Práctica 2: ¿Cómo realizar la limpieza y análisis de datos?

David Fernández Álvarez y Sara Robisco Cavite

Diciembre 2022

Contents

1. Introducción y descripción del dataset

1.1. ¿Por qué es importante y qué pregunta/problema pretende responder?

La importancia de este conjunto de datos radica en nuestra curiosidad por conocer más a fondo los datos que componen las **detecciones de ondas gravitacionales** detectadas por el consorcio **LIGO, VIRGO y KAGRA**, tanto las confirmadas como las rechazadas. La idea es aprender más de estos fenómenos gracias a sus datos.

Con estos datos queremos intentar responder algunas preguntas:

- ¿Qué intervalos de masas de objetos son los más detectados?
- ¿Hay periodos del año donde haya más probabilidad de detecciones? Si es así ¿De qué región del espacio provienen?
- ¿Qué hace que una señal se considere buena o se descarte?
- ¿Cuáles son las detecciones más cercanas? ¿Y las más lejanas?

1.2. Descripción visual y estructura

Para describir el dataset de una forma visual, cargamos las librerías `ggplot2` y `dplry`:

```
# https://cran.r-project.org/web/packages/ggplot2/index.html
if (!require('ggplot2')) install.packages('ggplot2'); library('ggplot2')
# https://cran.r-project.org/web/packages/dplyr/index.html
if (!require('dplyr')) install.packages('dplyr'); library('dplyr')
```

Ahora cargamos el fichero de datos:

```
dataset <- read.csv('../dataset/detecciones_ondas_gravitacionales.csv',
filas=dim(dataset)[1]
```

Para describir el conjunto de datos en profundidad vamos a comenzar verificando su estructura:

```
str(dataset)
```

```
## 'data.frame':    119 obs. of  36 variables:
##  $ name                          : chr  "GW200322_091133" "GW200316_215756" "GW200311_115853" "GW20
##  $ version                       : chr  "v1" "v1" "v1" "v1" ...
##  $ release                       : chr  "GWTC-3-confident" "GWTC-3-confident" "GWTC-3-confident" "GW
##  $ gps                           : num  1.27e+09 1.27e+09 1.27e+09 1.27e+09 1.27e+09 ...
##  $ mass_1                        : num  34 13.1 34.2 NA 36.4 28.3 37.8 19.3 40 38.9 ...
##  $ mass_1_upper                  : num  48 10.2 6.4 NA 11.2 17.1 8.7 5 6.9 14.1 ...
##  $ mass_1_lower                  : num  -18 -2.9 -3.8 NA -9.6 -7.7 -8.5 -3 -4.5 -8.6 ...
##  $ mass_2                        : num  14 7.8 27.7 NA 13.8 14.8 20 14 32.5 27.9 ...
##  $ mass_2_upper                  : num  16.8 1.9 4.1 NA 7.2 6.5 8.1 2.8 5 9.2 ...
##  $ mass_2_lower                  : num  -8.7 -2.9 -5.9 NA -3.3 -6.4 -5.7 -3.5 -7.2 -9 ...
##  $ network_snr                   : num  6 10.3 17.8 9.2 7.1 7.8 10.8 12.5 20 8.5 ...
##  $ network_snr_upper             : num  1.7 0.4 0.2 NA 0.5 0.4 0.3 0.3 0.2 0.3 ...
##  $ network_snr_lower             : num  -1.2 -0.7 -0.2 NA -0.5 -0.6 -0.4 -0.4 -0.2 -0.5 ...
##  $ distance                      : int  3600 1120 1170 NA 5400 2100 1480 1150 1710 4000 ...
##  $ distance_upper                : int  7000 470 280 NA 2700 1700 1020 510 490 2800 ...
##  $ distance_lower                : int  -2000 -440 -400 NA -2600 -1100 -700 -530 -640 -2200 ...
##  $ chi_eff                       : num  0.24 0.13 -0.02 NA 0.65 0.32 0.01 -0.12 0.1 -0.07 ...
##  $ chi_eff_upper                 : num  0.45 0.27 0.16 NA 0.17 0.28 0.25 0.17 0.15 0.27 ...
##  $ chi_eff_lower                 : num  -0.51 -0.1 -0.2 NA -0.21 -0.46 -0.26 -0.28 -0.15 -0.33 ...
##  $ total_mass                    : num  55 21.2 61.9 NA 50.6 43.9 57.8 33.5 72.2 67 ...
```

```
##  $ total_mass_upper               : num  37 7.2 5.3 NA 10.9 11.8 9.6 3.6 7.2 17 ...
##  $ total_mass_lower               : num  -27 -2 -4.2 NA -8.5 -7.5 -6.9 -3 -5.1 -12 ...
##  $ chirp_mass                     : num  15.5 8.75 26.6 NA 19 17.5 23.4 14.2 31.1 28.2 ...
##  $ chirp_mass_upper               : num  15.7 0.62 2.4 NA 4.8 3.5 4.7 1.5 3.2 7.3 ...
##  $ chirp_mass_lower               : num  -3.7 -0.55 -2 NA -2.8 -3 -3 -1.4 -2.6 -5.1 ...
##  $ detector_frame_chirp_mass      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ detector_frame_chirp_mass_upper: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ detector_frame_chirp_mass_lower: num  NA NA NA NA NA NA NA NA NA NA ...
##  $ redshift                       : num  0.6 0.22 0.23 NA 0.83 0.38 0.28 0.22 0.32 0.66 ...
##  $ redshift_upper                 : num  0.84 0.08 0.05 NA 0.32 0.24 0.16 0.09 0.08 0.36 ...
##  $ redshift_lower                 : num  -0.3 -0.08 -0.07 NA -0.35 -0.18 -0.12 -0.1 -0.11 -0.31 ...
##  $ false_alarm_rate               : chr  "140" "â‰¤ 1.0e-05" "â‰¤ 1.0e-05" "1.3" ...
##  $ p_astro                        : chr  "0.62" "â‰¥ 0.99" "â‰¥ 0.99" "0.19" ...
##  $ final_mass                     : chr  "53" "20.2" "59.0" "" ...
##  $ final_mass_upper               : num  38 7.4 4.8 NA 11.1 12.3 8.9 3.5 6.6 16 ...
##  $ final_mass_lower               : num  -26 -1.9 -3.9 NA -7.7 -6.9 -6.6 -2.8 -4.7 -11 ...
```

Observamos que tenemos **119 registros** corespondientes con datos de ondas gravitacionales y **36 variables** que los caracterizan. A continuación describimos las variables:

- **name**: cadena de caracteres con el identificador de la detección de la onda gravitacional.
- **version**: versión de la detección. Se revisan periódicamente.
- **release**: datos de la comunicación de la detección, si es confirmada, si es descartada…
- **gps**: fecha y hora de la detección en formato GPS.
- **mass__1**: masa del primer objeto en masas solares.
- **mass__1__upper**: valor máximo del rángo de error de la masa del primer objeto.
- **mass__1__lower**: valor mínimo del rángo de error de la masa del primer objeto.
- **mass__2**: masa del segundo objeto en masas solares.
- **mass__2__upper**: valor máximo del rángo de error de la masa del segundo objeto.
- **mass__2__lower**: valor mínimo del rángo de error de la masa del segundo objeto.
- **network__snr**: ratio señal/ruido en la red.
- **network__snr__upper**: valor máximo del rángo de error del ratio señal/ruido en la red.
- **network__snr__lower**: valor mínimo del rángo de error del ratio señal/ruido en la red.
- **distance**: distancia a la que se ha producido la colisión, en Megapársecs.
- **distance__upper**: valor máximo del rángo de error de la distancia.
- **distance__lower**: valor mínimo del rángo de error de la distancia.
- **chi__eff**: correlación de campo z de las fusiones de agujeros negros binarios.
- **chi__eff__upper**: valor máximo del rángo de error de la correlación de campo.
- **chi__eff__lower**: valor mínimo del rángo de error de la correlación de campo.
- **total__mass**: masa total de ambos cuerpos. Medida en masas solares.
- **total__mass__upper**: valor máximo del rángo de error de la masa total.
- **total__mass__lower**: valor mínimo del rángo de error de la masa total.
- **chirp__mass**: masa efectiva de un sistema binario. Medida en masas solares.

- **chirp_mass_upper**: valor máximo del rángo de error de la masa efectiva.
- **chirp_mass_lower**: valor mínimo del rángo de error de la masa efectiva.
- **detector_Frame_Chirp_Mass**: marco del detector de la masa efectiva. Medida en masas solares.
- **detector_Frame_Chirp_mass_upper**: valor máximo del rángo de error del marco del detector de la masa efectiva.
- **detector_Frame_Chirp_mass_lower**: valor mínimo del rángo de error del marco del detector de la masa efectiva.
- **redshift**: corrimiento al rojo, marca la velocidad a la que se alejan de nosotros.
- **redshift_upper**: valor máximo del rángo de error del corrimiento al rojo.
- **redshift_lower**: valor mínimo del rángo de error del corrimiento al rojo.
- **false_Alarm_Rate**: tasa de falsa alarma. La medida es años elevado a -1.
- **p_astro**: probabilidad de que el evento tenga un origen astrofísico.
- **final_mass**: masa final del objeto resultante tras la colisión. Medida en masas solares.
- **final_mass_upper**: valor máximo del rángo de error de la masa final.
- **final_mass_lower**: valor mínimo del rángo de error de la masa final.

1.3. Transformación de variables

Observamos que tenemos **seis variables** de tipo **caracter**: tres tienen el tipo adecuado, pero hay otras tres que deberían ser de tipo **numérico**: `false_alarm_rate`, `p_astro` y `final_mass`. Esto debemos corregirlo, para ello los transformaremos en numéricos:

```
dataset$false_alarm_rate <- as.numeric(dataset$false_alarm_rate)
dataset$p_astro <- as.numeric(dataset$p_astro)
dataset$final_mass <- as.numeric(dataset$final_mass)
```

También convertimos las fechas de formato GPS a fecha:

```
dataset$fecha <- as.POSIXct(dataset$gps, origin="1980-01-06", tz="UTC")
```

También tenemos un campo en el que se indica si la detección es buena o no, ese es el campo `release`. Debemos transformarlo para poder clasificar las detecciones entre confirmadas o no y así poder sacar mejores conclusiones. Vamos a meter ese valor en una nueva variable:

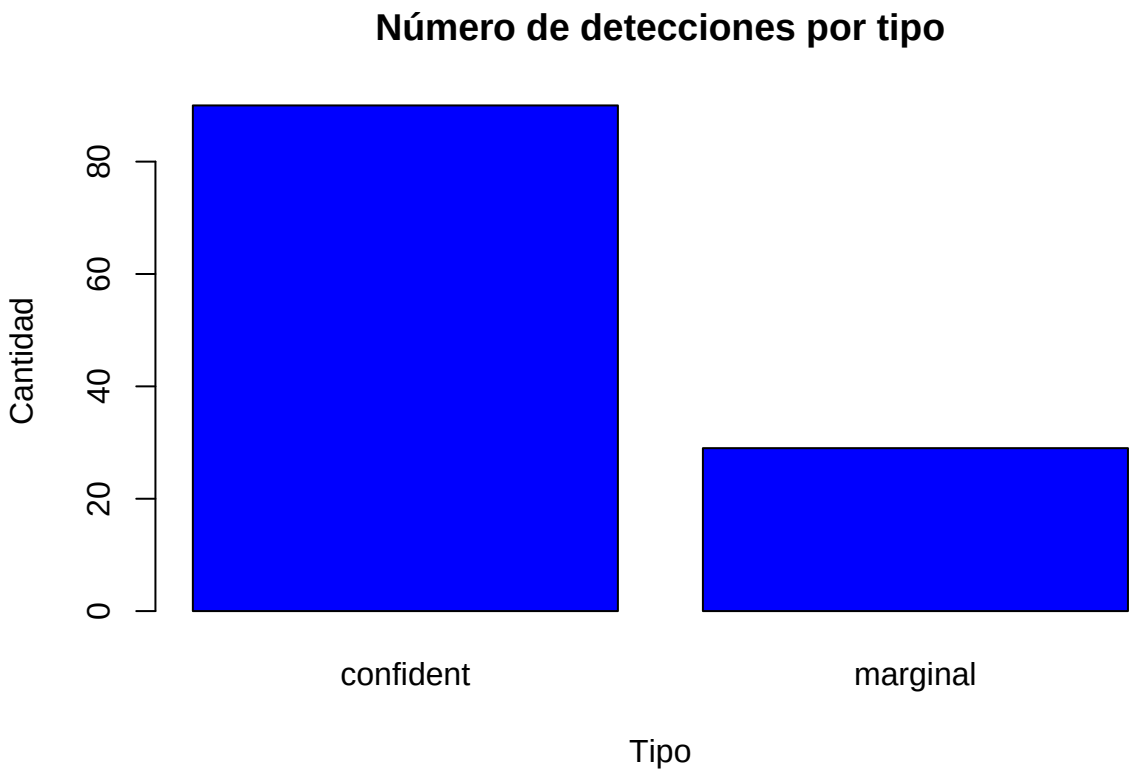
```
library(stringr)
```

```
## Warning: package 'stringr' was built under R version 4.2.2
```

```
dataset$tipo <- ifelse(str_detect(dataset$release, "confident"), "confident", "marginal")
```

Veamos ahora cuántas son detecciones de ondas gravitacionales **confirmadas** y cuántas no. Lo haremos mostrando un gráfico:

```
plot(factor(dataset$tipo),main="Número de detecciones por tipo",xlab="Tipo"
     , ylab="Cantidad",col = "blue")
```



Tenemos unas **29 detecciones de tipo marginal**, hemos tomado además como marginales aquellas que no estaban etiquetadas. Para hacer cálculos nos quedaremos como las etiquetadas como **buenas**:

```
# Eliminamos las filas de las ondas gravitacionales no confirmadas
dt_confident <- dataset[dataset$tipo!='marginal',]
```

Ahora mostramos cómo queda el análisis estadístico:

```
summary(dt_confident)
```

```
##      name              version            release              gps
##  Length:90          Length:90          Length:90          Min.   :1.126e+09
##  Class :character   Class :character   Class :character   1st Qu.:1.242e+09
##  Mode  :character   Mode  :character   Mode  :character   Median :1.251e+09
##                                                           Mean   :1.243e+09
##                                                           3rd Qu.:1.261e+09
##                                                           Max.   :1.269e+09
##
##      mass_1         mass_1_upper      mass_1_lower         mass_2
##  Min.   :  1.46   Min.   :  0.12   Min.   :-33.000   Min.   : 1.170
##  1st Qu.: 21.40   1st Qu.:  5.65   1st Qu.: -9.600   1st Qu.: 8.225
##  Median : 35.25   Median :  9.50   Median : -6.000   Median :22.750
##  Mean   : 35.05   Mean   : 12.84   Mean   : -7.676   Mean   :21.657
##  3rd Qu.: 42.15   3rd Qu.: 14.07   3rd Qu.: -3.200   3rd Qu.:29.000
##  Max.   :105.50   Max.   :104.00   Max.   : -0.100   Max.   :76.000
##
##   mass_2_upper      mass_2_lower       network_snr     network_snr_upper
##  Min.   : 0.070   Min.   :-36.500   Min.   : 6.000   Min.   :0.1000
```

```
##  1st Qu.: 2.250   1st Qu.: -9.275   1st Qu.: 9.025   1st Qu.:0.2000
##  Median : 5.350   Median : -5.850   Median :10.850   Median :0.3000
##  Mean   : 6.981   Mean   : -6.974   Mean   :12.358   Mean   :0.3365
##  3rd Qu.: 9.275   3rd Qu.: -2.425   3rd Qu.:13.350   3rd Qu.:0.4000
##  Max.   :27.100   Max.   : -0.060   Max.   :33.000   Max.   :1.7000
##                                                      NA's   :5
##  network_snr_lower    distance    distance_upper distance_lower
##  Min.   :-1.2000   Min.   :  40   Min.   :   7   Min.   :-4290.0
##  1st Qu.:-0.6000   1st Qu.: 930   1st Qu.: 360   1st Qu.:-1492.5
##  Median :-0.4000   Median :1580   Median : 755   Median : -650.0
##  Mean   :-0.4847   Mean   :2098   Mean   :1362   Mean   : -990.2
##  3rd Qu.:-0.3000   3rd Qu.:3258   3rd Qu.:1925   3rd Qu.: -380.0
##  Max.   :-0.2000   Max.   :8280   Max.   :7000   Max.   :  -15.0
##  NA's   :5
##     chi_eff          chi_eff_upper    chi_eff_lower       total_mass
##  Min.   :-0.29000   Min.   :0.0200   Min.   :-0.5100   Min.   :  3.40
##  1st Qu.:-0.03000   1st Qu.:0.1525   1st Qu.:-0.3075   1st Qu.: 31.85
##  Median : 0.05500   Median :0.2100   Median :-0.2300   Median : 58.10
##  Mean   : 0.08178   Mean   :0.2166   Mean   :-0.2282   Mean   : 58.10
##  3rd Qu.: 0.15750   3rd Qu.:0.2600   3rd Qu.:-0.1300   3rd Qu.: 74.30
##  Max.   : 0.68000   Max.   :0.5000   Max.   :-0.0100   Max.   :182.30
##                                                       NA's   :11
##  total_mass_upper total_mass_lower    chirp_mass     chirp_mass_upper
##  Min.   :  0.30   Min.   :-35.700   Min.   : 1.186   Min.   : 0.001
##  1st Qu.:  4.20   1st Qu.:-12.000   1st Qu.: 9.425   1st Qu.: 0.720
##  Median :  9.30   Median : -7.500   Median :24.400   Median : 3.350
##  Mean   : 13.34   Mean   : -8.459   Mean   :23.139   Mean   : 4.620
##  3rd Qu.: 17.55   3rd Qu.: -2.800   3rd Qu.:29.850   3rd Qu.: 7.250
##  Max.   :100.00   Max.   : -0.100   Max.   :76.000   Max.   :23.000
##  NA's   :11       NA's   :11
##  chirp_mass_lower   detector_frame_chirp_mass detector_frame_chirp_mass_upper
##  Min.   :-17.4000   Min.   : NA               Min.   : NA
##  1st Qu.: -4.9750   1st Qu.: NA               1st Qu.: NA
##  Median : -2.4500   Median : NA               Median : NA
##  Mean   : -3.3781   Mean   :NaN               Mean   :NaN
##  3rd Qu.: -0.6275   3rd Qu.: NA               3rd Qu.: NA
##  Max.   : -0.0010   Max.   : NA               Max.   : NA
##                     NA's   :90                NA's   :90
##  detector_frame_chirp_mass_lower    redshift      redshift_upper
##  Min.   : NA                     Min.   :0.0100   Min.   :0.0000
##  1st Qu.: NA                     1st Qu.:0.1900   1st Qu.:0.0625
##  Median : NA                     Median :0.2950   Median :0.1150
##  Mean   :NaN                     Mean   :0.3622   Mean   :0.1831
##  3rd Qu.: NA                     3rd Qu.:0.5475   3rd Qu.:0.2600
##  Max.   : NA                     Max.   :1.1800   Max.   :0.8400
##  NA's   :90
##  redshift_lower    false_alarm_rate       p_astro         final_mass
##  Min.   :-0.5300   Min.   :  0.00001   Min.   :0.5400   Min.   :  7.20
##  1st Qu.:-0.2200   1st Qu.:  0.00570   1st Qu.:0.8275   1st Qu.: 32.17
##  Median :-0.1150   Median :  0.18000   Median :0.9950   Median : 55.90
##  Mean   :-0.1517   Mean   :  5.12571   Mean   :0.9071   Mean   : 55.48
##  3rd Qu.:-0.0700   3rd Qu.:  2.40000   3rd Qu.:1.0000   3rd Qu.: 69.08
##  Max.   : 0.0000   Max.   :140.00000   Max.   :1.0000   Max.   :172.90
##                    NA's   :37          NA's   :18       NA's   :2
```

```
##  final_mass_upper  final_mass_lower      fecha
##  Min.   :  1.300   Min.   :-33.600   Min.   :2015-09-14 09:51:02
##  1st Qu.:  4.075   1st Qu.:-11.000   1st Qu.:2019-05-15 00:38:46
##  Median :  8.550   Median : -6.500   Median :2019-08-28 06:44:55
##  Mean   : 12.296   Mean   : -7.706   Mean   :2019-05-31 22:21:51
##  3rd Qu.: 16.000   3rd Qu.: -2.775   3rd Qu.:2019-12-21 12:49:50
##  Max.   :100.000   Max.   : -0.660   Max.   :2020-03-22 09:11:51
##  NA's   :2         NA's   :2
##      tipo
##  Length:90
##  Class :character
##  Mode  :character
##
##
##
##
```

2. Integración y selección de los datos de interés a analizar

Puede ser el resultado de adicionar diferentes datasets o una subselección útil de los datos originales, en base al objetivo que se quiera conseguir.

Esta parte la obtendremos a partir de la **limpieza de los datos**, dejando como producto final un dataset con los datos relevantes.

2.1. Limpieza de los datos

¿Los datos contienen ceros o elementos vacíos? Gestiona cada uno de estos casos.

Veamos las estadísticas de **valores nulos**:

```
# Lo hacemos así porque hemos tenido problemas con la librería missForest
colSums(is.na(dt_confident))
```

```
##                          name                       version
##                             0                             0
##                       release                           gps
##                             0                             0
##                        mass_1                  mass_1_upper
##                             0                             0
##                  mass_1_lower                        mass_2
##                             0                             0
##                  mass_2_upper                  mass_2_lower
##                             0                             0
##                    network_snr              network_snr_upper
##                             0                             5
##              network_snr_lower                      distance
##                             5                             0
##                 distance_upper                 distance_lower
##                             0                             0
##                       chi_eff                  chi_eff_upper
##                             0                             0
##                  chi_eff_lower                     total_mass
##                             0                            11
##                total_mass_upper               total_mass_lower
```

```
##                              11                              11
##                     chirp_mass               chirp_mass_upper
##                              0                               0
##               chirp_mass_lower      detector_frame_chirp_mass
##                              0                              90
## detector_frame_chirp_mass_upper detector_frame_chirp_mass_lower
##                             90                              90
##                       redshift                 redshift_upper
##                              0                               0
##                 redshift_lower               false_alarm_rate
##                              0                              37
##                        p_astro                     final_mass
##                             18                               2
##               final_mass_upper               final_mass_lower
##                              2                               2
##                          fecha                            tipo
##                              0                               0
```

En función a las dos tablas obtenidas vamos viendo qué valores no podemos usar debido a su enorme cantidad de valores vacíos.

Por ejemplo: `detector_frame_chirp_mass`, `detector_frame_chirp_mass_upper` y `detector_frame_chirp_mass_lower` tienen casi todos sus valores **nulos**. Por este motivo **descartaremos estas columnas**. Al ser datos del propio detector no son críticos y no afectarán a nuestro resultado final.

También tenemos **valores vacíos** en los campos `total_mass` y `final_mass`. Son poquitos valores por lo que los imputaremos mediante **regresión lineal**. Comenzamos por `final_mass`:

```r
# Tomamos los registros con valores NA
vacios <- which(is.na(dt_confident$final_mass))

# Generamos nuestro modelo de regresión lineal
modelo_fm <- lm(final_mass ~ mass_1+mass_2+chirp_mass, data=dt_confident)

# Evaluamos el modelo
summary(modelo_fm)
```

```
##
## Call:
## lm(formula = final_mass ~ mass_1 + mass_2 + chirp_mass, data = dt_confident)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.6133 -0.2711 -0.0426  0.2254  5.6391
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.42137    0.17599   2.394  0.01888 *
## mass_1       1.08545    0.03102  34.994  < 2e-16 ***
## mass_2       1.15748    0.08524  13.579  < 2e-16 ***
## chirp_mass  -0.39810    0.12384  -3.215  0.00185 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8019 on 84 degrees of freedom
##   (2 observations deleted due to missingness)
```

```
## Multiple R-squared:  0.9994, Adjusted R-squared:  0.9993
## F-statistic: 4.371e+04 on 3 and 84 DF,  p-value: < 2.2e-16
```

Vemos que el **coeficiente de determinación** ajustado es **0.9993** por lo que el ajuste es muy bueno. Lo aplicamos:

```
# Rellenamos los datos
dt_confident$final_mass[vacios] <- predict(modelo_fm,
                                           newdata=dt_confident[vacios,c(34,5,8,23)] )

# Redondeamos a dos decimales
dt_confident$final_mass[vacios] <- round(dt_confident$final_mass[vacios],2)
```

Hacemos lo mismo con `total_mass`:

```
# Tomamos los registros con valores NA
vacios <- which(is.na(dt_confident$total_mass))

# Generamos nuestro modelo de regresión lineal
modelo_tm <- lm(total_mass ~ chirp_mass+chi_eff+final_mass, data=dt_confident)

# Evaluamos el modelo
summary(modelo_tm)
```

```
##
## Call:
## lm(formula = total_mass ~ chirp_mass + chi_eff + final_mass,
##     data = dt_confident)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.84368 -0.10366 -0.01427  0.11641  0.69333
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.16432    0.05554  -2.958  0.00413 **
## chirp_mass   0.17225    0.01281  13.452  < 2e-16 ***
## chi_eff      1.91878    0.17372  11.045  < 2e-16 ***
## final_mass   0.97336    0.00571 170.461  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2433 on 75 degrees of freedom
##   (11 observations deleted due to missingness)
## Multiple R-squared:      1,  Adjusted R-squared:  0.9999
## F-statistic: 5.166e+05 on 3 and 75 DF,  p-value: < 2.2e-16
```

Vemos que el **coeficiente de determinación** ajustado es **0.9999** por lo que el ajuste es muy bueno. Lo aplicamos:

```
# Rellenamos los datos
dt_confident$total_mass[vacios] <- predict(modelo_tm,
                                           newdata=dt_confident[vacios,c(20,23,17,34)] )

# Redondeamos a dos decimales
dt_confident$total_mass[vacios] <- round(dt_confident$total_mass[vacios],2)
```

Procederemos a **eliminar las columnas con valores vacíos**, así como variables que no aportan nada, dejando nuestro conjunto de datos listo para trabajar con él:

```
# Eliminar columnas del dataframe
columnas_borrar <- c("version","release","gps","detector_frame_chirp_mass",
                     "detector_frame_chirp_mass_upper",
                     "detector_frame_chirp_mass_lower", "tipo",
                     "network_snr_upper","network_snr_lower",
                     "final_mass_upper","final_mass_lower","total_mass_upper",
                     "total_mass_lower","false_alarm_rate","p_astro")

ondas_g <- dt_confident[ , !(names(dt_confident) %in% columnas_borrar)]

head(ondas_g,5)
```

```
##             name mass_1 mass_1_upper mass_1_lower mass_2 mass_2_upper
## 1 GW200322_091133   34.0         48.0        -18.0   14.0         16.8
## 2 GW200316_215756   13.1         10.2         -2.9    7.8          1.9
## 3 GW200311_115853   34.2          6.4         -3.8   27.7          4.1
## 5 GW200308_173609   36.4         11.2         -9.6   13.8          7.2
## 6 GW200306_093714   28.3         17.1         -7.7   14.8          6.5
##   mass_2_lower network_snr distance distance_upper distance_lower chi_eff
## 1         -8.7         6.0     3600           7000          -2000    0.24
## 2         -2.9        10.3     1120            470           -440    0.13
## 3         -5.9        17.8     1170            280           -400   -0.02
## 5         -3.3         7.1     5400           2700          -2600    0.65
## 6         -6.4         7.8     2100           1700          -1100    0.32
##   chi_eff_upper chi_eff_lower total_mass chirp_mass chirp_mass_upper
## 1          0.45         -0.51       55.0      15.50            15.70
## 2          0.27         -0.10       21.2       8.75             0.62
## 3          0.16         -0.20       61.9      26.60             2.40
## 5          0.17         -0.21       50.6      19.00             4.80
## 6          0.28         -0.46       43.9      17.50             3.50
##   chirp_mass_lower redshift redshift_upper redshift_lower final_mass
## 1            -3.70     0.60           0.84          -0.30       53.0
## 2            -0.55     0.22           0.08          -0.08       20.2
## 3            -2.00     0.23           0.05          -0.07       59.0
## 5            -2.80     0.83           0.32          -0.35       47.4
## 6            -3.00     0.38           0.24          -0.18       41.7
##                 fecha
## 1 2020-03-22 09:11:51
## 2 2020-03-16 21:58:14
## 3 2020-03-11 11:59:11
## 5 2020-03-08 17:36:27
## 6 2020-03-06 09:37:32
```

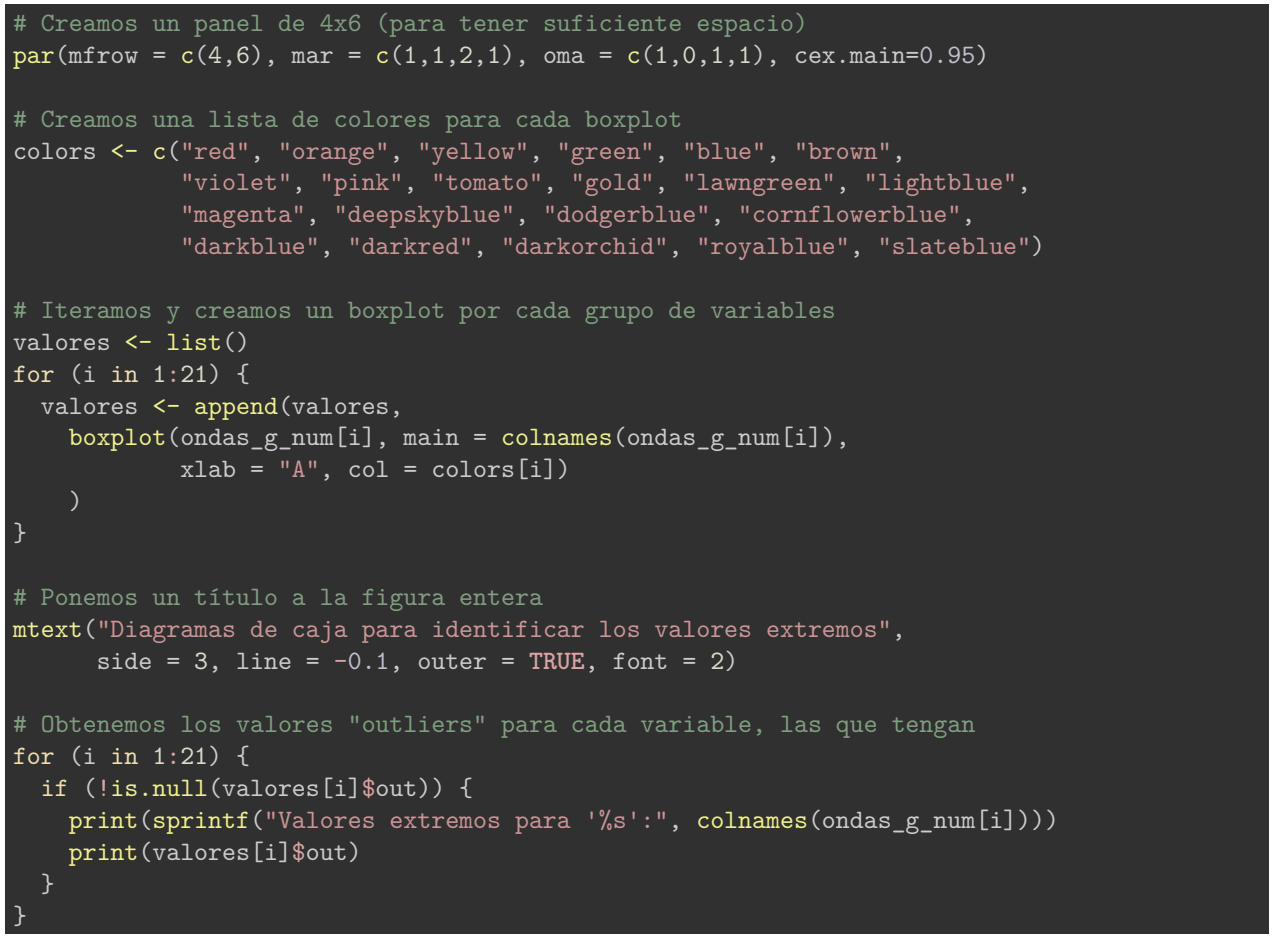
2.2. Identificación y gestión de los valores extremos

Creamos un subset "`ondas_g_num`" que contenga solamente las columnas **numéricas** de nuestro dataset para analizar sus valores extremos o *outliers*.

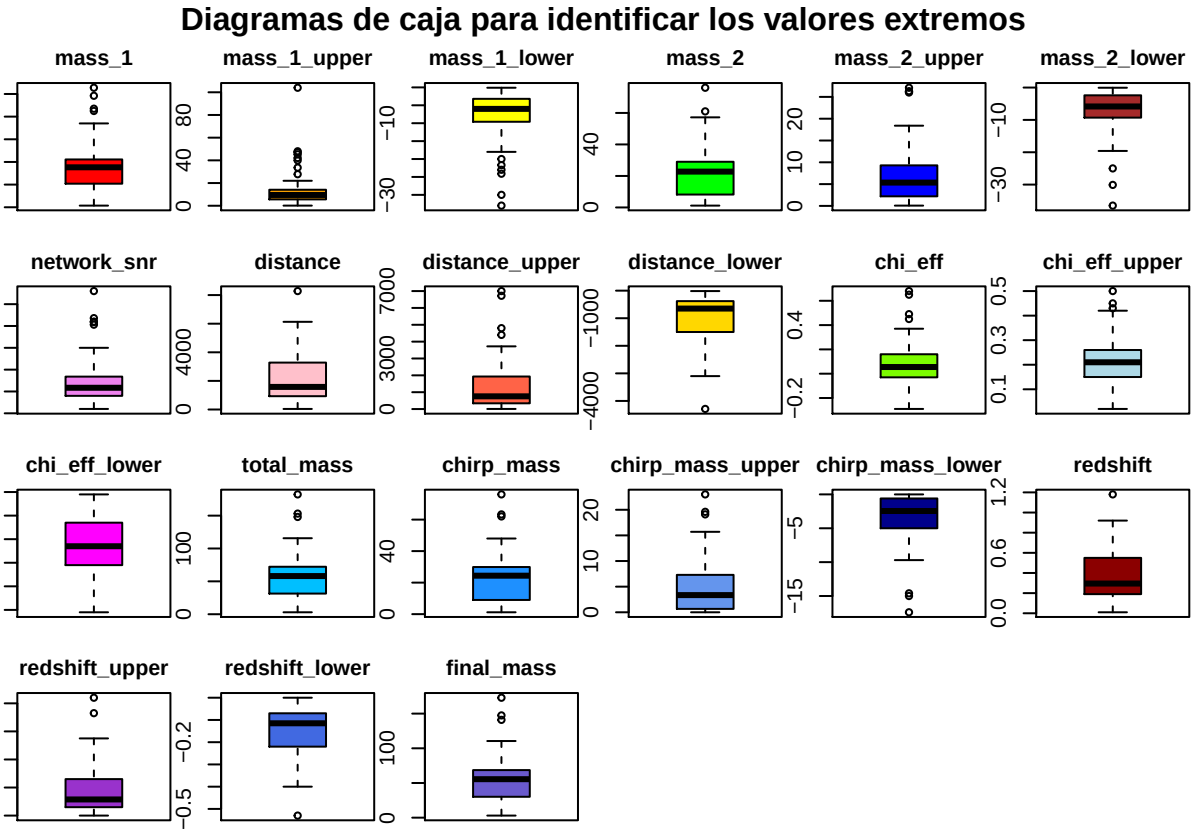
Tras esto, utilizamos la función `boxplot` para realizar un diagrama de caja por cada variable numérica, y obtenemos los valores *outliers* de aquellas que los tengan:

```
ondas_g_num <- ondas_g %>% select(-name, -fecha)
```

```r
# Creamos un panel de 4x6 (para tener suficiente espacio)
par(mfrow = c(4,6), mar = c(1,1,2,1), oma = c(1,0,1,1), cex.main=0.95)

# Creamos una lista de colores para cada boxplot
colors <- c("red", "orange", "yellow", "green", "blue", "brown",
            "violet", "pink", "tomato", "gold", "lawngreen", "lightblue",
            "magenta", "deepskyblue", "dodgerblue", "cornflowerblue",
            "darkblue", "darkred", "darkorchid", "royalblue", "slateblue")

# Iteramos y creamos un boxplot por cada grupo de variables
valores <- list()
for (i in 1:21) {
  valores <- append(valores,
    boxplot(ondas_g_num[i], main = colnames(ondas_g_num[i]),
            xlab = "A", col = colors[i])
    )
}

# Ponemos un título a la figura entera
mtext("Diagramas de caja para identificar los valores extremos",
      side = 3, line = -0.1, outer = TRUE, font = 2)

# Obtenemos los valores "outliers" para cada variable, las que tengan
for (i in 1:21) {
  if (!is.null(valores[i]$out)) {
    print(sprintf("Valores extremos para '%s':", colnames(ondas_g_num[i])))
    print(valores[i]$out)
  }
}
```



```
## [1] "Valores extremos para 'mass_2':"
## [1]  87.0  98.4 105.5  85.0
## [1] "Valores extremos para 'distance_lower':"
## [1]  48.0  40.0 104.0  47.0  42.1  33.6  45.3  27.8
## [1] "Valores extremos para 'chirp_mass_upper':"
## [1] -23.0 -30.0 -20.0 -21.7 -24.1 -33.0
```



Como podemos observar, hay **3 variables** con valores *outliers*, las cuales son:

- `mass_2`: Los valores outliers son `[87, 98.4, 105.5, 85]`
- `distance_lower`: Los valores outliers son `[48, 40, 104, 47, 42.1, 33.6, 45.3, 27.8]`
- `chirp_mass_upper`: Los valores outliers son `[-23, -30, -20, -21.7, -24.1, -33]`

Dado que no tenemos dudas ni pruebas para asegurar que la técnica de obtención de los datos ha sido incorrecta, ni consideramos que el uso de éstos afecte negativamente a nuestro análisis, tomaremos estos valores extremos como **datos válidos** y los seguiremos incluyendo en nuestro dataset.

3. Análisis de los datos

3.1. Selección de los grupos de datos que se quieren analizar/comparar

Por ejemplo, si se van a comparar grupos de datos, ¿cuáles son estos grupos y qué tipo de análisis se van a aplicar?

Hemos decidido comparar los valores: mass_1, mass_2, chirp_mass y final_mass. La finalidad es conocer si hay correlación entre la masa final tras la colisión con la masa de cada cuerpo y la masa del sistema binario.

3.2. Comprobación de la normalidad y homogeneidad de la varianza

Comenzaremos comprobando la normalidad, para ello emplearemos el test de Shapiro-Wilk. Comenzaremos por Mass_1:

```
shapiro.test(ondas_g$mass_1)
```

```
##
```

```
##  Shapiro-Wilk normality test
##
## data:  ondas_g$mass_1
## W = 0.92527, p-value = 6.667e-05
```

Observamos que el p-valor es menor que 0.05, por lo que se rechaza la hipótesis nula. Esto indica que los datos no siguen una distribución normal. Veamos qué ocurre con Mass_2:

```
shapiro.test(ondas_g$mass_2)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ondas_g$mass_2
## W = 0.93085, p-value = 0.0001316
```

De nuevo el p-valor es menor que 0.05, por lo que se rechaza la hipótesis nula. Esto indica que los datos no siguen una distribución normal. Veamos qué ocurre con chirp_mass:

```
shapiro.test(ondas_g$chirp_mass)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ondas_g$chirp_mass
## W = 0.92964, p-value = 0.0001134
```

Otra vez el p-valor es menor que 0.05, por lo que se rechaza la hipótesis nula. Por tanto, los datos no siguen una distribución normal. Veamos qué ocurre con final_mass:

```
shapiro.test(ondas_g$final_mass)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  ondas_g$final_mass
## W = 0.93133, p-value = 0.0001398
```

Aquí también el p-valor es menor que 0.05, por lo que se rechaza la hipótesis nula. Por este motivo, los datos no siguen una distribución normal. Procedemos ahora a calcular la Homocedasticidad. Al tener datos que no cumplen con la normalidad, debemos aplicar el test de Fligner-Killeen. Lo haremos por parejas con respecto al resultado, la masa final.

```
fligner.test(mass_1 ~ final_mass, data = ondas_g)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  mass_1 by final_mass
## Fligner-Killeen:med chi-squared = 82.082, df = 81, p-value = 0.4455
```

Observamos que el p-valor devuelto es muy superior a 0.05, por lo que podemos concluir que la variable mass_1 presenta varianzas estadísticamente similares para los diferentes valores de final_mass. Veamos si ocurre lo mismo con la masa del segundo objeto:

```
fligner.test(mass_2 ~ final_mass, data = ondas_g)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
```

```
## data:  mass_2 by final_mass
## Fligner-Killeen:med chi-squared = 82.031, df = 81, p-value = 0.4471
```

De manera similar, el p-valores superior a 0.05. Por lo que la masa del segundo objeto presenta varianzas estadísticamente parecidas a los diferentes valores de la masa final. Finalmente comparemos con la masa del sistema binario:

```
fligner.test(chirp_mass ~ final_mass, data = ondas_g)
```

```
##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  chirp_mass by final_mass
## Fligner-Killeen:med chi-squared = 82.125, df = 81, p-value = 0.4442
```

Como era de esperar, el p-valor es mayor a 0.05. Esto indica que la varianza de la masa del sistema binario antes de la colisión es muy similar a la varianza de la masa final tras la colisión.

Obtenemos homocedasticidad en los tres casos.

3.3. Aplicación de pruebas estadísticas para comparar los grupos de datos

En función de los datos y el objetivo del estudio, aplicar pruebas de contraste de hipótesis, correlaciones, regresiones, etc. Aplicar al menos tres métodos de análisis diferentes

Como no se cumple la normalidad y los datos son claramente dependientes, para comparar datos aplicaremos pruebas no paramétricas como Wilcoxon.

Después aplicaremos regresión lineal para aproximar la relación de dependencia lineal entre la masa final y las masas individuales de cada cuerpo, así como la masa del sistema binario.

Finalmente estudiaremos la correlación de las variables aplicando el test de correlación.

4. Representación de los resultados a partir de tablas y gráficas

Este apartado se puede responder a lo largo de la práctica, sin necesidad de concentrar todas las representaciones en este punto de la práctica. Lo dejo aquí para acordarnos de poner todas las gráficas y tablas posibles.

5. Resolución del problema

6. Vídeo