse toda la información así como inscribirse en la misma. No se utilizará en el presente estudio, porque se utilizará el dato agregado de provincia. No obstante, se convertirá a mayúsculas para homogeneizar el campo por si se deseara para algún tratamiento posterior.

```
# Se convierte la Web a Mayúsculas
datos$Web<- as.factor(toupper(datos$Web))
```

Atributo Infantil

Es un factor que acepta dos valores: true si la prueba es infantil y false si no lo es. No hay datos vacíos ni de ninguna otra índole, así que la única conversión necesaria es la conversión a mayúsculas para homogeneizar con el resto.

```
# Se convierte la Web a Mayúsculas
datos$Infantil <- as.factor(toupper(datos$Infantil))
```


Resumen

Una vez realizado todos los procesos de limpieza necesarios en todos los campos, a continuación se muestra el resumen del dataset y los primeros registros del mismo.

```
# Se muestra el resumen de los datos
summary(datos)
```

```
##              Fecha              Carrera
## 2019-12-31 00:00:00: 343  BINTER NIGHT RUN 10K      : 3
## 2019-04-07 00:00:00: 240  BINTER NIGHT RUN 5K      : 3
## 2019-03-31 00:00:00: 207  CARRERAS ALIMENTOS DE ESPAÑA 10K: 3
## 2019-03-24 00:00:00: 172  CARRERAS ALIMENTOS DE ESPAÑA 5K : 3
## 2019-03-10 00:00:00: 170  100% TRI PLATJA D'ARO      : 2
## 2019-03-17 00:00:00: 168  ANDAINA RÍA DE NOIA 25K     : 2
## (Other)      :4250  (Other)                        :5534
##      Ciudad      Provincia      Tipo      Distancia
## MADRID : 203  MADRID : 473  CICLISMO : 668  Min. : 54
## VALENCIA : 60  VALENCIA : 439  CROSS : 339  1st Qu.: 6552
## ZARAGOZA : 36  BARCELONA: 354  DUATLÓN : 335  Median : 12000
## BARCELONA: 34  ALICANTE : 209  OBSTÁCUL.: 103  Mean : 23568
## ALICANTE : 32  MURCIA : 191  RUNNING :2328  3rd Qu.: 27000
## IBIZA : 30  GIRONA : 183  TRAIL :1432  Max. :725000
## (Other) :5155  (Other) :3701  TRIATLÓN : 345
##              Web      Infantil
## HTTP://WWW.DIPUALBA.ES/CARRERASPOPULARES/#: 65  FALSE:1825
## HTTP://ELITECHIP.NET/ : 62  TRUE :3725
## HTTP://WWW.TRIATLONCASTILLAYLEON.COM/ : 52
## HTTP://RUNCANCER.COM/ : 43
## HTTP://RUNVASPORT.ES/ : 43
## HTTP://WWW.CIRCUITOBTT.COM/CBTT2019/INICIO: 39
## (Other) :5246
##      Estacion
## SPRING:2121
## SUMMER: 474
## FALL : 529
## WINTER:2426
##
##
##
```

```
# Se muestran los primeros registros del dataset
head(datos)
```

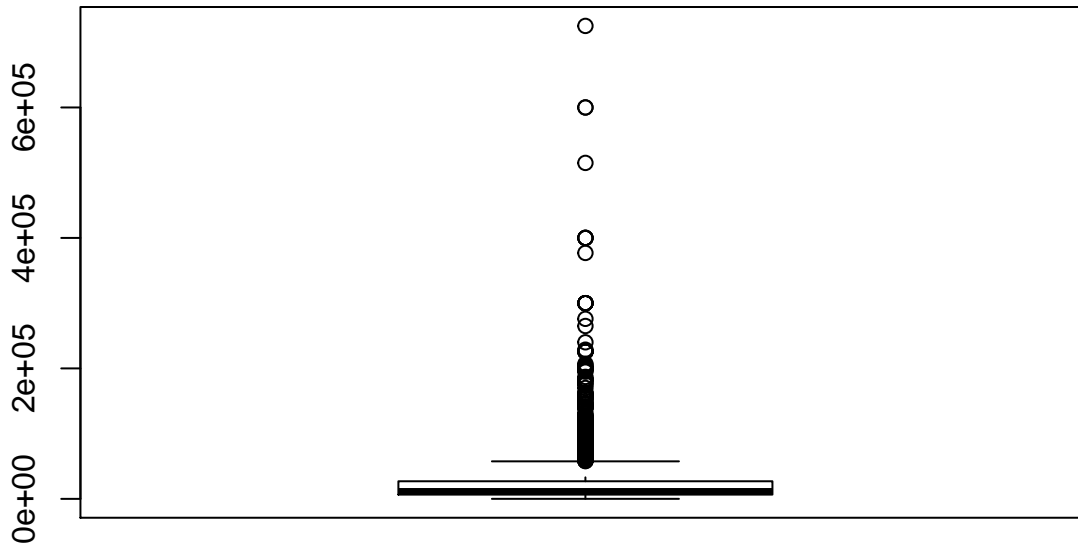
```
##              Fecha              Carrera
## 1 2019-01-01 00:00:00  SUBIDA A COSTIÑA DE CANEDO
## 2 2019-01-04 00:00:00  LEGUA DE NAVIDAD "LAS 4 TORRES" PAREDES DE NAVA
## 3 2019-01-05 00:00:00  CARRERA POPULAR DE ZARANDONA
## 4 2019-01-05 00:00:00  CURSA REIS
## 5 2019-01-05 00:00:00  CURSA INFANTIL DE REIS
## 6 2019-01-05 00:00:00  LA CORDERA REIS RACE 6KM
##      Ciudad      Provincia      Tipo      Distancia
## 1      EIRAS VEDRAS  OURENSE RUNNING      4000
## 2      PAREDES DE NAVA  PALENCIA RUNNING      5100
## 3      ZARANDONA      MURCIA RUNNING      7200
## 4  SANT JULIÀ DE CERDANYOLA  BARCELONA RUNNING      5000
```

```
## Web
## 1 HTTPS://DEPORTES.DEPOURENSE.ES/
## 2 HTTPS://VILLADEPAREDES.BLOGSPOT.COM.ES/
## 3 HTTP://WWW.ASUSPUESTOS.COM/
## 4 HTTPS://WWW.FACEBOOK.COM/CURSAREIS/
## 5 HTTP://ELITECHIP.NET/
## 6 HTTPS://RUNEDIA.MUNDODEPORTIVO.COM/CALENDARIO-CARRERAS/ESPANA/CATALUNA/LERIDA/
## Infantil Estacion
## 1 TRUE WINTER
## 2 FALSE WINTER
## 3 FALSE WINTER
## 4 FALSE WINTER
## 5 FALSE WINTER
## 6 FALSE WINTER
```

Identificación y tratamiento de datos extremos.

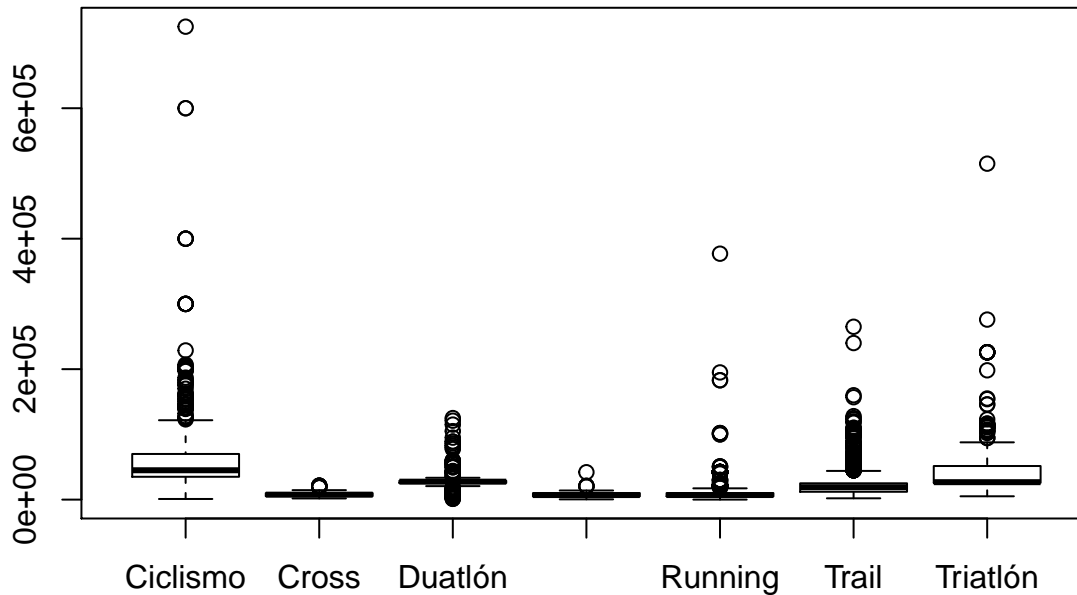
El único campo que puede tener valores extremos es Distancia, pues el resto son factores con un número de posibilidades limitada. Para detectar outliers, la forma más fácil es mediante un boxplot:

```
# Detección de outliers en el campo Distancia
boxplot(datos$Distancia)
```



No obstante, hacerlo así no tiene mucho sentido. Una distancia de 200km puede ser un outlier si se trata de una competición de footing, pero puede darse en ciclismo. Por tanto, la forma de analizar los outliers debe ser por tipo de prueba:

```
# Detección de outliers en el campo Distancia
boxplot(Distancia ~ Tipo, data = datos, names=c("Ciclismo", "Cross", "Duatlón", "Obstác.", "Running", "
```



Como se ve en el gráfico, hay outliers en Ciclismo, Running, Trail y Triatlón. La duda es de si son realmente outliers o se trata de pruebas muy extremas. Es muy significativo que en Cross, Duatlón y Obstáculos no existen, por lo que lo más probable es que se traten de pruebas extremas que están muy de moda últimamente.

Análisis de los datos

Selección de los grupos de datos

Los atributos que se utilizarán en el estudio son los indicados anteriormente: Estación, Provincia, Tipo, Distancia e Infantil. Son los parámetros fundamentales de las pruebas deportivas:

- En qué estación del año se celebran.
- Dónde se celebran. Se utilizará el dato disponible, que es la provincia. Pudiera ser interesante hacer una agrupación por comunidades autónomas. También hay que tener en cuenta que, después de la limpieza de datos, además de las provincias españolas hay una categoría adicional “resto del mundo”. Esta categoría incluye todas las pruebas anunciadas en la web utilizada en el estudio y que no se celebran en España.
- Tipo de prueba que se celebra.
- Distancia de la prueba.
- La prueba dispone de categoría infantil..

Es decir, los atributos seleccionados representan la información del cuándo (Estacion), dónde (Provincia), qué (Tipo), cuánto (Distancia), quién (Adultos o Infantil y Adultos), que son las cuestiones básicas respecto a cualquier problema.

Los estudios que se plantea realizar son los indicados anteriormente:

- Detección de las **reglas de asociación** entre mes de la prueba, provincia de la prueba, tipo de carrera, distancia y si es infantil o no.
- Detección de grupos mediante **técnicas de clustering** para los campos anteriores.
- Estudio de **regresión múltiple**.

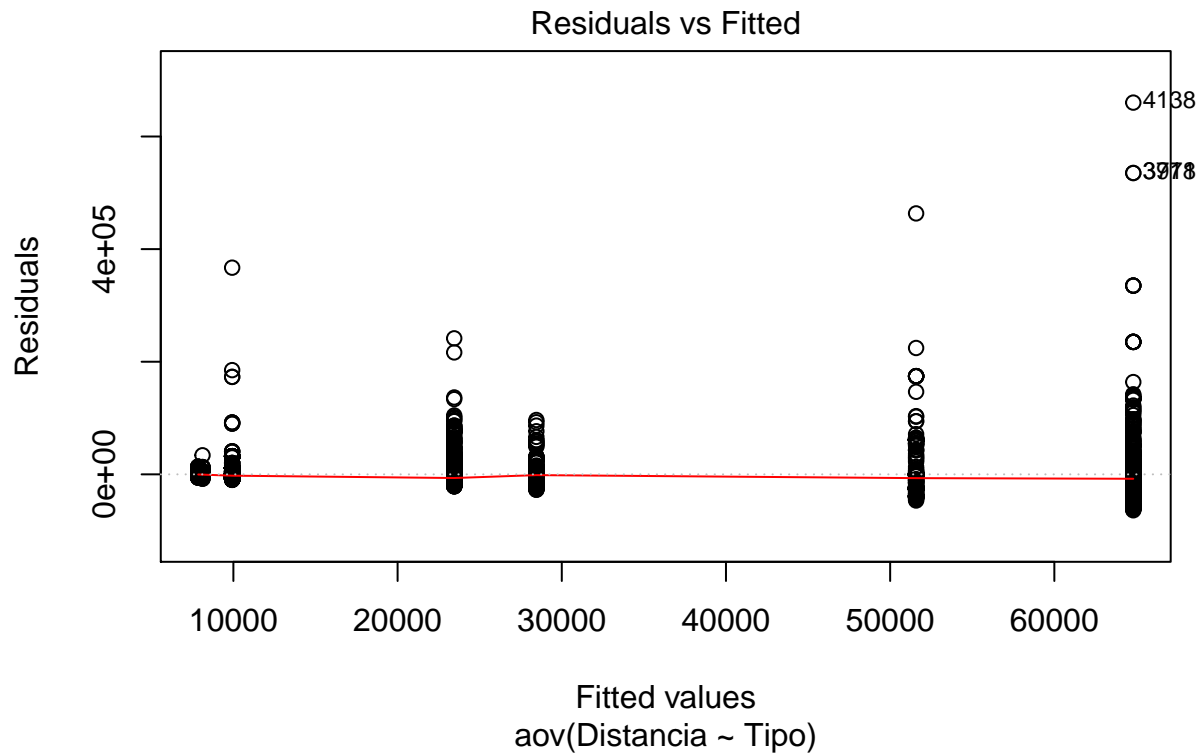
Comprobación de la normalidad y homogeneidad de la varianza

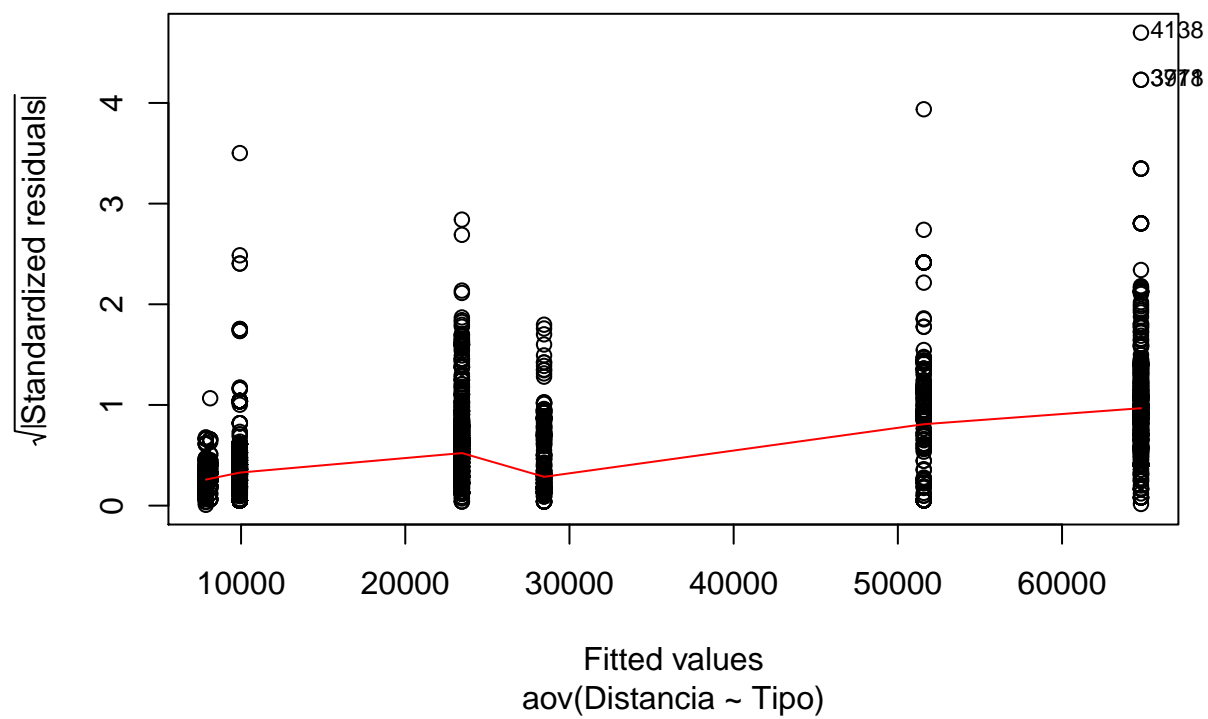
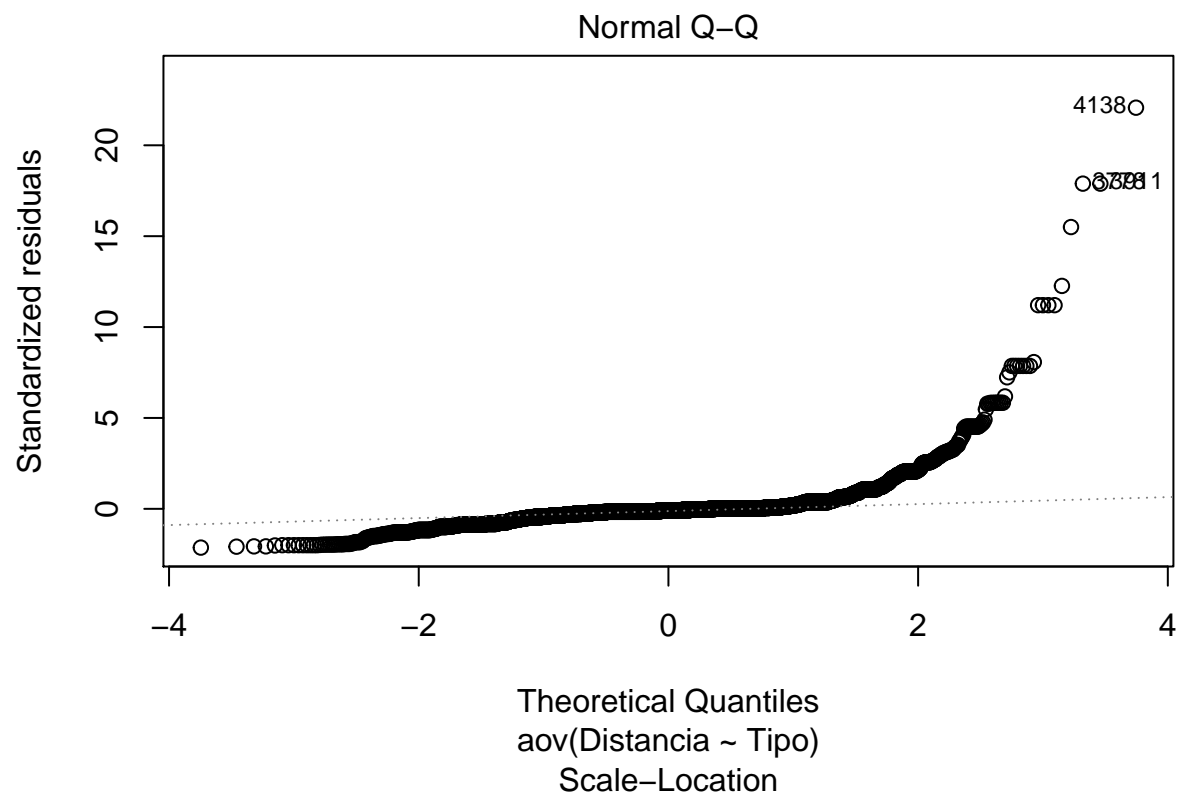
En este apartado se hace un análisis ANOVA contrastando la distancia y el tipo de prueba.

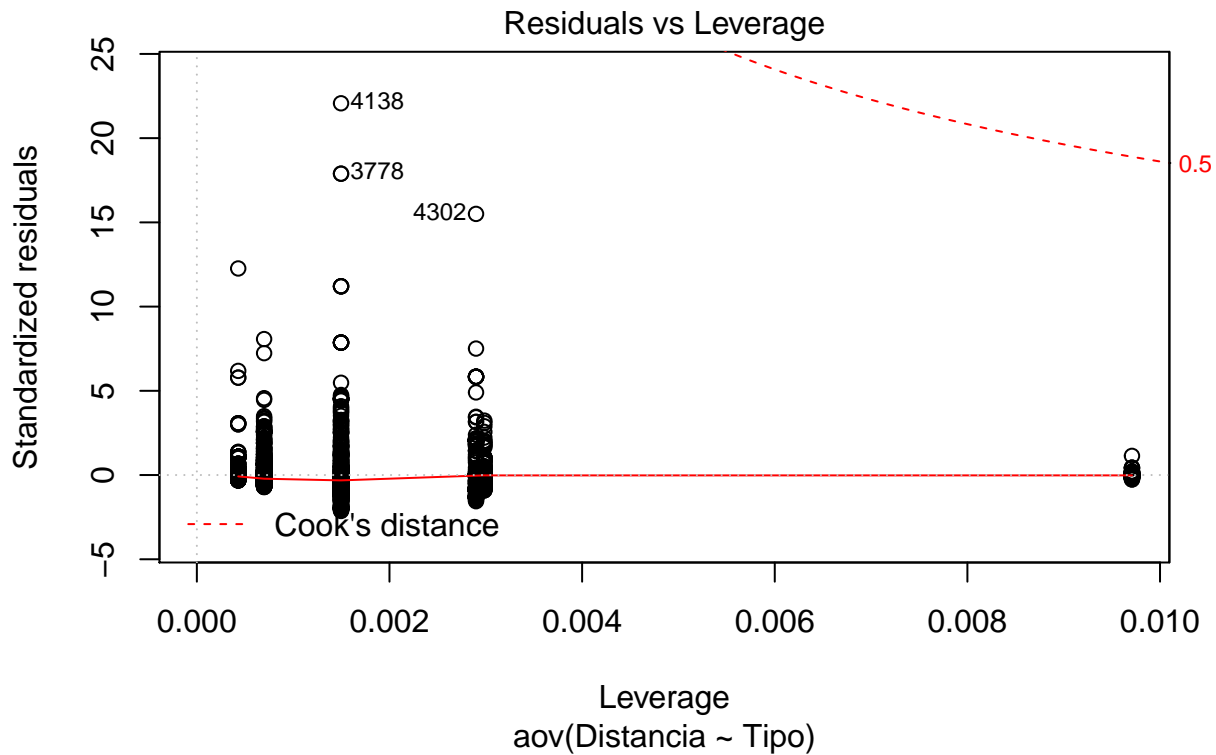
```
# Cálculo de ANOVA Distancia vs. Tipo
anova <- aov(Distancia ~ Tipo, data = datos)
summary(anova)
```

```
##              Df      Sum Sq  Mean Sq F value Pr(>F)
## Tipo          6 1.956e+12 3.261e+11   363.8 <2e-16 ***
## Residuals    5543 4.968e+12 8.962e+08
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(anova)
```







Aplicación de pruebas

Reglas de asociación

En este punto se utilizará el algoritmo *apriori* para obtener las reglas de asociación del conjunto de datos. Para ello hay que convertir la variable numérica distancia en un factor. Para ello se creará una nueva columna, llamada DistanciaF, en la que se categorizará la distancia de la siguiente forma:

- CORTA: $\leq 1\text{km}$.
- MEDIA: $> 1\text{km}$ y $\leq 5\text{km}$.
- LARGA: $> 5\text{km}$ y $\leq 20\text{km}$.
- FONDO: $> 20\text{km}$ y $\leq 50\text{km}$.
- GRAN FONDO: $> 50\text{km}$ y $\leq 100\text{km}$.
- ULTRAMARATÓN: $> 100\text{km}$.

```
datos$DistanciaF <-cut (datos[["Distancia"]], breaks=c(0, 1000, 5000, 20000, 50000, 100000, 1000000), 1)
table(datos$DistanciaF)
```

```
##
##          CORTA          MEDIA          LARGA          FONDO          GRAN FONDO
##           14           994          2397          1600             360
## ULTRAMARATON
##           185
```

En la tabla anterior se muestra el número de pruebas de cada categoría.

```
# Instalación (si es necesario) y carga de la librería arules
if(!require(arules)){
  install.packages('arules', repos='http://cran.rstudio.com')
  library(arules)
}
```

```
## Loading required package: arules
```

##

##

```
# Instalación (si es necesario) y carga de la librería dplyr
if(!require(dplyr)){
  install.packages('dplyr', repos='http://cran.rstudio.com')
  library(dplyr)
}
```

##

##

##

##

```
# Se ejecuta el modelo apriori en las columnas seleccionadas.
rules <- apriori(select(datos, Estacion, Provincia, Tipo, DistanciaF, Infantil), parameter = list(supp
```

##

##

##

##

```
## creating S4 object ... done [0.00s].
```

15

```
# Se ordenan las reglas.
rules <- sort(rules, by="confidence", decreasing=TRUE)
# Se muestran las reglas.
inspect(rules[1: 10], ruleSep = "---->", itemSep = " + ", setStart = "", setEnd = "", linebreak = FALSE)
```

```
##      lhs
## [1] Provincia=VALENCIA + DistanciaF=MEDIA          ---->
## [2] Estacion=SPRING + Tipo=CICLISMO + DistanciaF=ULTRAMARATON ---->
## [3] Estacion=SPRING + Tipo=CICLISMO + DistanciaF=GRAN FONDO ---->
## [4] Provincia=VALENCIA + DistanciaF=MEDIA + Infantil=TRUE ---->
## [5] Tipo=CICLISMO + DistanciaF=GRAN FONDO          ---->
## [6] Estacion=SPRING + DistanciaF=ULTRAMARATON      ---->
## [7] Tipo=CICLISMO + DistanciaF=ULTRAMARATON      ---->
## [8] Estacion=FALL + Tipo=CICLISMO                  ---->
## [9] Estacion=WINTER + Tipo=CICLISMO + DistanciaF=GRAN FONDO ---->
## [10] Estacion=SPRING + Tipo=CICLISMO               ---->
##      rhs      support    confidence lift    count
## [1] Tipo=RUNNING 0.01585586 1.0000000 2.384021 88
## [2] Infantil=TRUE 0.01171171 1.0000000 1.489933 65
## [3] Infantil=TRUE 0.01747748 1.0000000 1.489933 97
## [4] Tipo=RUNNING 0.01081081 1.0000000 2.384021 60
## [5] Infantil=TRUE 0.03621622 0.9950495 1.482557 201
## [6] Infantil=TRUE 0.02090090 0.9914530 1.477198 116
## [7] Infantil=TRUE 0.01639640 0.9891304 1.473738 91
## [8] Infantil=TRUE 0.01153153 0.9846154 1.467011 64
## [9] Infantil=TRUE 0.01135135 0.9843750 1.466653 63
## [10] Infantil=TRUE 0.05405405 0.9803922 1.460719 300
```

Como se ve de los datos mostrados, se han encontrado 46 reglas. Por ejemplo, la primera regla es “Si la provincia es VALENCIA y la distancia es MEDIA (entre 1 y 5 km), entonces el tipo de carrera es RUNNING”.

Árboles de decisión

El siguiente estudio a realizar también tiene que ver con la clasificación, y para ello se utilizarán árboles de decisión. Los datos ya se encuentran en el formato adecuado para ello, pues todas las variables que se utilizarán están factorizadas. En este caso se utilizarán las mismas variables del punto anterior.

Lo primero que hay que realizar es dividir el dataset en dos subconjuntos, uno de test y otro de training. El conjunto de entrenamiento tendrá 2/3 de los registros, y el de test tendrá el tercio restante. En primer lugar hay que desordenar los registros:

```
# Se desordenan los registros
set.seed(666)
# A continuación, se desordenan los datos.
datos_random <- datos[sample(nrow(datos)),]
```

El siguiente paso es crear el conjunto de entrenamiento y el de prueba. La clasificación se hará por la columna DistanciaF.

```
# Se inicializa la semilla
set.seed(23542)
# Se crean dos nuevas columnas: ProvinciaSA y TipoSA, suprimiendo los acentos para que el algoritmo C50
datos_random$ProvinciaSA <- as.factor(iconv(Provincia, "UTF-8", "ASCII//TRANSLIT"))
datos_random$TipoSA <- as.factor(iconv(Tipo, "UTF-8", "ASCII//TRANSLIT"))
# Se utiliza la columna Distancia para la clasificación.
cy <- as.factor(datos_random[,10])
```



```
# Se seleccionan las otras tres columnas.
cX <- select(datos_random, Estacion, TipoSA, ProvinciaSA, Infantil)

# Se dividen los dos conjuntos
ind = sample(1:nrow(datos_random), size=floor((2/3)*nrow(datos_random)))
entrenX<-cX[ind,]
entreny<-cy[ind]
pruebaX<-cX[-ind,]
pruebay<-cy[-ind]
```

A continuación, se genera el modelo utilizando el algoritmo C5.0:

```
# Instalación (si es necesario) y carga de la librería C50
if(!require(C50)){
  install.packages('C50', repos='http://cran.rstudio.com')
  library(C50)
}
```

```
## Loading required package: C50
```

```
modelo <- C5.0(entrenX, entreny, rules=TRUE )
summary(modelo)
```

```
##
## Call:
## C5.0.default(x = entrenX, y = entreny, rules = TRUE)
##
##
## C5.0 [Release 2.07 GPL Edition]      Mon May 27 15:55:40 2019
## -----
##
## Class specified by attribute `outcome'
##
## Read 3700 cases (5 attributes) from undefined.data
##
## Rules:
##
## Rule 1: (12/5, lift 3.1)
##   Estacion = WINTER
##   TipoSA = Ciclismo
##   ProvinciaSA = Barcelona
##   Infantil = TRUE
##   -> class MEDIA [0.571]
##
## Rule 2: (11/6, lift 2.5)
##   TipoSA in {Running, Trail}
##   ProvinciaSA = Portugal
##   -> class MEDIA [0.462]
##
## Rule 3: (1600/1210, lift 1.3)
##   Estacion = WINTER
##   -> class MEDIA [0.244]
##
## Rule 4: (7/1, lift 1.8)
##   Estacion = SPRING
```

```

## TipoSA in {Running, Trail, Triatl'on}
## ProvinciaSA = Valladolid
## Infantil = TRUE
## -> class LARGA [0.778]
##
## Rule 5: (34/9, lift 1.7)
## TipoSA in {Cross, Duatl'on, Triatl'on}
## ProvinciaSA = Barcelona
## -> class LARGA [0.722]
##
## Rule 6: (30/10, lift 1.6)
## Estacion in [SUMMER-WINTER]
## TipoSA in {Running, Trail}
## ProvinciaSA = C'adiz
## -> class LARGA [0.656]
##
## Rule 7: (53/25, lift 1.2)
## ProvinciaSA = Las Palmas
## -> class LARGA [0.527]
##
## Rule 8: (1211/591, lift 1.2)
## Infantil = FALSE
## -> class LARGA [0.512]
##
## Rule 9: (231/120, lift 1.1)
## TipoSA = Duatl'on
## -> class LARGA [0.481]
##
## Rule 10: (1964/1087, lift 1.1)
## Estacion in [FALL-WINTER]
## -> class LARGA [0.447]
##
## Rule 11: (4, lift 2.8)
## Estacion in [SUMMER-WINTER]
## TipoSA = Ciclismo
## ProvinciaSA = Alicante
## Infantil = TRUE
## -> class FONDO [0.833]
##
## Rule 12: (13/3, lift 2.5)
## Estacion in [SPRING-SUMMER]
## ProvinciaSA = Segovia
## Infantil = TRUE
## -> class FONDO [0.733]
##
## Rule 13: (16/5, lift 2.2)
## Estacion in [SPRING-FALL]
## ProvinciaSA = Burgos
## Infantil = TRUE
## -> class FONDO [0.667]
##
## Rule 14: (15/5, lift 2.2)
## TipoSA in {Cross, Obst'acul.}
## ProvinciaSA = Alicante

```

```

## Infantil = TRUE
## -> class FONDO [0.647]
##
## Rule 15: (22/9, lift 2.0)
## TipoSA in {Duatl'on, Trail}
## ProvinciaSA = Cantabria
## Infantil = TRUE
## -> class FONDO [0.583]
##
## Rule 16: (36/15, lift 1.9)
## Estacion in [SPRING-FALL]
## TipoSA = Running
## ProvinciaSA = Barcelona
## Infantil = TRUE
## -> class FONDO [0.579]
##
## Rule 17: (14/6, lift 1.9)
## Estacion = WINTER
## TipoSA = Trail
## ProvinciaSA = Valencia
## Infantil = TRUE
## -> class FONDO [0.563]
##
## Rule 18: (39/18, lift 1.8)
## TipoSA in {Ciclismo, Duatl'on, Triatl'on}
## ProvinciaSA = Valencia
## Infantil = TRUE
## -> class FONDO [0.537]
##
## Rule 19: (41/20, lift 1.7)
## Estacion in [SPRING-FALL]
## ProvinciaSA = Asturias
## Infantil = TRUE
## -> class FONDO [0.512]
##
## Rule 20: (26/13, lift 1.7)
## Estacion in [SPRING-FALL]
## TipoSA = Running
## ProvinciaSA = Illes Balears
## Infantil = TRUE
## -> class FONDO [0.500]
##
## Rule 21: (28/14, lift 1.7)
## Estacion in [SUMMER-WINTER]
## ProvinciaSA = Las Palmas
## Infantil = TRUE
## -> class FONDO [0.500]
##
## Rule 22: (35/19, lift 1.5)
## TipoSA = Running
## ProvinciaSA = Murcia
## Infantil = TRUE
## -> class FONDO [0.459]
##

```

```

## Rule 23: (760/438, lift 1.4)
## ProvinciaSA in {'Avila, Albacete, Almer'ia, Gipuzkoa, Girona, Huelva,
##               Ja'en, Madrid, Melilla, Navarra, Sevilla, Soria,
##               Tarragona, Zamora, Zaragoza}
## Infantil = TRUE
## -> class FONDO [0.424]
##
## Rule 24: (126/78, lift 1.3)
## Estacion in [FALL-WINTER]
## TipoSA = Triatl'on
## -> class FONDO [0.383]
##
## Rule 25: (2489/1563, lift 1.2)
## Infantil = TRUE
## -> class FONDO [0.372]
##
## Rule 26: (3/1, lift 9.6)
## TipoSA = Duatl'on
## ProvinciaSA = Illes Balears
## Infantil = TRUE
## -> class GRAN FONDO [0.600]
##
## Rule 27: (3/1, lift 9.6)
## Estacion = SPRING
## ProvinciaSA = Rest of the world
## Infantil = TRUE
## -> class GRAN FONDO [0.600]
##
## Rule 28: (17/9, lift 7.6)
## Estacion = SPRING
## ProvinciaSA = Teruel
## Infantil = TRUE
## -> class GRAN FONDO [0.474]
##
## Default class: LARGA
##
## Evaluation on training data (3700 cases):
##
##      Rules
##      -----
##      No      Errors
##
##      28 1961(53.0%)  <<
##
##      (a)  (b)  (c)  (d)  (e)  (f)  <-classified as
##      ----  ----  ----  ----  ----  ----
##
##              9
##              11  497  163    2
##              4  1012  544    3
##              2   395  705    3
##              2    58  160   11
##              1    29   89
##              (a): class CORTA
##              (b): class MEDIA
##              (c): class LARGA
##              (d): class FONDO
##              (e): class GRAN FONDO
##              (f): class ULTRAMARATON

```

```
##
##
## Attribute usage:
##
## 100.00% Infantil
## 57.30% Estacion
## 32.19% ProvinciaSA
## 16.24% TipoSA
##
##
## Time: 0.0 secs
```

Como se ve en los resultados mostrados, el algoritmo encuentra 28 reglas. Por ejemplo, la primera regla es “Si la estación es invierno, el tipo es Ciclismo, la provincia es Barcelona y es una prueba Infantil, entonces hay un 57% de probabilidad de que la distancia sea media”.

Regresión Múltiple

En este apartado se realizará la regresión múltiple entre los atributos Distancia, que es un atributo cualitativo y Estacion, Tipo, e Infantil que son cualitativos. La categoría de referencia para el Tipo será

```
regmod <- (lm(Distancia ~ Tipo + Estacion + Infantil, data = datos))
summary(regmod)
```

```
##
## Call:
## lm(formula = Distancia ~ Tipo + Estacion + Infantil, data = datos)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -66294  -8521  -2326   2291  663160
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   58782.3    1509.6   38.938 < 2e-16 ***
## TipoCROSS     -52965.6    2070.3  -25.584 < 2e-16 ***
## TipoDUATLÓN   -34605.7    2011.8  -17.202 < 2e-16 ***
## TipoOBSTÁCUL. -55138.1    3177.9  -17.351 < 2e-16 ***
## TipoRUNNING   -51752.7    1395.0  -37.100 < 2e-16 ***
## TipoTRAIL     -40620.7    1403.7  -28.938 < 2e-16 ***
## TipoTRIATLÓN -12412.1    2018.2   -6.150 8.29e-10 ***
## Estacion.L     -2310.9     754.5   -3.063 0.0022 **
## Estacion.Q      1779.7    1067.2    1.668 0.0954 .
## Estacion.C     -2952.7    1293.5   -2.283 0.0225 *
## InfantilTRUE    5411.4     947.3    5.712 1.17e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29770 on 5539 degrees of freedom
## Multiple R-squared:  0.2908, Adjusted R-squared:  0.2895
## F-statistic: 227.1 on 10 and 5539 DF,  p-value: < 2.2e-16
```

De los datos obtenidos se deduce que el tipo de prueba y la categoría (infantil o adulto) están fuertemente correlacionadas con la Distancia. La estación también lo está, pero en menor grado.

Por otro lado, el valor de R^2 es 0,2908. Esto quiere decir que el modelo explica el 29% de la variabilidad de

la variable Distancia.

Grabación del fichero

Se genera el fichero `runnings_clean.csv`, que incluye los procesos de limpieza y las variables generadas.

```
# Se graba el fichero runnings_clean.csv
write.csv(datos, file="../csv/runnings_clean.csv")
```

Representación Gráfica

En este apartado se mostrarán diferentes gráficos para una mejor comprensión de los datos.

Gráficos de tartas

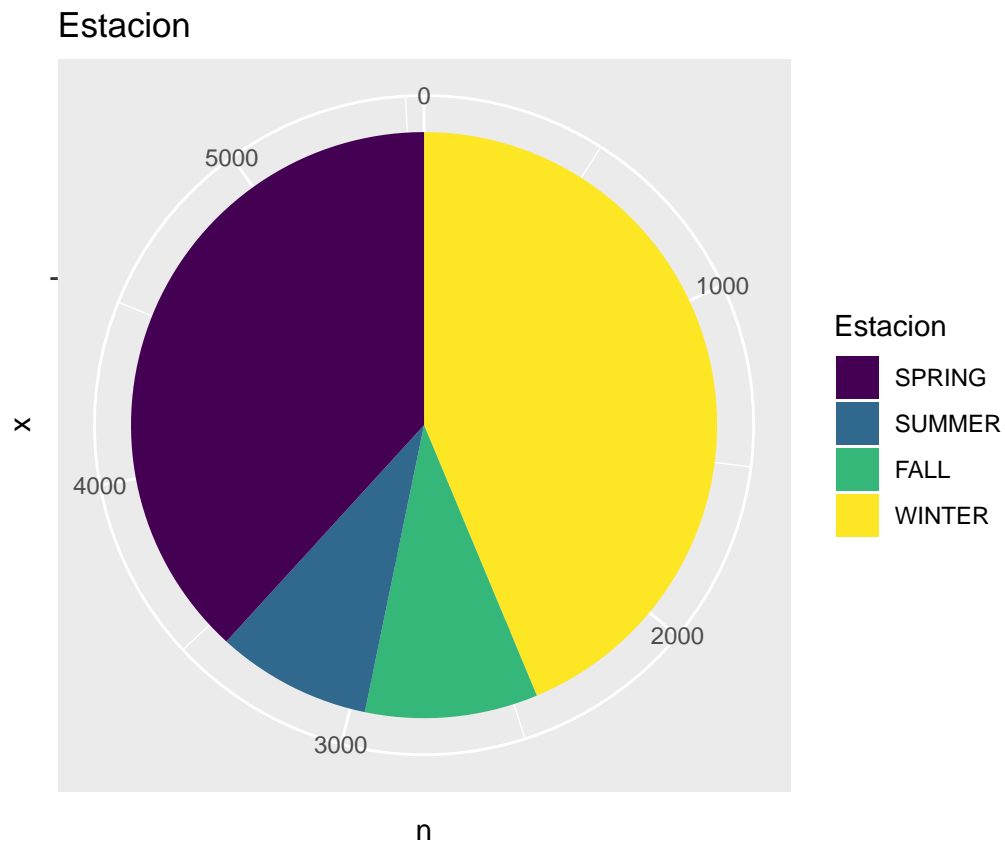
Pruebas por Estación

A continuación se muestra la tarta de pruebas por estación:

```
# Instalación (si es necesario) y carga de la librería ggplot2
if(!require(ggplot2)){
  install.packages('ggplot2', repos='http://cran.rstudio.com')
  library(ggplot2)
}
```

```
## Loading required package: ggplot2
```

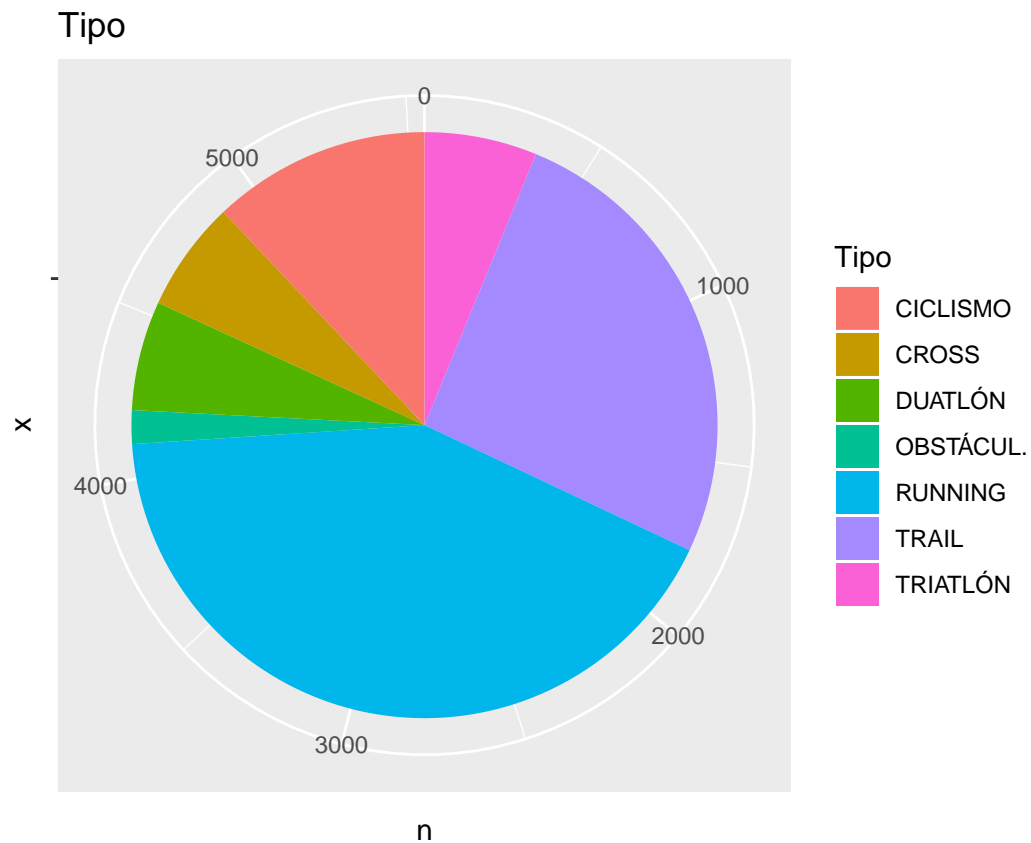
```
# Se representa la tarta de pruebas por estación.
Estacionsum <- summarize( group_by(datos, Estacion), n=length(Estacion))
ggplot(Estacionsum, aes(x="", y=n, fill=Estacion)) +
  geom_bar(width = 1, stat = "identity") +
  coord_polar("y", start=0) + ggtitle("Estacion")
```



Pruebas por Tipo

A continuación se muestra la tarta de pruebas por tipo:

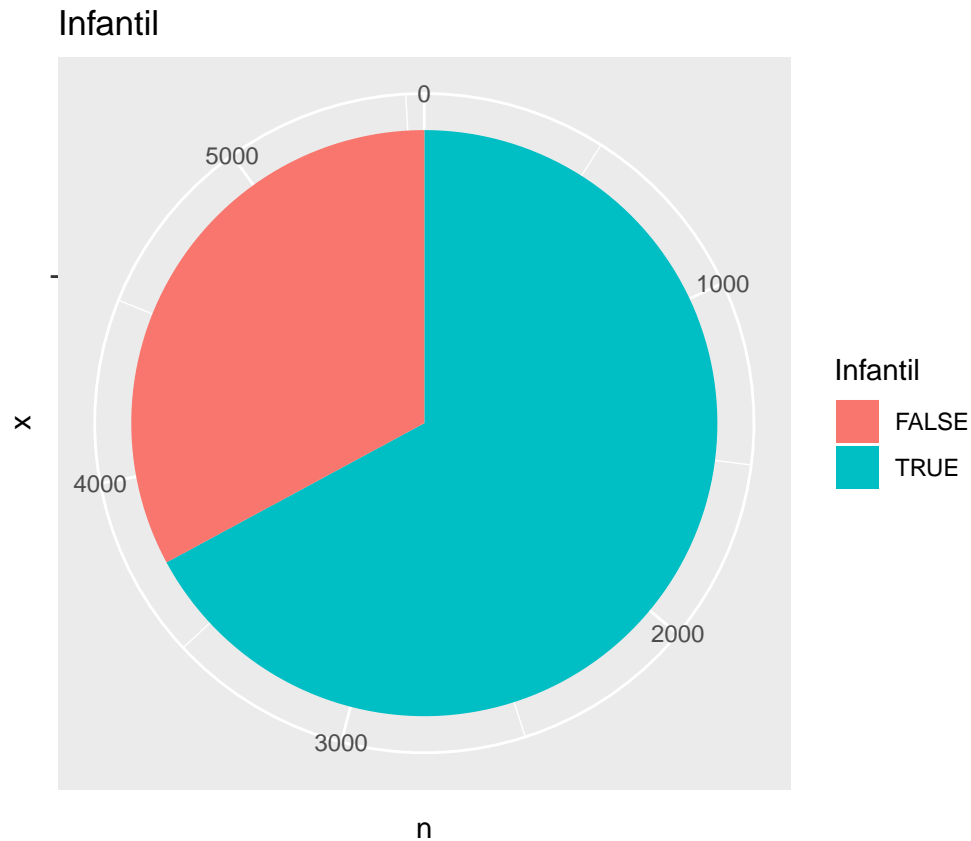
```
# Se representa la tarta de pruebas por tipo.
Tiposum <- summarize( group_by(datos, Tipo), n=length(Tipo))
ggplot(Tiposum, aes(x="", y=n, fill=Tipo)) +
  geom_bar(width = 1, stat = "identity") +
  coord_polar("y", start=0) + ggtitle("Tipo")
```



Pruebas por Categoría (Infantil o Adulto)

A continuación se muestra la tarta de pruebas por la categoría, Infantil o Adulto:

```
# Se representa la tarta de pruebas por categoría.
Infantilsum <- summarize( group_by(datos, Infantil), n=length(Infantil))
ggplot(Infantilsum, aes(x="", y=n, fill=Infantil)) +
  geom_bar(width = 1, stat = "identity") +
  coord_polar("y", start=0) + ggtitle("Infantil")
```

Diagramas de barras

Diagrama de pruebas por estación y categoría

En el diagrama siguiente se muestran las pruebas que se celebran en cada estación, diferenciando las que son de categoría infantil y las de adulto. Es fácil extraer la conclusión que los seis meses correspondientes a la primavera y el invierno son los que concentran el mayor número de pruebas.

```
# Se representa el diagrama por estación y categoría
ggplot(data=datos[1:dim(datos)[1],],aes(x=Estacion, fill=Infantil))+geom_bar()
```

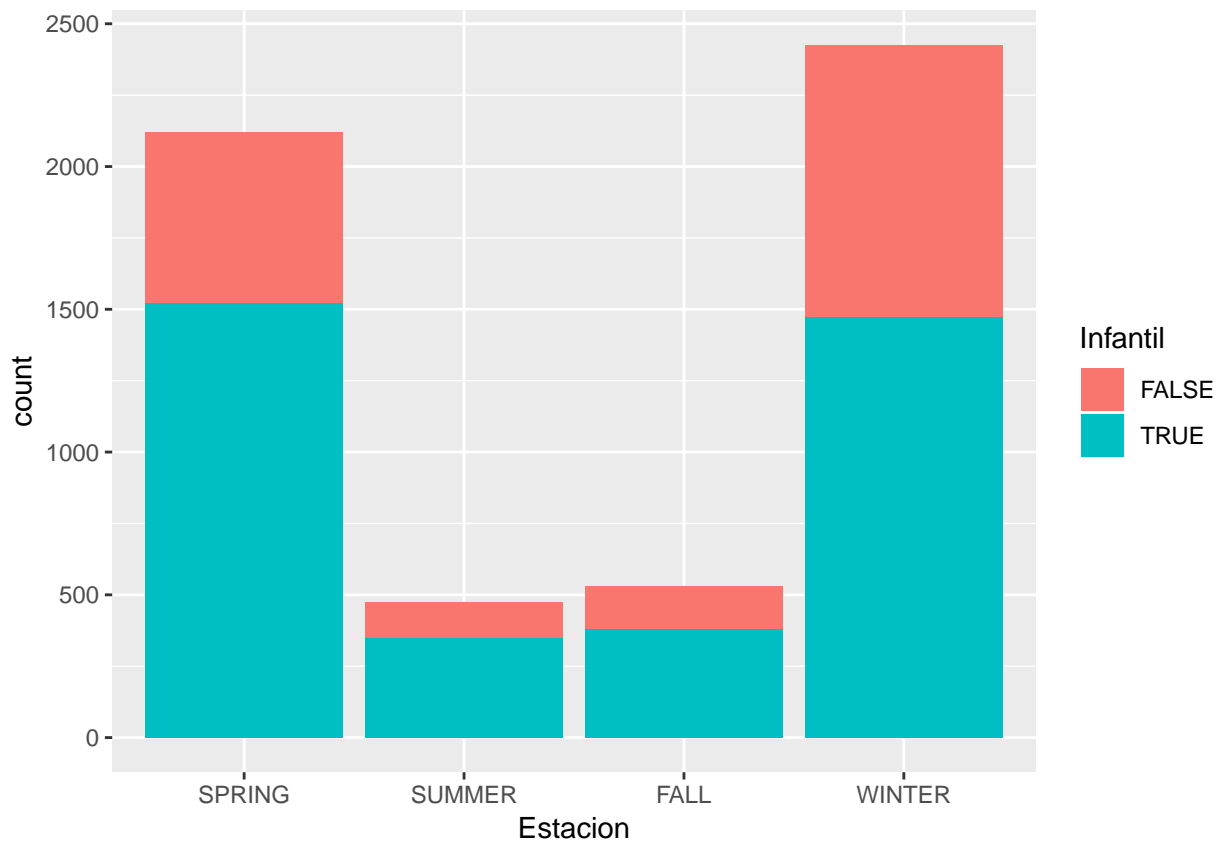
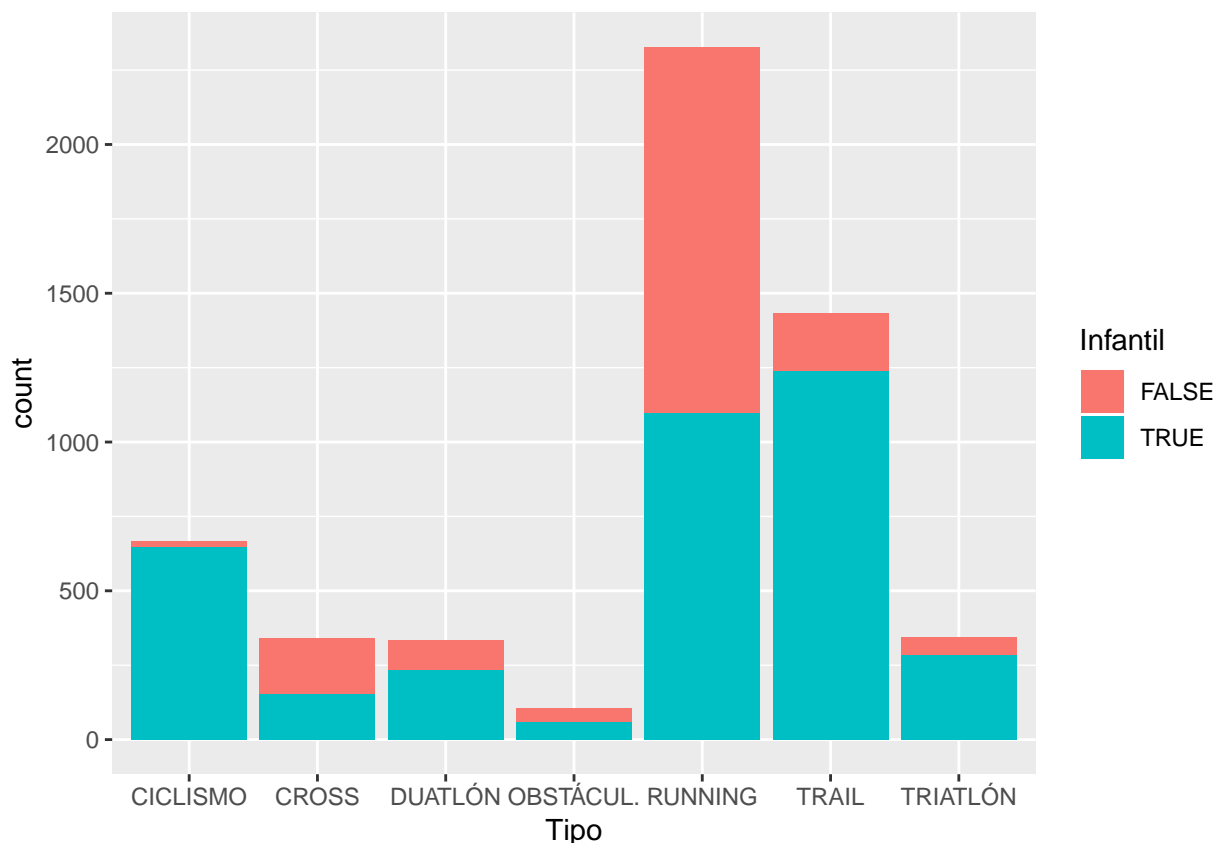


Diagrama de pruebas por tipo y categoría

En el diagrama siguiente se muestran las pruebas que se celebran de cada tipo diferenciando las que son de categoría infantil y las de adulto. En este caso, hay dos conclusiones obvias. La primera es que los dos tipos mayoritarios son running y trail. La segunda es que hay categorías que apenas se celebran en categoría infantil, como las pruebas ciclistas, trail y triatlón.

```
# Se representa el diagrama por tipo y categoría
ggplot(data=datos[1:dim(datos)[1],],aes(x=Tipo, fill=Infantil))+geom_bar()
```



Conclusiones

Del análisis realizado se extraen las siguientes conclusiones:

- La mayoría de las pruebas se concentran en primavera e invierno, mientras que en verano y otoño se celebran muy pocas. En concreto, en los seis meses correspondientes a primavera e invierno se celebran cinco veces más pruebas que en verano y otoño: el 82% de las pruebas se celebran en los seis meses de primavera e invierno.
- El dataset se ha obtenido de una página que se limita a publicitar qué pruebas se celebran. Sería interesante que el dataset incluyera el dato de número de inscritos, que permitiría obtener muchas más conclusiones.
- Otro dato interesante a incorporar al dataset sería el de la población por provincia, para poder analizar su correlación con la celebración de pruebas. Es intuitivo pensar que sí deberían estar correlacionados, pero se han detectado casos que llaman la atención. Por ejemplo, en Valencia se celebran casi las mismas pruebas que en Madrid y muchas más que en Barcelona, que tiene mayor población y un clima similar.
- El tipo de prueba mayoritaria es *running*, que se corresponde con la actividad física básica, y la segunda es *trail*, es decir, *running* campo a través. Entre ambas categorías suman casi el 70% de las pruebas.
- Las variables Tipo, Distancia, Estación e Infantil están correlacionadas.
- Las pruebas más duras (ciclismo, *trail* y triatlón) no se suelen celebrar en categoría infantil.

Contribuciones

Contribuciones	Firma
Investigación Previa	ECS - ARFR

Contribuciones	Firma
Redacción de las Respuestas	ECS - ARFR
Desarrollo Código	ECS - ARFR