Tipología y ciclo de vida de los datos

Práctica 2: Limpieza y validación de los datos

Daniel Mato Regueira e Iago Veiras Lens

Junio de 2019

Índice

1. Detalles de la actividad

1.1. Descripción

En esta práctica se elabora un caso práctico orientado a aprender a identificar los datos relevantes para un proyecto analítico y usar las herramientas de integración, limpieza, validación y análisis de las mismas. Para hacer esta práctica tendréis que trabajar en grupos de 2 personas. Tendréis que entregar un solo archivo con el enlace Github (https://github.com) donde se encuentren las soluciones incluyendo los nombres de los componentes del equipo. Podéis utilizar la Wiki de Github para describir vuestro equipo y los diferentes archivos que corresponden a vuestra entrega. Cada miembro del equipo tendrá que contribuir con su usuario Github.

1.2. Objetivos

Los objetivos concretos de esta práctica son:

- Aprender a aplicar los conocimientos adquiridos y su capacidad de resolución de problemas en entornos nuevos o poco conocidos dentro de contextos más amplios o multidisciplinares.
- Saber identificar los datos relevantes y los tratamientos necesarios (integración, limpieza y validación) para llevar a cabo un proyecto analítico.
- Aprender a analizar los datos adecuadamente para abordar la información contenida en los datos.
- Identificar la mejor representación de los resultados para aportar conclusiones sobre el problema planteado en el proceso analítico.
- Actuar con los principios éticos y legales relacionados con la manipulación de datos en función del ámbito de aplicación.
- Desarrollar las habilidades de aprendizaje que les permitan continuar estudiando de un modo que tendrá que ser en gran medida autodirigido o autónomo.
- Desarrollar la capacidad de búsqueda, gestión y uso de información y recursos en el ámbito de la ciencia de datos

1.3. Competencias

En esta práctica se desarrollan las siguientes competencias del Máster de Data Science:

- Capacidad de analizar un problema en el nivel de abstracción adecuado a cada situación y aplicar las habilidades y conocimientos adquiridos para abordarlo y resolverlo.
- Capacidad para aplicar las técnicas específicas de tratamiento de datos (integración, transformación, limpieza y validación) para su posterior análisis.

2. Descripción del dataset

Los datos con los que se trabaja en esta práctica están recogidos en el repositorio de machine learning de la UCI. En concreto, se trata de una colección de 368 casos de cólicos en caballos repartidos en dos ficheros *csv*: *horse-colic.data* y *horse-colic.test*.

En el dataset se cuenta con, para cada caso, las siguientes variables:

- *surgery*: variable indicadora de procedimiento quirúrgico.
- *age*: edad del caballo (adulto o joven).
- *hospital_number*: número del hospital asignado.
- *rectal_temperature*: temperatura rectal expresada en grados Celsius.
- *pulse*: pulsaciones por minuto.
- *respiratory_rate*: ratio respiratorio del animal.
- *temperature_of_extremities*: temperatura de las extremidades.
- *peripheral_pulse*: pulso periférico del caballo.
- *mucous_membranes*: variable que indica el color de las mucosas.
- *capillary_refill_time*: tiempo de recarga capilar.
- *pain*: nivel de dolor del caballo.
- *peristalsis*: indicador de actividad en la tripa del caballo.
- *abdominal_distension*: distensión abdominal.
- *nasogastric_tube*: variable que se refiere a la salida de gas del tubo.
- *nasogastric_reflux*: reflujo gástrico.
- *nasogastric_reflux_PH*: ph del reflujo gástrico.
- *rectal_examination*: nivel de heces expulsadas por el animal.
- *abdomen*: calidad del abdomen del caballo.
- *packed_cell_volume*: número de células rojas en sangre.
- *total_protein*: valor total de proteínas en sangre medido en g/dL.
- *abdominocentesis_appearance*: apariencia del abdominocentesis (extraída del fluido de la cavidad abdominal del caballo).
- *abdomcentesis_total_protein*: cantidad total de proteínas en el abdominocenesis.
- *outcome*: variable que indica qué le ha pasado al caballo después del caso (vive, muere o fue sacrificado).
- *surgical_lesion*: indicador si el animal necesitó tratamiento quirúrgico.
- *type_lesion_i*: conjunto de tres variables indicadoras del tipo de lesión del caballo.
- *cp_data*: variable que indica si existen datos de patología para el caso.

Los datasets están localizables en el repositorio de esta práctica.

2.1. Importancia y objetivo del análisis

A partir de este conjunto de datos se plantea determinar qué variables tienen más afectación sobre el destino del caballo tras el cólico (*outcome*). Además, utilizaremos técnicas de regresión logística que permitan predecir dicha variable en función de otras características medidas, junto con contrastes de hipótesis que ayuden a identificar propiedades interesantes en la muestra.

Estudios como este tiene una gran importancia en la medicina veterinaria, haciendo que existan datos y conclusiones que los profesionales puedan aprovechar para mejorar sus procedimientos e implementar políticas de cuidado animal adecuadas.

3. Integración y selección de los datos

Comenzaremos la práctica cargando los conjuntos datos del repositorio. Para ello, se utilizará el comando *read.csv2()* creando así dos dataframes que anexaremos.

```
data <- read.csv2(
  file = "https://archive.ics.uci.edu/ml/machine-learning-databases/horse-colic/horse-colic.data",
  header = F, sep = "")
data <- rbind(
  data,
  read.csv(
    file = "https://archive.ics.uci.edu/ml/machine-learning-databases/horse-colic/horse-colic.test",
    header = F, sep = ""))
```

Debido a que los dataframes no poseen cabecera, creamos un vector con los nombres de las columnas y se las añadimos al dataset con la función *colnames()*:

```
colnames(data) <- c("surgery", "age", "Hospital_Number", "rectal_temperature", "pulse",
                    "respiratory_rate", "temperature_of_extremities", "peripheral_pulse",
                    "mucous_membranes", "capillary_refill_time", "pain", "peristalsis",
                    "abdominal_distension", "nasogastric_tube", "nasogastric_reflux",
                    "nasogastric_reflux_PH", "rectal_examination", "abdomen",
                    "packed_cell_volume","total_protein", "abdominocentesis_appearance",
                    "abdomcentesis_total_protein", "outcome", "surgical_lesion",
                    "type_lesion_1", "type_lesion_2", "type_lesion_3", "cp_data")
```

Realizamos a continuación una inspección breve de las características del dataframe:

```
summary(data)
```

```
##  surgery      age        Hospital_Number   rectal_temperature
##  ?:  2   Min.   :1.000   Min.   : 514279   ?      : 69
##  1:214   1st Qu.:1.000   1st Qu.: 528915   38.00  : 21
##  2:152   Median :1.000   Median : 530299   38.30  : 17
##          Mean   :1.609   Mean   :1112334   37.80  : 16
##          3rd Qu.:1.000   3rd Qu.: 534728   38.20  : 14
##          Max.   :9.000   Max.   :5305629   38.40  : 12
##                                            (Other):219
##      pulse     respiratory_rate temperature_of_extremities
##  48     : 35   ?      : 71      ?: 65
##  60     : 33   20     : 35      1: 95
##  ?      : 26   24     : 30      2: 39
##  40     : 21   12     : 27      3:135
##  44     : 16   16     : 26      4: 34
##  42     : 15   30     : 26
##  (Other):222   (Other):153
##  peripheral_pulse mucous_membranes capillary_refill_time pain
##  ?: 83            ?:48             ?: 38                 ?:63
##  1:151            1:98             1:232                 1:49
```

```
##  2:  6           2:38            2: 96               2:77
##  3:116           3:81            3:  2               3:82
##  4: 12           4:50                                4:47
##                  5:28                                5:50
##                  6:25
##  peristalsis abdominal_distension nasogastric_tube nasogastric_reflux
##  ?: 52       ?: 65                ?:131           ?:133
##  1: 49       1:101                1: 89           1:141
##  2: 22       2: 75                2:121           2: 45
##  3:154       3: 85                3: 27           3: 49
##  4: 91       4: 42
##
##
##  nasogastric_reflux_PH rectal_examination abdomen packed_cell_volume
##  ?      :299           ?:128              ?:143   ?      : 37
##  2      : 10           1: 68              1: 31   44.00  : 15
##  7.00   :  9           2: 14              2: 24   37.00  : 14
##  5.00   :  7           3: 61              3: 19   40.00  : 14
##  6.50   :  6           4: 97              4: 55   45.00  : 13
##  5.50   :  4                              5: 96   43.00  : 11
##  (Other): 33                                      (Other):264
##  total_protein abdominocentesis_appearance abdomcentesis_total_protein
##  ?      : 43   ?:194                       ?      :235
##  7.50   : 17   1: 52                       2      : 31
##  6.50   : 16   2: 62                       1      : 19
##  7.00   : 14   3: 60                       3.90   :  5
##  6.60   : 12                              2.60   :  4
##  6.00   : 11                              2.80   :  4
##  (Other):255                              (Other): 70
##  outcome surgical_lesion type_lesion_1   type_lesion_2
##  ?:  2   Min.   :1.00    Min.   :    0   Min.   :   0.00
##  1:225   1st Qu.:1.00    1st Qu.: 2112   1st Qu.:   0.00
##  2: 89   Median :1.00    Median : 3025   Median :   0.00
##  3: 52   Mean   :1.37    Mean   : 3651   Mean   :  96.97
##          3rd Qu.:2.00    3rd Qu.: 3209   3rd Qu.:   0.00
##          Max.   :2.00    Max.   :41110   Max.   :7111.00
##
##  type_lesion_3         cp_data
##  Min.   :   0.000   Min.   :1.000
##  1st Qu.:   0.000   1st Qu.:1.000
##  Median :   0.000   Median :2.000
##  Mean   :   6.003   Mean   :1.663
##  3rd Qu.:   0.000   3rd Qu.:2.000
##  Max.   :2209.000   Max.   :2.000
##
```

Explorando los datos, podemos ver cómo los valores faltantes están marcados con un '?' en el dataframe. Para simplificar cálculos, convertiremos el caracter '?' por NaN (Not a Number).

```
data[data == '?'] <- NaN
```

Luego, hacemos un estudio de qué variables nos importan para la práctica. Así llegamos a la conclusión de que el número del hospital asignado, el ratio respiratorio, el indicador de si existe patología previa del caballo

y el ph del reflujo gástrico no son demasiado relevantes para el estudio; ya sea por no aportar información o por no estar informadas en la mayoría de casos.

```
data <- data[-c(3, 6, 16, 28)]
```

Los datos categóricos están formados por el código, información que no podemos traducir de manera sencilla. Por lo tanto, el siguiente paso que se realiza es la refactorización de las variables categóricas. Este paso hará que sea más sencillo la representación e interpretación de los resultados.

```
data$surgery <- factor(data$surgery, labels = c("yes", "no"), levels = c(1, 2))
data$age <- factor(data$age, labels = c("adult", "young"), levels = c(1, 9))
data$temperature_of_extremities <- factor(data$temperature_of_extremities,
                                          labels = c("normal", "warm", "cool", "cold"),
                                          levels = c(1, 2, 3, 4))
data$peripheral_pulse <- factor(data$peripheral_pulse,
                                labels = c("normal", "increased", "reduced", "absent"),
                                levels = c(1, 2, 3, 4))
data$mucous_membranes <- factor(data$mucous_membranes,
                                labels = c("normal pink", "bright pink", "pale pink",
                                           "pale cyanotic", "bright red",
                                           "dark cyanotic"),
                                levels = c(1, 2, 3, 4, 5, 6))
data$capillary_refill_time <- factor(data$capillary_refill_time,
                                     labels = c("< 3s", "> 3s"), levels = c(1, 2))
data$pain <- factor(data$pain,
                    labels = c("alert", "depressed", "intermittent mild pain",
                               "intermittent severe pain", "continuous severe pain"),
                    levels = c(1, 2, 3, 4, 5))
data$peristalsis <- factor(data$peristalsis,
                           labels = c("hypermotile", "normal", "hypomotile", "absent"),
                           levels = c(1, 2, 3, 4))
data$abdominal_distension <- factor(data$abdominal_distension,
                                    labels = c("none", "slight", "moderate", "severe"),
                                    levels = c(1, 2, 3, 4))
data$nasogastric_tube <- factor(data$nasogastric_tube,
                                labels = c("none", "slight", "significant"),
                                levels = c(1, 2, 3))
data$nasogastric_reflux <- factor(data$nasogastric_reflux,
                                  labels = c("none", "> 1l", "< 1l"),
                                  levels = c(1, 2, 3))
data$rectal_examination <- factor(data$rectal_examination,
                                  labels = c("normal", "increased", "decreased",
                                             "absent"),
                                  levels = c(1, 2, 3, 4))
data$abdomen <- factor(data$abdomen,
                       labels = c("normal", "other", "firm feces li", "distended si",
                                  "distended li"),
                       levels = c(1, 2, 3, 4, 5))
data$abdominocentesis_appearance <- factor(data$abdominocentesis_appearance,
                                           labels = c("clear", "cloudy",
                                                      "serosanguineous"),
                                           levels = c(1, 2, 3))
data$outcome <- factor(data$outcome,
                       labels = c("lived", "died", "euthanized"), levels = c(1, 2, 3))
```

```
data$surgical_lesion <- factor(data$surgical_lesion,
                               labels = c("yes", "no"), levels = c(1, 2))
```

Además, podemos comprobar que algunas de las variables numéricas no están guardadas así. Añadimos entonces el cambio con las siguientes líneas de comando:

```
data$rectal_temperature <- as.numeric(levels(data$rectal_temperature))[
  data$rectal_temperature]
data$pulse <- as.numeric(levels(data$pulse))[data$pulse]
data$packed_cell_volume <- as.numeric(levels(data$packed_cell_volume))[
  data$packed_cell_volume]
data$total_protein <- as.numeric(levels(data$total_protein))[data$total_protein]
data$abdomcentesis_total_protein <-
  as.numeric(levels(data$abdomcentesis_total_protein))[data$abdomcentesis_total_protein]
```

También nos damos cuenta de que hay tres variables que aportan mayor información juntas. Estamos hablando del tipo de lesión. Así, la combinación de estas tres variables nos ayudan a determinar el número de lesiones del caballo.

```
data$type_lesion_1[data$type_lesion_1 > 0] <- 1
data$type_lesion_2[data$type_lesion_2 > 0] <- 1
data$type_lesion_3[data$type_lesion_3 > 0] <- 1
data$num_lesion <- data$type_lesion_1 + data$type_lesion_2 + data$type_lesion_3
data <- data[-c(22, 23, 24)]
```

Finalmente, conseguimos los datos integrados (aunque no limpios).

```
summary(data)
```

```
##  surgery       age      rectal_temperature     pulse
##  yes :214   adult:340   Min.   :35.40      Min.   : 30.00
##  no  :152   young: 28   1st Qu.:37.80      1st Qu.: 48.00
##  NA's:  2               Median :38.10      Median : 60.00
##                         Mean   :38.13      Mean   : 70.76
##                         3rd Qu.:38.50      3rd Qu.: 88.00
##                         Max.   :40.80      Max.   :184.00
##                         NA's   :69         NA's   :26
##  temperature_of_extremities  peripheral_pulse      mucous_membranes
##  normal: 95                  normal   :151     normal pink  :98
##  warm  : 39                  increased:  6     bright pink  :38
##  cool  :135                  reduced  :116     pale pink    :81
##  cold  : 34                  absent   : 12     pale cyanotic:50
##  NA's  : 65                  NA's     : 83     bright red   :28
##                                                dark cyanotic:25
##                                                NA's         :48
##  capillary_refill_time                        pain         peristalsis
##  < 3s:232              alert                    :49   hypermotile: 49
##  > 3s: 96              depressed                :77   normal     : 22
##  NA's: 40              intermittent mild pain   :82   hypomotile :154
##                        intermittent severe pain:47   absent     : 91
##                        continuous severe pain   :50   NA's       : 52
##                        NA's                     :63
```

```
##
##  abdominal_distension    nasogastric_tube nasogastric_reflux
##  none    :101         none       : 89     none:141
##  slight  : 75         slight     :121     > 1l: 45
##  moderate: 85         significant: 27     < 1l: 49
##  severe  : 42         NA's       :131     NA's:133
##  NA's    : 65
##
##
##  rectal_examination          abdomen    packed_cell_volume total_protein
##  normal   : 68      normal       : 31   Min.   : 4.00      Min.   : 3.30
##  increased: 14      other        : 24   1st Qu.:37.25      1st Qu.: 6.50
##  decreased: 61      firm feces li: 19   Median :44.00      Median : 7.50
##  absent   : 97      distended si : 55   Mean   :45.66      Mean   :24.77
##  NA's     :128      distended li : 96   3rd Qu.:52.00      3rd Qu.:58.00
##                     NA's         :143   Max.   :75.00      Max.   :89.00
##                                         NA's   :37         NA's   :43
##   abdominocentesis_appearance abdomcentesis_total_protein       outcome
##  clear          : 52          Min.   : 0.100              lived     :225
##  cloudy         : 62          1st Qu.: 2.000              died      : 89
##  serosanguineous: 60          Median : 2.100              euthanized: 52
##  NA's           :194          Mean   : 2.948              NA's      :  2
##                               3rd Qu.: 3.900
##                               Max.   :10.100
##                               NA's   :235
##  surgical_lesion   num_lesion
##  yes:232         Min.   :0.0000
##  no :136         1st Qu.:1.0000
##                  Median :1.0000
##                  Mean   :0.8478
##                  3rd Qu.:1.0000
##                  Max.   :3.0000
##
```

4. Limpieza de los datos

En este apartado trataremos con los datos vacíos en el dataset original y realizaremos un análisis de los valores extremos de sus variables continuas.

4.1. Elementos vacíos

En primer lugar, trataremos los registros que tienen pocas variables informadas. Realizamos este paso para evitar generar después registros "casi artificiales", reservándonos la imputación de valores faltantes para aquellos registros que tengan cierto grado de carencia (no superior al 50 %). Para ello, eliminaremos las filas del dataset que cumplan dicha condición.

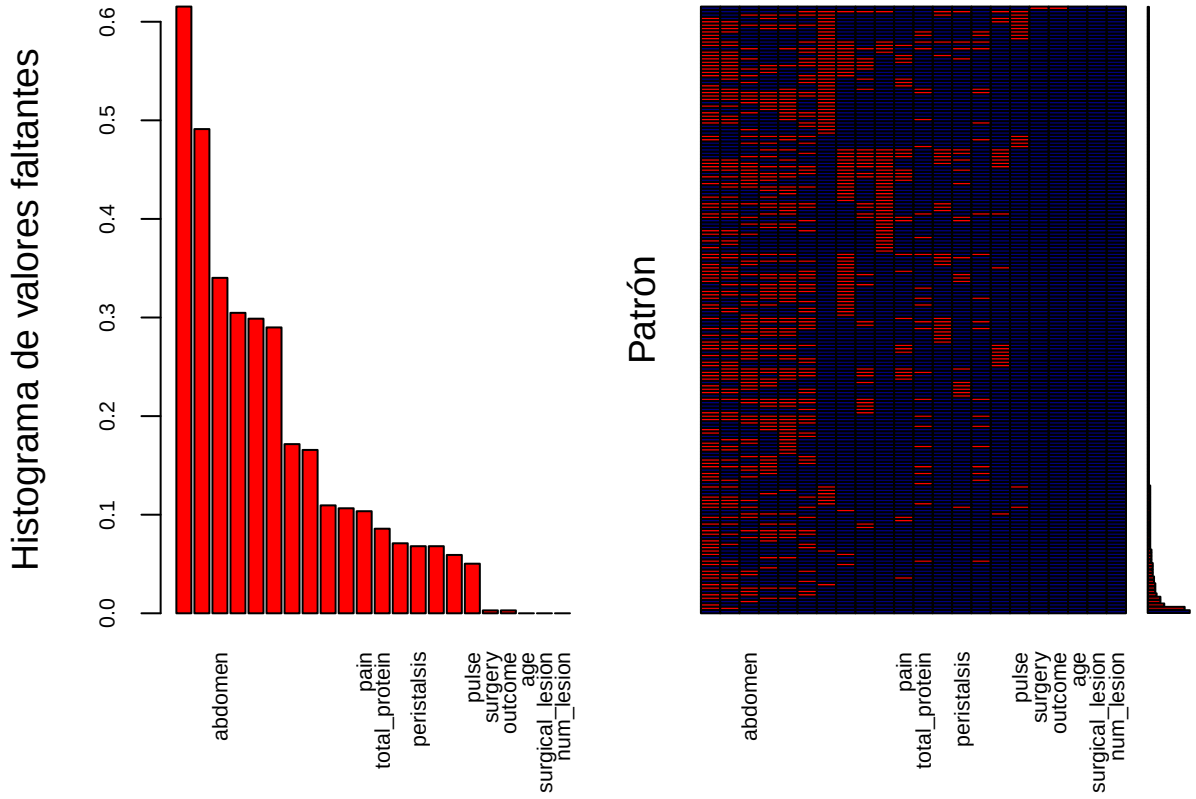
```
filas_nan <- apply(data, 1, function(y) sum(is.na(y))) < dim(data)[2]/2
data <- data[filas_nan,]
```

Para tratar con el resto de valores faltantes, cargaremos en primer lugar la librería "VIM" para estudiar los diferentes patrones de datos faltantes en los registros de nuestro conjunto de datos.

```
if (!require("VIM")) install.packages("VIM")
library(VIM)
```

Apoyándonos en la función "aggr", representaremos los patrones de datos faltantes junto con el porcentaje da registros faltantes por variable.

```
aggr_plot <- aggr(data, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE,
                  labels=names(data), cex.axis=.7,
                  gap=3, ylab=c("Histograma de valores faltantes", "Patrón"))
```




```
##
##  Variables sorted by number of missings:
##                       Variable      Count
##  abdomcentesis_total_protein 0.61538462
##  abdominocentesis_appearance 0.49112426
##                      abdomen 0.34023669
##           nasogastric_reflux 0.30473373
##             nasogastric_tube 0.29881657
##           rectal_examination 0.28994083
##           rectal_temperature 0.17159763
##             peripheral_pulse 0.16568047
##         abdominal_distension 0.10946746
##   temperature_of_extremities 0.10650888
##                         pain 0.10355030
##                total_protein 0.08579882
##             mucous_membranes 0.07100592
##                   peristalsis 0.06804734
##           packed_cell_volume 0.06804734
##        capillary_refill_time 0.05917160
##                        pulse 0.05029586
##                      surgery 0.00295858
##                      outcome 0.00295858
##                         age 0.00000000
##              surgical_lesion 0.00000000
##                   num_lesion 0.00000000
```

Según la tabla anterior, vemos como tanto la variable "abdomcentesis_total_protein" como "abodminocentesis_appearance" tienen, aproximadamente, más de un 50 % de registros faltantes, por lo que podemos ignorarlas a efectos del estudio que estamos llevando a cabo.

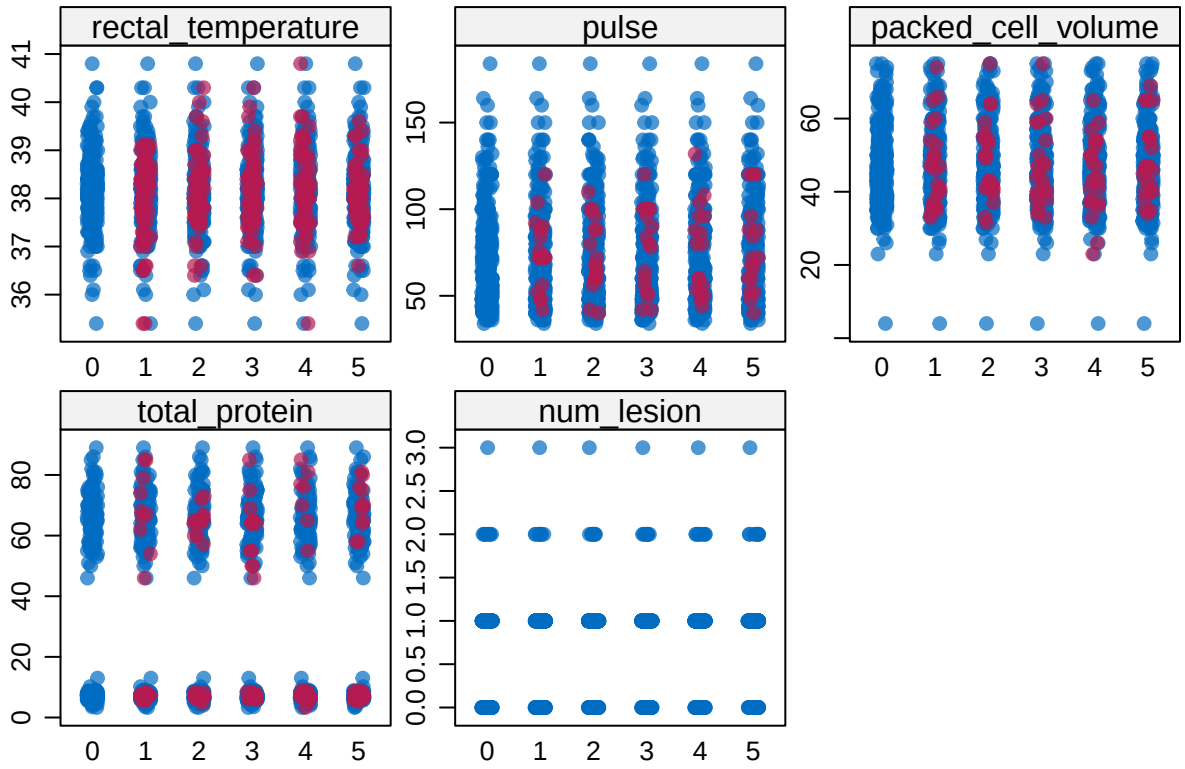
```
data <- data[-c(18, 19)]
```

Para rellenar los datos faltantes, haremos uso de la librería "MICE".

```
if (!require("mice")) install.packages("mice")
library(mice)
```

Usando la función de mismo nombre "mice", generaremos cinco capas de imputación y las representaremos para las variables numéricas en un gráfico, de tal manera que podamos comprobar que los patrones de imputación no difieren demasiado respecto de la distribución de datos originales.

```
temp_data <- mice(data, printFlag = F)
stripplot(temp_data, pch=20, cex = 1.2)
```



Veamos ahora qué métodos de imputación está utilizando la función "mice" para las variables de nuestro dataset. Para las variables numéricas se utiliza el método "Predictive mean matching", para las variables categóricas con dos clases se utiliza "Logistic regression" y para variables categóricas con más de dos clases sin orden se utiliza "Polytomous logistic regression".

```
temp_data$method
```

```
##                    surgery                        age
##                   "logreg"                         ""
##         rectal_temperature                      pulse
##                      "pmm"                      "pmm"
## temperature_of_extremities          peripheral_pulse
##                   "polyreg"                  "polyreg"
##           mucous_membranes      capillary_refill_time
##                   "polyreg"                   "logreg"
##                       pain                peristalsis
##                   "polyreg"                  "polyreg"
##       abdominal_distension           nasogastric_tube
##                   "polyreg"                  "polyreg"
##         nasogastric_reflux         rectal_examination
##                   "polyreg"                  "polyreg"
##                    abdomen         packed_cell_volume
##                   "polyreg"                      "pmm"
##              total_protein                    outcome
##                      "pmm"                  "polyreg"
##            surgical_lesion                 num_lesion
##                         ""                         ""
```

Por último, imputamos los datos faltantes utilizando la función "complete".

```
data <- complete(temp_data)
```

4.2. Valores extremos

En este apartado estudiaremos los valores extremos de todas las variables numéricas. Comenzaremos con "recta_temperature". Tal y como vemos a continuación, todos los valores de temperatura son plausibles, ya que oscilan entre 35.4ºC y 40.8ºC.

```
boxplot.stats(data$rectal_temperature)$out
```

```
##  [1] 39.9 35.4 36.5 40.3 39.7 36.4 40.3 35.4 36.5 36.5 35.4 36.1 40.8 40.0
## [15] 36.5 39.7 36.0
```

Estudiaremos ahora los outliers de "pulse". En este caso sí que nos encontramos con valores anómalos, ya que pulsos de más de 150 pulsaciones por minuto no son anatómicamente alcanzables por los caballos. Es por esto por lo que eliminaremos los registros que alcanzan dichos valores.

```
boxplot.stats(data$pulse)$out
```

```
## [1] 164 160 184 150 150
```

```
out_pulse <- boxplot.stats(data$pulse)$out
data <- data[-which(data$pulse %in% out_pulse), ]
```

Veamos ahora el estado de los valores extremos de la variable "packed_cell_volume". En este caso tenemos valores muy elevados y muy pequeños que tampoco son posibles, por lo que al igual que en el caso anterior, eliminamos los registros pertinentes.

```
boxplot.stats(data$packed_cell_volume)$out
```

```
## [1] 75 75 74  4 74
```

```
out_cells <- boxplot.stats(data$packed_cell_volume)$out
data <- data[-which(data$packed_cell_volume %in% out_cells), ]
```

En el caso de la variable "total_protein", no encontramos ningún valor extremo, por lo que no tendremos que realizar ninguna corrección sobre el conjunto de datos.

```
boxplot.stats(data$total_protein)$out
```

```
## numeric(0)
```

Por último, al tratarse "num_lesion" de una variable calculada en un apartado anterior, no tiene sentido estudiar sus valores extremo, ya que todos ellos son completamente plausibles. Dejamos esta variable intacta.

```
boxplot.stats(data$num_lesion)$out
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 2 0 2 2 0 0 0 0 0 0 0 0 0 0 0
## [36] 0 0 2 0 0 0 0 0 0 2 0 0 3 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0 0 0 0 0 2
```

Veamos entonces el tamaño del dataset completamente limpio y listo para ser analizado. Nos quedamos entonces con 328 registros con 20 variables diferentes.

```
dim(data)
```

```
## [1] 328  20
```

Para terminar este apartado, generaremos un fichero csv con el conjunto de datos filtrados y tratados.

```
write.csv(data, file = "clean_data_horse_colic.csv", row.names = F)
```

5. Análisis de los datos

En este apartado realizaremos varios análisis sobre el conjunto de datos obtenido en el apartado anterior.

5.1. Selección de grupos a analizar

Uno de los análisis que haremos en este apartado será la comprobación de si los caballos que han sido operados presentan un valor de "total_protein" medio similar a los que no han sido operados. Generaremos estos dos conjuntos de datos y los guardaremos para el estudio posterior.

```
data_oper <- data[data$surgery == "yes", ]$total_protein
data_noper <- data[data$surgery == "no", ]$total_protein
```

5.2. Comprobación de normalidad y homocedasticidad

Comprobaremos, en primer lugar, la normalidad de las cuatro variables numéricas que podrían presentarla ("rectal_temperature", "pulse", "packed_cell_volume" y "total_protein") mediante el test de normalidad de Shapiro-Wilk. En este caso, ninguna de las cuatro variables presenta una normalidad evidente en su distribución.

```
shapiro.test(data$rectal_temperature)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data$rectal_temperature
## W = 0.97219, p-value = 5.9e-06
```

```
shapiro.test(data$pulse)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data$pulse
## W = 0.91849, p-value = 2.323e-12
```

```
shapiro.test(data$packed_cell_volume)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data$packed_cell_volume
## W = 0.95995, p-value = 7.995e-08
```

```
shapiro.test(data$total_protein)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  data$total_protein
## W = 0.67076, p-value < 2.2e-16
```

Para realizar la comprobación de la homogeneidad de las varianzas de los dos grupos creados en el apartado anterior, cargaremos la librería "car".

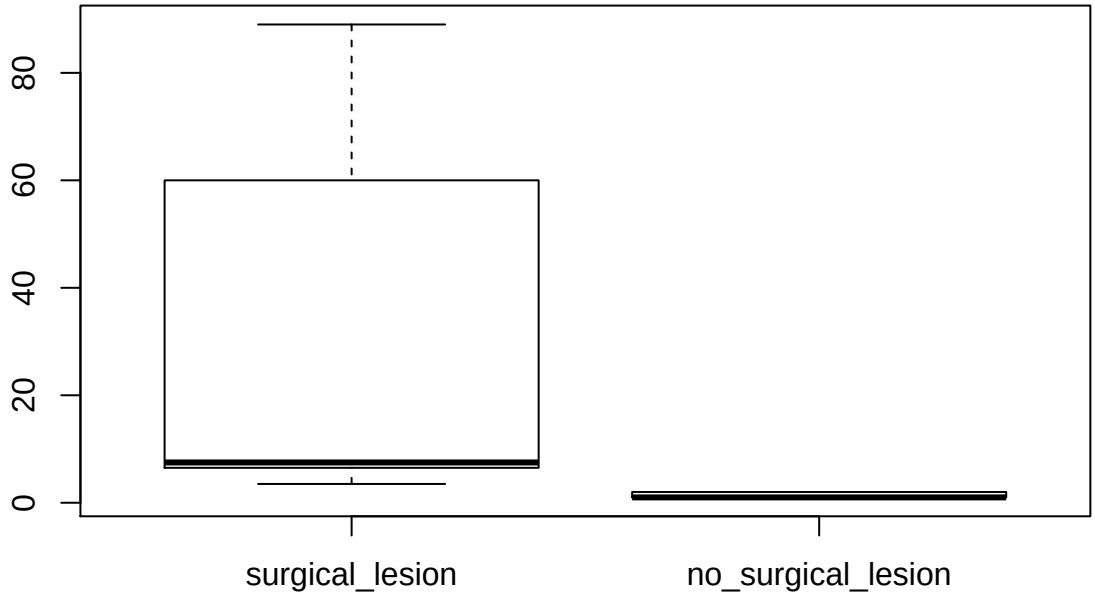
```
if (!require("car")) install.packages("car")
library(car)
```

Al realizar el test de Levene sobre la variable "total_protein" según si el caballo ha sido o no operado, comprobamos como no tienen varianza semejantes, ya que se rechaza la hipótesis nula. Si realizamos una gráfica de cajas sobre dichas variables, podemos comprobar fácilmente como se rechaza dicha hipótesis de homocedasticidad.

```
leveneTest(total_protein ~ surgical_lesion, data)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value Pr(>F)
## group   1  1.4509 0.2293
##       326
```

```
boxplot(data$total_protein, data$surgical_lesion,
        names = c("surgical_lesion", "no_surgical_lesion"))
```



5.3. Aplicación de pruebas estadísticas

Realizaremos a continuación un estudio de correlación, un contraste de hipótesis y un modelo de regresión logística.

5.3.1. Estudio de correlación

Para el estudio de correlación, necesitaremos convertir a formato numérico la variable categórica "abdominal_distension". Para ello, es necesario cargar la librería "dummies".

```
if (!require("dummies")) install.packages("dummies")
library(dummies)
```

Ayudándonos de dicha librería, extendemos la variable de interés según sus cuatro valores.

```
data <- dummy.data.frame(data, names = "abdominal_distension",
                         dummy.classes="ALL", sep = "_")
```

Para estudiar la correlación entre las distintas variables numéricas y el "outcome" del caballo, primero deberemos unificar dos de los posibles resultados de dicha variable. Para este estudio, entenderemos que los caballos que han tenido que ser sacrificados ("euthanized") equivalen a los caballos que han muerto ("died"). A continuación, convertiremos la variable categórica en una numérica.

```
levels(data$outcome) <- c(0, 1, 1)
data$outcome <- as.numeric(data$outcome) - 1
```

De entre todos las variable disponibles, seleccionaremos ahora únicamente aquellas que son numéricas.

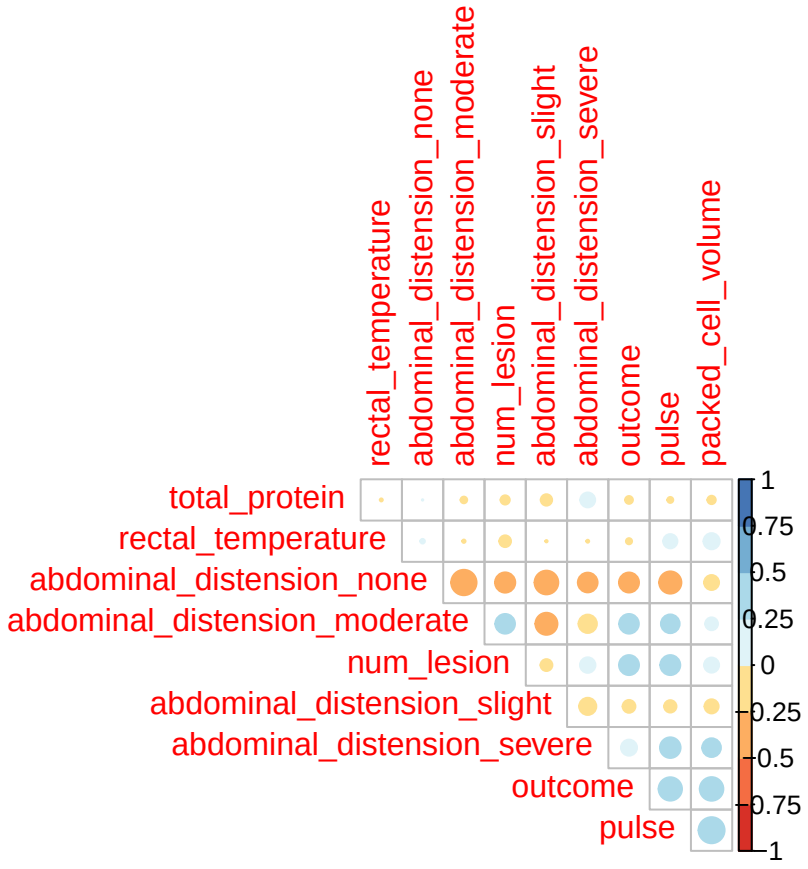
```
nums <- unlist(lapply(data, is.numeric))
data_nums <- data[ , nums]
```

Para realizar el estudio de correlación de una manera visual y fácil de entender, cargaremos las librerías "corrplot" y "RColorBrewer".

```
if (!require("corrplot")) install.packages("corrplot")
library(corrplot)
if (!require("RColorBrewer")) install.packages("RColorBrewer")
library(RColorBrewer)
```

Representamos la correlación entre las diferentes variables continuas, haciendo especial hincapié en los resultados relacionados con la variable "outcome". Podemos ver como las variables con mayor correlación con esta última son "packed_cell_volume", "pulse", "abdominal_distension_moderate" y "num_lesion". Estas variables con mayor correlación serán las que utilizaremos para estimar el modelo de regresión logística del último apartado.

```
corrplot(cor(data_nums), type = "upper", order = "hclust",
         col = brewer.pal(n = 8, name = "RdYlBu"), diag = F)
```



5.3.2. Prueba de contraste de hipótesis

Con los dos conjuntos de caballos creados al comienzo de este apartado, realizaremos un contraste de hipótesis sobre las medias de los valores del total de proteínas entre los operados y los no operados. Podemos enunciar el contraste entonces del siguiente modo:

$$H_0 : \mu_O = \mu_{NO}$$
$$H_1 : \mu_O \neq \mu_{NO}$$

Realizaremos entonces un test de Welch sobre las dos muestras para comprobar la hipótesis anterior. En esta caso, dado el p-valor de 0.07, no podemos rechazar la hipótesis nula, por lo que no podemos afirmar que los dos conjuntos tienen una diferencia significativa en sus medias.

```
t.test(data_noper, data_oper)
```

```
##
##  Welch Two Sample t-test
##
## data:  data_noper and data_oper
## t = -1.8186, df = 307.07, p-value = 0.06994
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -11.7777393   0.4638019
## sample estimates:
## mean of x mean of y
##  22.52628  28.18325
```

5.3.3. Modelo de regresión logística

Para estimar el modelo de regresión logística, dividiremos de manera aleatoria el conjunto original en dos muestra, una para el entrenamiento y otra para comprobar la precisión del mismo (80 % - 20 % respectivamente).

```
train_ind <- sample(seq_len(nrow(data)), size = 0.8*dim(data)[1])
train <- data[train_ind, ]
test <- data[-train_ind, ]
```

Estimamos el modelo de regresión logística mediante el comando "glm" utilizando las variables identificadas como con mayor correlación en el primer análisis realizado sobre el conjunto "train". A continuación, mostraremos por pantalla las características más destacadas del modelo estimado. Podemos ver que todas las variables utilizadas son, hasta cierto punto, significativas.

```
out_model <- glm(outcome ~ pulse + packed_cell_volume + num_lesion +
                   abdominal_distension_moderate,
                 data = train, family = "binomial")
summary(out_model)
```

```
##
## Call:
## glm(formula = outcome ~ pulse + packed_cell_volume + num_lesion +
##     abdominal_distension_moderate, family = "binomial", data = train)
```

```
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2134  -0.7555  -0.4322   0.8392   2.3583
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                   -6.619394   0.908918  -7.283 3.27e-13 ***
## pulse                          0.018489   0.006821   2.711 0.006715 **
## packed_cell_volume             0.074858   0.017648   4.242 2.22e-05 ***
## num_lesion                     0.994079   0.461839   2.152 0.031363 *
## abdominal_distension_moderate  1.157666   0.326615   3.544 0.000393 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 346.40  on 261  degrees of freedom
## Residual deviance: 259.76  on 257  degrees of freedom
## AIC: 269.76
##
## Number of Fisher Scoring iterations: 5
```

Para poder comprobar la precisión del modelo estimado, tenemos que cargar la librería "caret".

```
if (!require("caret")) install.packages("caret")
library(caret)
```

Calcularemos ahora la matriz de confusión sobre los datos de test. Para ello, aplicaremos antes la predicción del modelo estimado sobre el conjunto de datos de test. El resultado arrojado por la matriz es que el modelo tiene más de un 70 % de precisión.

```
test_pred <- predict.glm(out_model, test, type="response")
confusionMatrix(factor(round(test_pred)), factor(test$outcome))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 36 10
##          1  9 11
##
##                Accuracy : 0.7121
##                  95% CI : (0.5875, 0.817)
##     No Information Rate : 0.6818
##     P-Value [Acc > NIR] : 0.3511
##
##                   Kappa : 0.328
##  Mcnemar's Test P-Value : 1.0000
##
##             Sensitivity : 0.8000
##             Specificity : 0.5238
##          Pos Pred Value : 0.7826
```

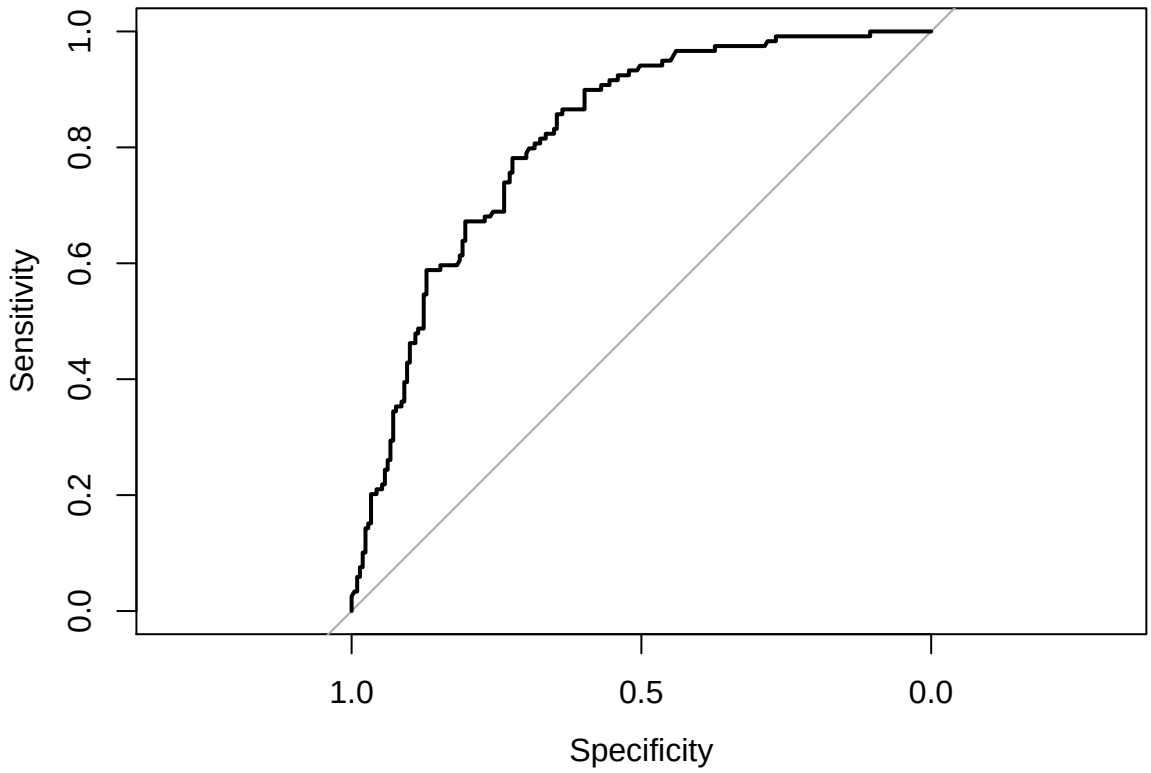
```
##          Neg Pred Value : 0.5500
##              Prevalence : 0.6818
##          Detection Rate : 0.5455
##    Detection Prevalence : 0.6970
##       Balanced Accuracy : 0.6619
##
##        'Positive' Class : 0
##
```

Cargamos la librería "pROC" para dibujar la curva ROC del modelo estimado.

```
if (!require("pROC")) install.packages("pROC")
library(pROC)
```

Por último, representaremos la curva ROC del modelo estimado. Podemos comprobar como la curva resultante nos confirma los buenos resultados obtenidos en la comprobaciones anteriores.

```
curva_roc <- roc(outcome ~ predict.glm(out_model, data, type = "response"), data)
plot(curva_roc)
```



6. Conclusiones

A lo largo de esta práctica, hemos podido extraer una serie de reflexiones relativas a los casos de cólicos en caballos. En primer lugar, cabe destacar la cantidad de técnicas diferentes que se han tenido que utilizar para poder extraer un dataset valioso para la formación de modelos estadísticos. Esto nos hace pensar que se necesita una mejora en los procedimientos de recogida de datos. Un claro ejemplo es la variable de proteínas totales en abdomcentesis, en donde nos encontramos que faltan más del 60 % de los casos. A pesar de esto, después de realizar las correspondientes técnicas de imputación sobre las variables se pudo extraer un conjunto de datos interesante para nuestro estudio. Otra conclusión relativa al pretratamiento de los datos en un proyecto de análisis de datos como este es la cantidad de tiempo y recursos que hay que destinar a preparar un dataset en condiciones para extraer resultados interesantes.

Una vez se han tratado los datos, hemos podido ver que existe una distinta variabilidad del valor total de las proteínas en sangre entre los caballos que han sido sometidos a un procedimiento quirúrgico. Esto puede ser de mucha utilidad para saber cómo tratar al animal después de una operación de este estilo, por ejemplo.

Por último, se ha estudiado la variable outcome (destino final del caballo). Así, con el gráfico de correlación podemos ver de una manera sencilla, qué variables ejercen una mayor influencia sobre el resultado del cólico. Además, se ha creado un modelo de regresión logística con el cual podemos predecir esta variable con una precisión bastante alta utilizando simplemente el pulso, el número de lesiones, la presencia de distensión abdominal moderada y el número de células rojas en sangre (todas estas significativas para el modelo). Este tipo de modelos puede ser de gran utilidad para poder priorizar los tratamientos que se le aplican a los caballos ingresados en función de sus parámetros vitales.