Introducción a la limpieza y análisis de los datos

PID_00265704

Laia Subirats Maté
Diego Oswaldo Pérez Trenard
Mireia Calvo González

Tiempo mínimo de dedicación recomendado: 6 horas

Universitat Oberta de Catalunya

Laia Subirats Maté

Ingeniera de Telecomunicaciones por la Universidad Pompeu Fabra (2008), Máster en Telemática por la Universidad Politécnica de Cataluña (2009) y doctora en Informática por la Universidad Autónoma de Barcelona (2015). Desde 2009, trabaja como investigadora en Eurecat (Centro Tecnológico de Cataluña) aplicando la ciencia de datos a diferentes campos como la salud, el medio ambiente o la educación. Desde 2016, colabora con la UOC como docente en el Máster de Data Science y en el grado de Informática. Es especialista en inteligencia artificial, ciencia de datos, salud digital y representación del conocimiento.

Diego Oswaldo Pérez Trenard

Ingeniero electrónico por la Universidad Simón Bolívar (2015), especialización en High Tech Imaging (HTI) por la Universidad Télécom SudParis (2014) y doctor en Señales, Imágenes y Visión por la Universidad de Rennes 1 (2018). Desde 2014, ha trabajado como ingeniero de investigación y desarrollo en el Instituto Nacional de Salud e Investigación Médica (INSERM) y en el Laboratorio de Procesamiento de Señales e Imágenes (LTSI), aplicando conocimientos en electrónica y en procesamiento de datos al estudio de diferentes enfermedades neurológicas, cardíacas y respiratorias. Desde 2018, colabora como docente en el máster de Data Science de la UOC.

Mireia Calvo González

Ingeniera de telecomunicaciones por la Universidad Politécnica de Cataluña (2011), Máster en Ingeniería Biomédica por la Universidad de Barcelona y la Universidad Politécnica de Cataluña (2014) y Doctora en Procesamiento de señales y telecomunicaciones por la Universidad de Rennes 1 y en Ingeniería Biomédica por la Universidad Politécnica de Cataluña (2017). Desde 2012 ha trabajado como investigadora en diferentes entornos académicos, clínicos e industriales, aplicando el procesamiento de datos al estudio de diferentes enfermedades cardíacas y respiratorias. Desde 2017 colabora con la UOC como docente en el Máster de Data Science.

El encargo y la creación de este recurso de aprendizaje UOC han sido coordinados por la profesora: Isabel Guitart Hormigo (2019)

Primera edición: septiembre 2019
© Mireia Calvo, Diego Pérez, Laia Subirats
Todos los derechos reservados
© de esta edición, FUOC, 2019
Av. Tibidabo, 39-43, 08035 Barcelona
Realización editorial: FUOC

Ninguna parte de esta publicación, incluido el diseño general y la cubierta, puede ser copiada, reproducida, almacenada o transmitida de ninguna forma, ni por ningún medio, sea este eléctrico, químico, mecánico, óptico, grabación, fotocopia, o cualquier otro, sin la previa autorización escrita de los titulares de los derechos.

Índice

Introducción

En la actualidad, enormes cantidades de datos son almacenados a diario, por lo que la aplicación de métodos robustos que permitan analizar y extraer información, y posteriormente conocimiento de dichos datos es de suma importancia. Cada vez que compramos por Internet, por ejemplo, generamos una serie de datos relacionados con el proceso de compra que, tras ser limpiados y analizados, se convierten en información útil para los propietarios de los *ecommerce*. Gracias al conocimiento extraído de observar tendencias y patrones de comportamiento en los diferentes tipos de clientes, se pueden identificar sus gustos para así mejorar su experiencia de compra y aumentar, en última instancia, el número de ventas.

Cabe destacar que, para dotar de robustez a los análisis aplicados con el objetivo de generar conocimiento, es clave la calidad de los datos analizados.

La **limpieza de datos**, o *data cleaning* en inglés, es el conjunto de procesos que permiten corregir o eliminar aquellas muestras erróneas de una base de datos. Estos procesos permiten identificar datos incompletos, incorrectos, inexactos o no pertinentes, con el fin de eliminarlos o corregirlos y así obtener bases de datos de mayor calidad.

Asimismo, verificar que los datos cumplen las suposiciones requeridas por las pruebas estadísticas aplicadas es fundamental. Los análisis estadísticos más comúnmente utilizados, como la correlación de Pearson o la prueba t de Student, asumen ciertas características o suposiciones sobre los datos que deben cumplirse para que dichas técnicas, y por lo tanto las conclusiones extraídas, sean válidas. Algunas de las suposiciones más habituales son el hecho de que los datos se encuentren distribuidos normalmente, así como que los grupos de datos presenten varianzas similares.

Aunque la robustez de los métodos de análisis se mide generalmente en la tasa de error de Tipo I, el *data cleaning* también puede afectar de forma significativa a la potencia estadística, el tamaño del efecto y la exactitud a la hora de aplicar dichos métodos, y por lo tanto a su replicabilidad, así como a la minimización de la tasa de error de Tipo II. Los datos erróneos no solo pueden conducir a la violación de suposiciones en los datos, como la normalidad o la homogeneidad de la varianza (homocedasticidad), sino que también pueden llevar a la estimación errónea de parámetros y efectos sin causar una desviación significativa de dichas suposiciones. Tratar eficazmente los valores extre-

mos (*outliers*) de una muestra generalmente mejorará la potencia estadística y el tamaño del efecto y disminuirá los errores de Tipo I y de Tipo II, por lo que tenderá a mejorar el resultado de los análisis y estimaciones.

No obstante, esta importante etapa de preprocesado no siempre recibe la atención necesaria. En un trabajo de revisión bibliográfica realizado sobre los artículos publicados durante 2009 en las revistas científicas de la American Psychological Association (APA, por sus siglas en inglés), se reportó que solo entre el 22 % y el 38 % de los trabajos hacían referencia a algún proceso de *data cleaning*. Entre el 16 % y el 18 % de los estudios reportaron análisis de valores extremos (*outliers*), entre el 10 % y el 32 % verificaron la distribución de los datos, y entre el 32 % y el 45 % reportaron la aplicación de algún método para gestionar los datos perdidos. Esto no debe interpretarse como que menos de la mitad de los trabajos realizaron algún tipo de limpieza en los datos analizados, pero dado su impacto en los resultados, resulta sorprendente que estos procesos de limpieza no se describan siempre en detalle, tal y como se hace con los métodos de análisis aplicados. Por ejemplo, ante una base de datos que contenga un número significativo de datos perdidos, será relevante describir si estos datos se eliminaron o imputaron y, de ser así, mediante qué método, ya que esto modificará los resultados y, por lo tanto, las conclusiones extraídas de dichos resultados.

Fuente bibliográfica

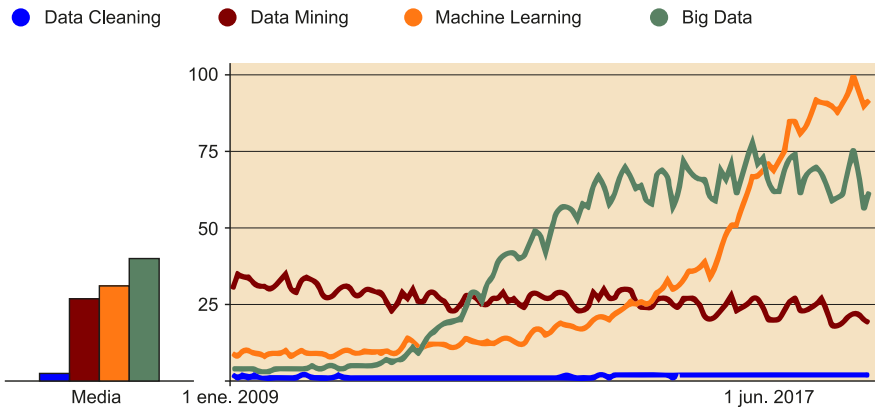
Osborne, Jason W.; Kocher, Brady, Tillman, David (2012). «Sweating the small stuff: do authors in APA journals clean data or test assumptions (and should anyone care if they do)?» [conferencia]. En: *Annual meeting of the Eastern Education Research Association* (2012: Hilton Head, SC).

De hecho, se estima que el 80 % del trabajo de un científico de datos es invertido en procesos de limpieza. No obstante, la figura 1, extraída de Google Trends, muestra como el *data cleaning* no ha conseguido capturar la misma atención que el *big data*, el *data mining*, o el *machine learning* a lo largo de la última década.

Bibliografía recomendada

Squire, Megan (2015). *Clean Data*. Birmingham: Packt Publishing.

Figura 1. Popularidad de términos relacionados con los datos, entre enero de 2009 y 2019



Fuente: Google Trends.

La **minería** o **exploración de los datos** (*data mining*, en inglés) es el análisis automático o semiautomático de los datos, con el objetivo de descubrir información útil de forma eficaz, escalable y flexible.

En función del contexto, este análisis puede perseguir diferentes objetivos: identificar grupos de registros de datos (*clustering*), detectar registros poco usuales (anomalías) o identificar dependencias (minería por reglas de asociación), entre otros. Aunque tradicionalmente se realizaba mediante métodos estadísticos, actualmente estos métodos se combinan con otras técnicas procedentes de la inteligencia artificial, el aprendizaje automático y los sistemas de bases de datos, que cada vez gozan de mayor popularidad.

Todo proceso de descubrimiento de conocimiento a partir de una serie de datos incluirá, por tanto, la limpieza de los mismos, así como su análisis. Por ello, tras tratar la captura y el almacenamiento de los datos (ver módulo *web scraping*), este módulo didáctico se centra en los procesos de limpieza y análisis de las bases de datos almacenadas.

El apartado "Limpieza de datos" detalla las etapas de limpieza más habituales; pasando por la integración, selección y reducción de los datos, posibles transformaciones como la discretización, así como la gestión de datos perdidos (*missing data*) y de valores extremos (*outliers*).

A continuación, en el apartado "Análisis de datos", se enumeran las principales técnicas de análisis que permiten explorar los datos, con el objetivo de identificar tendencias y patrones de interés. Desde el análisis estadístico descriptivo e inferencial, pasando por la regresión, la correlación, los análisis de supervivencia y enumerando algunos modelos supervisados y no supervisados. Asimismo, se comenta brevemente la representación visual de los resultados como método adicional de análisis.

Por último, tras un breve resumen de los contenidos más relevantes del módulo, se proponen algunos ejercicios de autoevaluación, así como sus soluciones, con los que poder comprobar la asimilación de los principales conceptos que aquí se presentan.

Este módulo didáctico se acompaña de un repositorio Github donde se incluye el código descargable de algunos de los ejemplos presentados. Aunque estos ejemplos se proporcionan en R, los procesos de limpieza y análisis descritos en este material pueden implementarse en otros lenguajes de programación como Python.

Todos los ejemplos están basados en conjuntos de datos disponibles en R. No obstante, muchos de los métodos utilizados se basan en técnicas *randomizadas*, por lo que, al no fijar semillas específicas en cada uno de los ejemplos, los resultados mostrados en este material no tienen por qué coincidir siempre con los que se pueden encontrar.

Asimismo, los ejemplos propuestos ilustran algunas de las funcionalidades de los paquetes presentados, pero se recomienda consultar la ayuda y documentación de cada una de las funciones proporcionadas, para profundizar en sus funcionalidades y poder sacar así el máximo partido.

Objetivos

En este material didáctico se proporcionan las herramientas fundamentales que permitirán asimilar los siguientes objetivos:

1. Comprender el significado y los potenciales beneficios de la limpieza de datos.
2. Conocer la dificultad de limpiar una base de datos determinada.
3. Conocer los principales métodos para la limpieza de datos.
4. Conocer las principales técnicas de exploración de los datos.
5. Saber aplicar procesos de limpieza, validación y análisis utilizando R.
6. Ser capaz de extraer información útil de las bases de datos disponibles.

1. Limpieza de datos

En el ciclo de vida de los datos, la etapa de limpieza (o preprocesado) se compone del conjunto de procesos que permiten identificar aquellos registros incompletos, incorrectos, inexactos y/o no pertinentes de un conjunto de datos, con el fin de eliminarlos o corregirlos. Esta etapa permite mejorar la calidad de los datos, por lo que será extremadamente importante a la hora de sacar el máximo rendimiento a su posterior análisis.

Los siguientes apartados describen algunos de los métodos de limpieza más utilizados. No obstante, es importante destacar que estos se aplicarán en función del contexto, es decir, del tipo de datos a tratar y del análisis que se quiera realizar posteriormente. Así, aunque se enumeran varios métodos, no siempre será necesario aplicarlos todos, del mismo modo que podrán añadirse otros métodos al proceso de limpieza.

1.1. Integración

La **integración** o **fusión de los datos** consiste en la combinación de datos procedentes de múltiples fuentes, con el fin de crear una estructura de datos coherente y única que contenga mayor cantidad de información.

Bibliografía recomendada

Han, Jiawei; Kamber, Micheline; Pei, Jian (2011). *Data mining: Concepts and Techniques*. Waltham: Elsevier.

Esta fusión puede realizarse de forma horizontal, es decir, añadiendo nuevos atributos a la base de datos original. Dado que las diferentes fuentes no siempre tendrán el mismo número de registros y estos no estarán ordenados siguiendo el mismo criterio, antes de la integración será fundamental identificar un atributo que sirva de «identificador único» y, por lo tanto, que relacione adecuadamente los nuevos atributos con los registros existentes. Con el objetivo de evitar inconsistencias y redundancias, es importante destacar que este identificador podrá tener diferentes nombres y formatos en cada una de las fuentes. Si utilizáramos, por ejemplo, el nombre de los diferentes clientes como identificador, este podría escribirse de diferentes formas en cada una de las bases de datos (solo con el primer apellido, con o sin el nombre, especificando el título, etc.). También podría ocurrir que diferentes clientes tuvieran el mismo nombre. Por eso, para evitar duplicidades, es habitual utilizar identificadores numéricos sintéticos que identifiquen unívocamente cada uno de los registros; en el ejemplo, cada uno de los clientes.

En el siguiente ejemplo se fusiona la información de un grupo de autores con la de los libros que han escrito, tomando los atributos «name» y «surname» como identificadores. En este caso, no podemos distinguir si las dos entradas de Ripley hacen referencia al mismo autor.

Ejemplo

```
>authors <- data.frame( surname = c("Tukey", "Venables", "Tierney", "Ripley",
"McNeil"), nationality = c("US", "Australia", "US", "UK", "Australia"),
retired = c("yes", rep("no", 4)))

>books <- data.frame( name = c("Tukey", "Venables", "Tierney", "Ripley",
"Ripley", "McNeil"), title = c("Exploratory Data Analysis", "Modern Applied
Statistics ...", "LISP-STAT", "Spatial Statistics", "Stochastic Simulation",
"Interactive Data Analysis"), other.author = c(NA, "Ripley", NA, NA, NA, NA))

>example1<-merge(authors, books, by.x="surname", by.y="name")
>example1

  name                          title  other.author nationality   retired
1   McNeil  Interactive Data Analysis          <NA>   Australia        no
2   Ripley         Spatial Statistics          <NA>          UK        no
3   Ripley      Stochastic Simulation          <NA>          UK        no
4  Tierney                  LISP-STAT          <NA>          US        no
5    Tukey  Exploratory Data Analysis          <NA>          US       yes
6 Venables Modern Applied Statistics ...    Ripley    Australia        no
```

Por otro lado, se pueden realizar fusiones verticales con el objetivo de incluir nuevos registros a una base de datos original. Por ejemplo, cuando el inventario de las diferentes tiendas de una cadena se recoja en bases de datos separadas, estas pueden querer integrarse para analizarlas conjuntamente. El siguiente ejemplo muestra la fusión vertical de dos bases de datos, mediante la función `rbind()`:

Ejemplo

```
>data1 <- data.frame(CustomerId = c(1:6), Product = c(rep("Oven", 3),
rep("Television", 3)))

>data2 <- data.frame(CustomerId = c(4:7), Product = c(rep("Television", 2),
rep("Air conditioner", 2)))

>example2<-rbind(data1,data2)
example2

   CustomerId     Product
1         1           Oven
2         2           Oven
3         3           Oven
4         4     Television
5         5     Television
6         6     Television
7         4     Television
8         5     Television
9         6 Air conditioner
10        7 Air conditioner
```

En este tipo de fusión será muy importante que el formato de las bases de datos a integrar sea el mismo; de lo contrario, aparecerán inconsistencias y errores en la estructura de datos final. Así, puede darse el caso de que en una de las tiendas

no proceda recoger uno de los atributos porque se trate de un servicio que solo se ofrezca en otras tiendas de la cadena. Si este atributo solo aparece en alguna de las bases de datos de origen, será necesario añadirlo también en aquellas tiendas donde no proceda recogerlo, para después dejar el campo vacío. De este modo, se mantendrá el mismo formato en todas las bases a fusionar.

Asimismo, la información de las diferentes tiendas puede recogerse en formatos distintos. Por ejemplo, si estas tiendas se encuentran en Europa y Estados Unidos, los atributos que indiquen medidas o precios se recogerán muy probablemente en unidades diferentes, por lo que habrá que aplicar un proceso previo de conversión de modo que todos los registros se representen en un mismo sistema de medida y en una misma moneda.

Por último, una vez integrados los datos, siempre será necesario verificar que tanto la fusión como las conversiones previas se han realizado correctamente, así como que no existen elementos duplicados en la nueva base de datos. R proporciona las funciones `duplicated()` y `unique()` para identificar los registros duplicados. Asimismo, la función `distinct()` del paquete `dplyr` permite analizar esta duplicidad acotando la búsqueda solo a aquellos atributos especificados. Esta herramienta es particularmente útil cuando alguno de los atributos debe forzosamente mostrar valores únicos para cada registro.

Ejemplo

```
>unique(example1)

  name                      title   other.author nationality    retired
1   McNeil  Interactive Data Analysis      <NA>    Australia         no
2   Ripley         Spatial Statistics      <NA>           UK         no
3   Ripley      Stochastic Simulation      <NA>           UK         no
4  Tierney                  LISP-STAT      <NA>           US         no
5    Tukey  Exploratory Data Analysis      <NA>           US        yes
6 Venables Modern Applied Statistics ...  Ripley   Australia         no

>example1 %>% distinct(surname, .keep_all=TRUE)

  name                      title   other.author nationality   retired
1   McNeil  Interactive Data Analysis      <NA>    Australia        no
2   Ripley         Spatial Statistics      <NA>           UK        no
3  Tierney                  LISP-STAT      <NA>           US        no
4    Tukey  Exploratory Data Analysis      <NA>           US       yes
5 Venables Modern Applied Statistics ...  Ripley   Australia        no
```

1.2. Selección

Una de las primeras etapas en el preprocesado de los datos es el **filtrado** o **selección de datos** de interés. Para un estudio en particular nos puede interesar analizar solo aquellas personas mayores de 50 años, o solo la muestra procedente de un municipio concreto.

En esta fase también es habitual realizar una exploración de los datos (*screening*, en inglés), con el objetivo de analizar globalmente sus características e identificar fuertes correlaciones entre atributos, de modo que se pueda prescindir de aquella información más redundante. Asimismo, se suele aprovechar este primer *screening* para analizar las suposiciones en los datos requeridas por las pruebas estadísticas que se aplicarán posteriormente.

Por otro lado, y contrariamente a la selección, el preprocesado de los datos también puede incluir la **creación de nuevas variables** a partir de la extracción de características de los datos originales. Por ejemplo, una variable comúnmente utilizada en medicina es el índice de masa corporal (IMC), que relaciona la altura y el peso de un individuo según la siguiente fórmula:

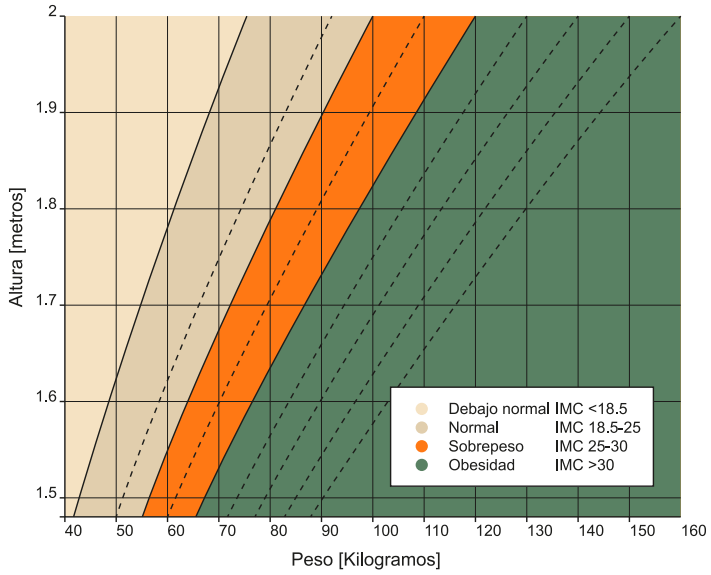
Bibliografía recomendada

Han, Jiawei; Kamber, Micheline; Pei, Jian (2011). *Data Mining: Concepts and Techniques*. Waltham: Elsevier.

$$IMC = \frac{masa}{altura^2} \qquad (1)$$

Este parámetro se utiliza como indicador de grasa corporal y por tanto de obesidad, factor de riesgo en numerosas enfermedades como la diabetes, la hipertensión, el cáncer o la apnea del sueño, entre otras. La figura 2 muestra la relación entre el IMC y los diferentes grados de obesidad.

Figura 2. Índice de masa corporal (IMC)



Fuente: *Wikipedia*.

1.3. Reducción

Trabajar con grandes cantidades de datos puede convertir la tarea de análisis en un proceso muy complejo e incluso impracticable. Las técnicas de **reducción de datos** permiten obtener una representación reducida de los mismos,

manteniendo la integridad de la muestra original. Así, los análisis aplicados sobre la muestra de datos reducida producirán los mismos (o muy similares) resultados que si se aplicaran sobre la muestra total.

Los siguientes apartados describen en detalle algunas de las técnicas de reducción más comúnmente utilizadas, agrupándolas en dos grupos principales: reducción de la dimensionalidad y reducción de la cantidad.

Algunos autores, como Jiawei *et al*. (2011), además añaden la compresión de los datos como un grupo de métodos diseñados para la reducción de los mismos. Como su nombre indica, estos métodos obtienen una representación comprimida de los datos originales que, cuando son capaces de reconstruir la muestra original por completo, se denominan métodos sin pérdidas (*lossless*). En tratamiento de imágenes, existen diversos formatos de almacenamiento que pueden comprimir dichas imágenes, con o sin pérdidas. El formato PNG, por ejemplo, permite comprimir una imagen de forma totalmente reversible, por lo que la imagen recuperada es idéntica a la original. Por otro lado, JPG es el método de compresión más utilizado en fotografía ya que, tras eliminar la información menos apreciable, permite obtener importantes compresiones manteniendo calidades de imagen muy elevadas.

1.3.1. Reducción de la dimensionalidad

Este conjunto de métodos tiene por objetivo reducir el número de atributos bajo consideración. Pueden dividirse en métodos paramétricos y no paramétricos. Los primeros estiman los datos mediante un modelo, de modo que solo es necesario almacenar los parámetros de dicho modelo en lugar de la base de datos original. Algunos ejemplos de este tipo son los modelos de regresión, que se estudiarán en el apartado 2.2.3. Regresión.

Dentro de los no paramétricos, entre los más utilizados se encuentran el análisis de componentes principales y las transformaciones *wavelet*, los cuales proyectan los datos originales en un espacio de dimensiones más reducido. Por otro lado, en la selección de subconjuntos de atributos (*Attribute subset selection*), se detectan y eliminan aquellos atributos más irrelevantes y/o redundantes del conjunto de datos. En Jiawei *et al*. (2011), se detalla más en profundidad cada uno de estos métodos.

Sin entrar en detalle en la explicación teórica del **análisis de componentes principales** (ACP), esta técnica permite describir un conjunto de datos de n atributos, en términos de m nuevas variables no correlacionadas, o componentes principales, donde $m < n$. Estas componentes se ordenan según la cantidad de varianza de los datos originales que describen. Técnicamente, el ACP es una transformación lineal que busca la proyección según la cual los datos queden mejor representados en términos de mínimos cuadrados, convirtien-

do un conjunto de observaciones de atributos posiblemente correlacionados en un conjunto de nuevas variables sin correlación lineal, llamadas componentes principales.

El siguiente ejemplo se basa en el conjunto de datos `mtcars`, disponible en R, que comprende los datos de 11 atributos, para 32 modelos de coche. Dado que el ACP funciona principalmente con datos numéricos, se excluyen las variables categóricas de la muestra, dejando 9 atributos para los 32 modelos. A continuación, se aplica el ACP mediante la función `prcomp()`, centrando y escalando los datos para que el hecho de combinar valores a diferentes escalas no afecte significativamente en los resultados.

Ejemplo

```
>mtcars.pca <- prcomp(mtcars[,c(1:7,10,11)], center = TRUE, scale = TRUE)
>summary(mtcars.pca)

Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6     PC7    PC8     PC9
Standard deviation     2.3782 1.4429 0.71008 0.51481 0.42797 0.35184 0.32413 0.2419 0.14896
Proportion of Variance 0.6284 0.2313 0.05602 0.02945 0.02035 0.01375 0.01167 0.0065 0.00247
Cumulative Proportion  0.6284 0.8598 0.91581 0.94525 0.96560 0.97936 0.99103 0.9975 1.00000
```

El resultado son 9 componentes principales (PC1-PC9), cada una de las cuales explica un porcentaje de varianza del *dataset* original. Así, la primera componente principal explica prácticamente 2/3 de la varianza total y las dos primeras componentes describen el 86 % de la varianza. Dado que las cuatro primeras componentes ya explican el 95 % de la varianza, se podría trabajar solo con este subconjunto (PC1-PC4), que contiene prácticamente la totalidad de la información contenida en el conjunto de datos original. Además de la solución obtenida para cada registro en el nuevo espacio de componentes principales (`mtcars.pca$x`), entre otras informaciones, el resultado de aplicar esta función permite analizar los pesos asociados a cada atributo en la transformación lineal aplicada por el ACP resultante.

Ejemplo

```
> mtcars.pca$rotation
             PC1         PC2         PC3          PC4         PC5         PC6         PC7         PC8          PC9
mpg  -0.3931477  0.02753861 -0.22119309 -0.006126378 -0.3207620  0.72015586 -0.38138068 -0.12465987  0.11492862
cyl   0.4025537  0.01570975 -0.25231615  0.040700251  0.1171397  0.22432550 -0.15893251  0.81032177  0.16266295
disp  0.3973528 -0.08888469 -0.07825139  0.339493732 -0.4867849 -0.01967516 -0.18233095 -0.06416707 -0.66190812
Hp    0.3670814  0.26941371 -0.01721159  0.068300993 -0.2947317  0.35394225  0.69620751 -0.16573993  0.25177306
drat -0.3118165  0.34165268  0.14995507  0.845658485  0.1619259 -0.01536794  0.04767957  0.13505066  0.03809096
Wt    0.3734771 -0.17194306  0.45373418  0.191260029 -0.1874822 -0.08377237 -0.42777608 -0.19839375  0.56918844
qsec -0.2243508 -0.48404435  0.62812782 -0.030329127 -0.1482495  0.25752940  0.27622581  0.35613350 -0.16873731
gear -0.2094749  0.55078264  0.20658376 -0.282381831 -0.5624860 -0.32298239 -0.08555707  0.31636479  0.04719694
carb  0.2445807  0.48431310  0.46412069 -0.214492216  0.3997820  0.35706914 -0.20604210 -0.10832772 -0.32045892
```

1.3.2. Reducción de la cantidad

Este grupo de métodos sustituye el volumen de datos original por una representación alternativa con un volumen de muestras o registros menor. Algunos ejemplos de este tipo son los histogramas, el *clustering* o el *sampling*.

Bibliografía recomendada

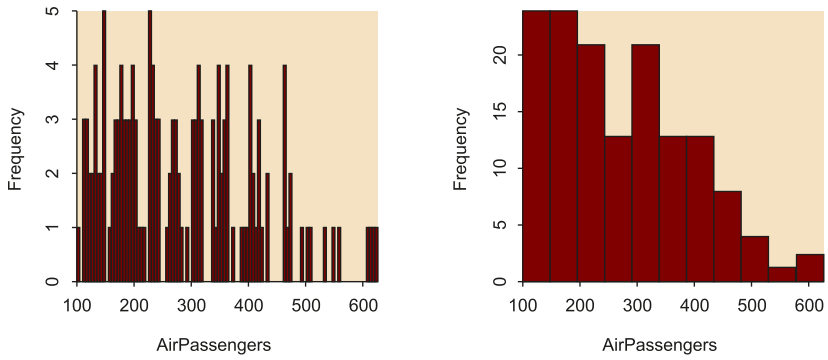
Han, Jiawei; Kamber, Micheline; Pei, Jian (2011). *Data Mining: Concepts and Techniques*. Waltham: Elsevier.

Los **histogramas** dividen la distribución de datos de un atributo en subconjuntos disociados, denominados contenedores (*bins*). Aunque estos *bins* pueden representar un único par de atributo-valor/frecuencia cuando representan rangos continuos para un atributo dado, también permiten reducir la cantidad de datos. El siguiente ejemplo muestra el histograma de un mismo atributo (`AirPassengers`, disponible en R), cuando se reduce la cantidad de *bins* representados, de 100 a 10, para reducir así el tamaño de los datos.

Ejemplo

```
> hist(AirPassengers,breaks=100)
> hist(AirPassengers,breaks=10)
```

Figura 3. Histograma de `AirPassengers`, cuando se representan 100 y 10 bins



Por otro lado, las técnicas de *clustering* dividen los registros de datos en grupos, o clústeres, de modo que los registros dentro de un mismo clúster sean similares entre sí y diferentes a los registros de otros clústeres. La similitud se define generalmente en términos de cuán cerca se encuentran los registros en el espacio, basándose en una función de distancia.

Aunque el *clustering* se explica con más detalle en el apartado 2.5. Modelos no supervisados ya que se trata de un método de análisis, estas representaciones también pueden ser utilizadas para reemplazar los datos originales, con el objetivo de reducir su tamaño. No obstante, la eficacia de esta técnica dependerá de la naturaleza de los datos y de su capacidad para representarse en grupos.

Finalmente, el muestreo (*sampling*, en inglés) puede ser utilizado como una técnica de reducción de la cantidad ya que permite que un gran conjunto de datos sea representado por una muestra (o subconjunto) de datos aleatorios mucho más pequeña. Esta reducción puede realizarse mediante diferentes métodos, representados en la figura 4:

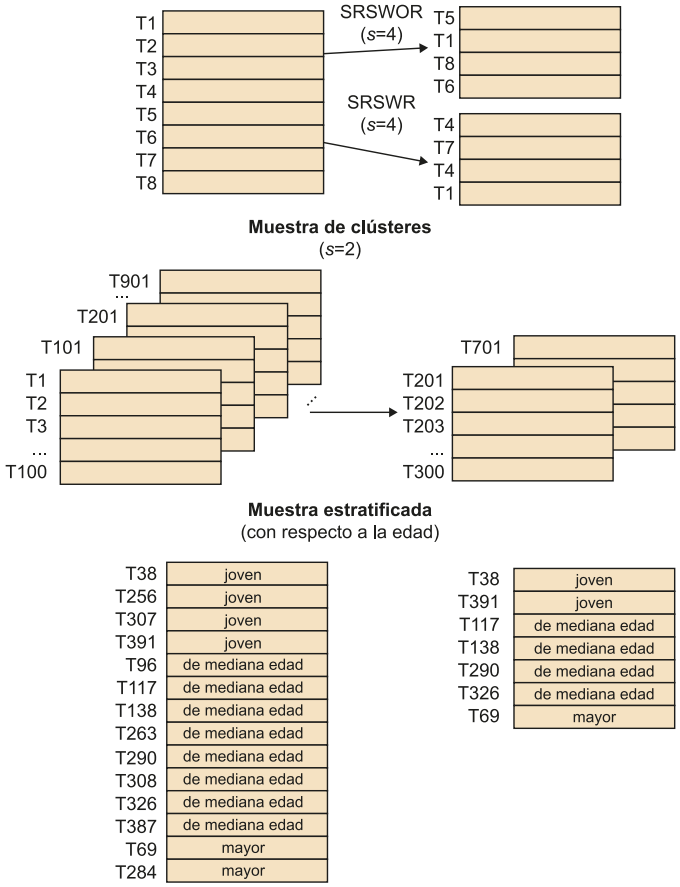
- **Muestra aleatoria simple sin sustitución** (SRSWOR, por sus siglas en inglés *Simple Random Sample WithOut Replacement*): de los N registros que forman el conjunto D, se escogen s ($s < N$), donde la probabilidad de cada registro de ser seleccionado es la misma ($1/N$).

- **Muestra aleatoria simple con sustitución** (SRSWR, por sus siglas en inglés *Simple Random Sample With Replacement*): en este caso, cada vez que se selecciona un registro, este se vuelve a tener en cuenta en la siguiente selección, por lo que cada registro puede escogerse varias veces.

- **Muestra de clústeres**: la selección se realiza por clústeres, es decir, si los registros en el conjunto D se agrupan en M clústeres diferentes, se seleccionan s clústeres de forma aleatoria, donde $s < M$.

- **Muestra estratificada**: si el conjunto D se divide en partes perfectamente disociadas llamadas estratos, este método selecciona una muestra aleatoria para cada estrato del conjunto. Esto permite generar una muestra representativa de los datos cuando estos son muy sesgados.

Figura 4. El muestreo (*sampling*) como método de reducción de los datos



Adaptación de Jiawei *et al.* (2011).

El paquete `sampling` permite trabajar con este conjunto de métodos en R. Algunas funciones interesantes son `srswr()`, `srswor()`, `cluster()` o `strata()`. El siguiente ejemplo muestra el código utilizado para seleccionar 3 clústeres del conjunto de datos `swissmunicipalities`, disponible en R, tomando la variable `REG` como aquella que separa los $M = 7$ clústeres o grupos de datos y utilizando el método de muestreo aleatorio simple sin sustitución (SRSWOR).

Ejemplo

```
>data(swissmunicipalities)
>cl=cluster(swissmunicipalities,clustername=c("REG"),size=3,method="srswor")
>getdata(swissmunicipalities, cl)
```

1.4. Conversión

En la etapa de **conversión**, los datos son transformados con el objetivo de que el análisis posterior sea más eficiente y/o los resultados obtenidos sean más fácilmente interpretables. Algunas técnicas de conversión habituales son la normalización, la transformación de Box-Cox o la discretización.

1.4.1. Normalización

La **normalización**, o estandarización, permite reducir el sesgo causado por la combinación de valores medidos a diferentes escalas al ajustarlos a una escala común, típicamente entre (-1,1) o entre (0,1).

Dependiendo del contexto, esta normalización se puede aplicar mediante diferentes métodos, siendo la normalización min-max y la normalización z-score los más comunes.

Bibliografía recomendada

Han, Jiawei; Kamber, Micheline; Pei, Jian (2011). *Data Mining: Concepts and Techniques*. Waltham: Elsevier.

La normalización min-max realiza una transformación lineal de los datos originales preservando la relación entre los valores del conjunto de datos original. Suponiendo A un atributo numérico con n valores observados (v_1,..., v_n), donde *minA* y *maxA* son los valores mínimo y máximo de este atributo, este método ajusta el valor v_i de A a v_i' en el rango (*minA'*, *maxA'*), por ejemplo (0,1), mediante la fórmula:

$$v_i' = \frac{v_i - minA}{maxA - minA}(maxA' - minA') + minA' \qquad (2)$$

Por otro lado, la normalización z-score transforma el atributo original basándose en su media (μ_A) y desviación estándar (σ_A) de la siguiente manera:

$$v_i' = \frac{v_i - \mu_A}{\sigma_A} \qquad (3)$$

La función `scale()` aplica este tipo de normalización z-score en R. Dado que esta técnica solo tendrá sentido en variables numéricas, el siguiente ejemplo normaliza las 9 variables numéricas del conjunto de datos `mtcars`.

Ejemplo

```
>mtcars.scaled <- scale(mtcars[,c(1:7,10,11)])
```

1.4.2. Transformación de Box-Cox

Como se menciona en la introducción, algunas pruebas estadísticas asumen ciertas suposiciones sobre los datos que deben cumplirse para que dichas pruebas, y por lo tanto las conclusiones extraídas, sean válidas. Así, para poder aplicar pruebas por contraste de hipótesis paramétricas, como la prueba t de Student:

1) Las variables de los datos analizados deben estar normalmente distribuidas.

2) Las varianzas de dichas variables deben permanecer constantes a lo largo del rango observado de alguna otra variable.

Cuando no sea así, se puede optar por utilizar una alternativa no paramétrica, como las pruebas de Wilcoxon o Mann-Whitney (apartado 2.2.1), pero esto generalmente implicará la pérdida de potencia estadística. Por ello, existe otra alternativa previa al uso de pruebas no paramétricas que consiste en convertir los datos para tratar de mejorar su normalidad y homocedasticidad: la **transformación** (o familia de transformaciones) **de Box-Cox**.

Bibliografía recomendada

Osborne, Jason W. (2013). *Best practices in data cleaning: A complete guide to everything you need to do before and after collecting your data*. Thousand Oaks, CA: Sage Publications.

Tras estimar un coeficiente de transformación λ óptimo, los diferentes valores en el nuevo conjunto de datos quedan definidos a partir de la siguiente expresión:

$$y_i^{\lambda} = \left(y_i^{\lambda} - 1\right) / \lambda \quad \lambda \neq 0;$$

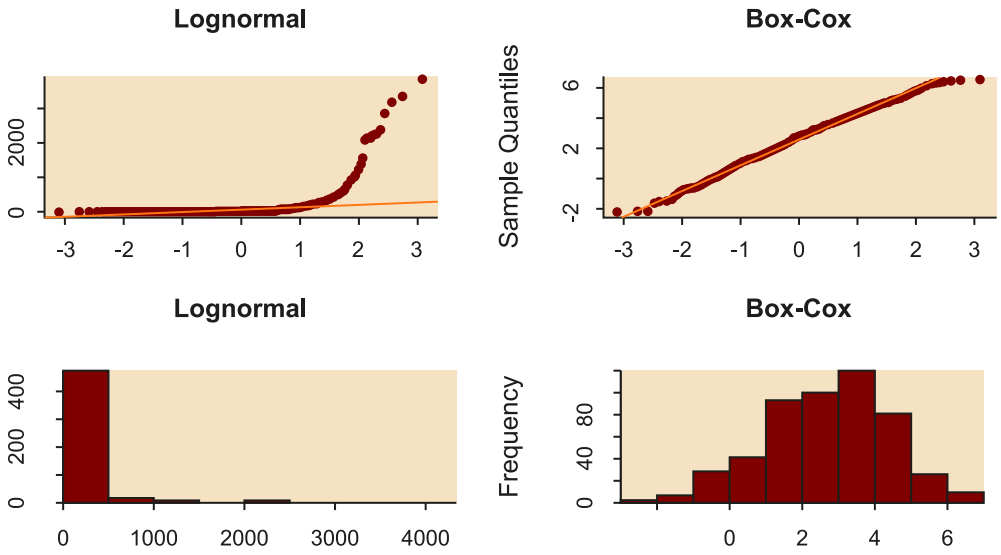
$$y_i^{\lambda} = ln\left(y_i\right) \quad \lambda = 0.$$

El paquete `DescTools` de R permite estimar el valor óptimo de λ mediante la función `BoxCoxLambda()`, para luego aplicar dicha transformación con la función `BoxCox()`. A continuación, se muestra un ejemplo donde la transformación de Box-Cox permite normalizar una serie de datos sintéticos de tipo *lognormal* (figura 5).

Ejemplo

```
>x <- rlnorm(500, 3, 2)
>x.norm <- BoxCox(x, lambda = BoxCoxLambda(x))

>par(mfrow=c(2,2))
>qqnorm(x, main="Lognormal")
>qqline(x,col=2)

>qqnorm(bx, main="Box-Cox")
>qqline(bx,col=2)

>hist(x,main="Lognormal")
>hist(bx, main="Box-Cox")
```

Figura 5. Gráfico Q-Q e histograma del conjunto de datos *lognormal*, antes y después de aplicar la transformación de Box-Cox



1.4.3. Discretización

La **discretización** consiste en reemplazar los valores numéricos de un atributo por etiquetas, categorías o niveles, que pueden ser conceptuales (niño/adulto) o rangos de valores (0-18/19-100). La **dicotomización** es un caso particular de discretización en dos etiquetas o categorías.

Aunque esta etapa de preprocesado resulta muy conveniente en ocasiones a la hora de interpretar y comparar los resultados de diferentes grupos de datos, se debe utilizar con moderación ya que al discretizar los datos se estará perdiendo información que puede resultar de interés.

Bibliografía recomendada

Osborne, Jason W. (2013). *Best practices in data cleaning: A complete guide to everything you need to do before and after collecting your data*. Thousand Oaks, CA: Sage Publications.

En este contexto, los métodos de *clustering* vuelven a ser una herramienta para discretizar un atributo numérico, dividiendo sus valores en grupos o clústeres. Esta técnica tiene en cuenta la distribución del atributo, así como la cercanía de los datos a cada grupo, por lo que permite una discretización de los datos de alta calidad.

Otro grupo de métodos ampliamente utilizados discretiza los datos aplicando criterios de *equal-width* o *equal-frequency binning*, es decir, dividiendo los datos de modo que el ancho de los rangos o las frecuencias de cada categoría sean iguales, y reemplazando los datos por las medias o medianas de cada *bin*. Aunque son métodos menos sofisticados por no tener en cuenta la información de cada categoría, su simplicidad es lo que los hace particularmente interesantes.

Bibliografía recomendada

Han, Jiawei; Kamber, Micheline; Pei, Jian (2011). *Data Mining: Concepts and Techniques*. Waltham: Elsevier.

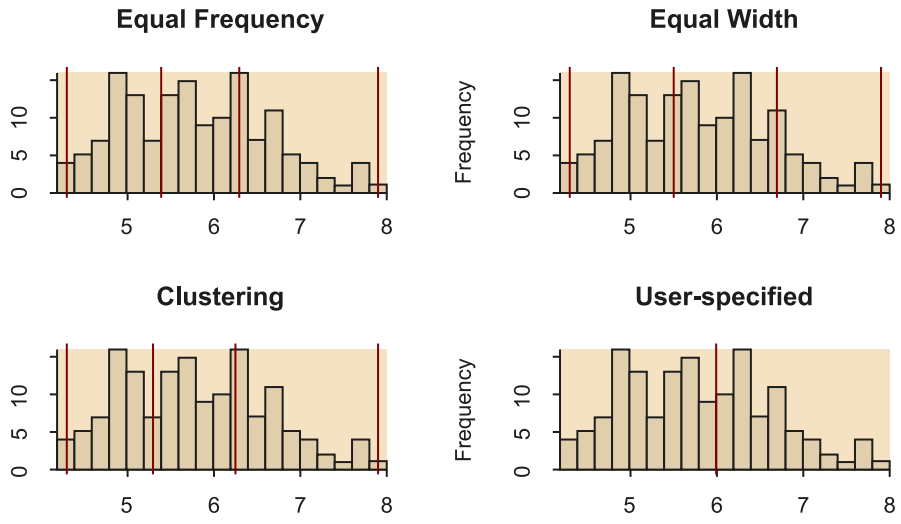
En R, el paquete `arules` permite discretizar los datos utilizando los métodos mencionados, mediante la función `discretize()`. El siguiente código muestra un ejemplo de discretización en 3 niveles (la base de datos `iris` se compone de 3 tipos de flores), teniendo en cuenta los métodos *equal-frequency*,

equal-width, *clustering* y fijando los intervalos de forma manual (*user-specified*). Se muestra la división en 3 categorías de forma gráfica (figura 6), sobre el histograma original.

Ejemplo

```
>data(iris)
>x <- iris[,1]

>par(mfrow=c(2,2))

>#equal frequency
>hist(x, breaks = 20, main = "Equal Frequency")
>abline(v = discretize(x, breaks = 3, onlycuts = TRUE), col = "red")

>#equal interval width
>hist(x, breaks = 20, main = "Equal Width")
>abline(v = discretize(x, method = "interval", breaks = 3, onlycuts = TRUE),
col = "red")

>#  k-means clustering
>hist(x, breaks = 20, main = "Clustering")
>abline(v = discretize(x, method = "cluster", breaks = 3, onlycuts = TRUE),
col = "red")

# user-specified
hist(x, breaks = 20, main = "User-specified")
abline(v = discretize(x, method = "fixed", breaks = c(-Inf, 6, Inf), onlycuts
= TRUE), col = "red")
```

Figura 6. Resultado de discretizar un conjunto de datos mediante los métodos de *equal-frequency binning*, *equal-width binning*, *clustering* y escogiendo manualmente los intervalos (*user-specified*)



1.5. Datos perdidos

Uno de los riesgos asociados a la conversión de datos es la pérdida de información. Esta puede ser parcial, cuando se reduce el tamaño o el nivel de precisión con respecto a los datos originales, pero también puede ser completa (**datos perdidos** o *missing data*).

Bibliografía recomendada

Squire, Megan (2015). *Clean Data*. Birmingham: Packt Publishing.

Los datos vacíos o no definidos pueden presentarse en distintos formatos, típicamente "", " " o NA (*Not Available* en inglés), pero en algunos contextos pueden incluso tomar valores numéricos como 0 o 999. Por tanto, será fundamental identificar en cada caso los valores que indiquen la pérdida de datos.

En el caso de valores numéricos, estos generalmente se considerarán datos perdidos cuando no formen parte del dominio del atributo. Por ejemplo, en el caso del índice de masa corporal o IMC (apartado 1.2), este nunca podrá ser 0 por lo que, si en un conjunto de datos encontramos una serie de registros con el valor del IMC a 0, este dato muy probablemente estará indicando la pérdida de información.

Asimismo, en ocasiones se puede tratar de valores vacíos legítimos, y no datos perdidos, cuando ese campo no proceda para un registro en particular. Por ejemplo, cuando se pretenda recoger información sobre los resultados de una prueba clínica y esta no se haya realizado para un paciente concreto.

Según Osborne (2013), dependiendo del contexto, existen múltiples posibles soluciones a la pérdida de datos. Por un lado, si la información es conocida y supone una inversión de tiempo aceptable, la mejor solución consiste en completar manualmente los registros faltantes.

Más información en:

Osborne, Jason W. (2013). *Best practices in data cleaning: A complete guide to everything you need to do before and after collecting your data*. Thousand Oaks, CA: Sage Publications.

Asimismo, se puede reemplazar el conjunto de valores perdidos por una misma constante o etiqueta, como por ejemplo «Desconocido». Esta técnica puede ser también útil cuando los valores perdidos tengan un significado común, como «No procede».

Otras aproximaciones reemplazan los registros perdidos por una misma medida de tendencia central, es decir, por la media o la mediana de ese atributo, dependiendo de la distribución de los datos. Esta media se puede calcular para toda la muestra o para cada una de las clases o categorías que la describan. Por ejemplo, se podrían calcular por separado las medias de hombres y mujeres.

Finalmente, otras aproximaciones se basan en la implementación de métodos probabilistas para predecir (o imputar) los valores perdidos. Algunos de estos métodos son las regresiones, las inferencias basadas en modelos bayesianos o los árboles de decisión.

Aunque las aproximaciones que tratan de imputar valores perdidos se presentan como una alternativa particularmente interesante con el objetivo de no perder información útil, deberán aplicarse con cautela y utilizando métodos adecuados, para no introducir error y falsear los resultados (Osborne, 2013). El último grupo de aproximaciones, no obstante, es el que goza de mayor popularidad ya que trata de capturar la mayor cantidad de información de los datos para predecir así los valores perdidos.

Uno de los métodos más populares es el kNN (por sus siglas en inglés, *k-Nearest Neighbours*), ya que permite predecir valores en conjuntos de datos multidimensionales formados por datos mixtos (continuos, discretos, ordinales y/o nominales). No obstante, este método es muy sensible a la elección del valor *k*. Cuando este es demasiado elevado, se incluyen valores significativamente diferentes del resultado esperado, mientras que cuando *k* es demasiado bajo esto implica la pérdida de valores significativos. Por ello, otro método más robusto que está ganando popularidad en los últimos años ya que también permite trabajar con conjuntos de datos mixtos multidimensionales es *missForest*.

En R, existen varios paquetes que permiten aplicar la imputación de datos mediante kNN, como `DMwR` mediante la función `knnImputation()`, o `VIM` mediante `kNN()`. Este último permite además representar gráficamente los valores perdidos de un conjunto de datos mediante la función `aggr()`. Asimismo, el paquete `missForest` permite aplicar este método de imputación de datos perdidos.

En el siguiente ejemplo se muestran los resultados de aplicar estas funciones (figura 7), tras utilizar la función `missForest::prodNa()` para introducir valores perdidos, de forma sintética, en el conjunto de datos `iris`.

Bibliografía recomendada

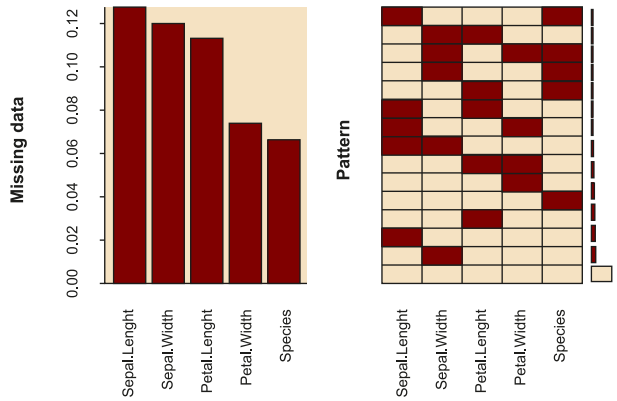
Osborne, Jason W. (2013). *Best practices in data cleaning: A complete guide to everything you need to do before and after collecting your data*. Thousand Oaks, CA: Sage Publications.

Stekhoven, Daniel J.; Bühlmann, Peter (2011, enero). «MissForest: Non-parametric missing value imputation for mixed-type data». *Bioinformatics* (vol. 28, n.º 1, págs. 112-118).

Ejemplo

```
>data(iris)
>iris.mis <- prodNA(iris, noNA = 0.1)

>aggr(iris.mis, numbers=TRUE, sortVars=TRUE, labels=names(iris.mis),
cex.axis=.7, gap=3, ylab=c("Missing data","Pattern"))
```

Figura 7. Porcentaje de valores perdidos en cada variable del conjunto de datos (izquierda). Representación de patrones en el conjunto de datos, teniendo en cuenta –en rojo– los valores perdidos (derecha)




```
>##kNN 1 (VIM package)
>kNN1.imp<-kNN(iris.mis, k=3)

>##kNN 2 (DMwR package)
>kNN2.imp <- knnImputation(iris.mis, 3)

>##missForest
```

```
>missForest.imp<-missForest(iris.mis, variablewise = TRUE)
```

1.6. Valores extremos

Los **valores extremos** (*extreme scores* o *outliers*) son aquellos datos que se encuentran muy alejados de la distribución normal de una variable o población. Son observaciones que se desvían tanto del resto que levantan sospechas sobre si fueron generadas mediante el mismo mecanismo. Asimismo, estos valores pueden afectar de forma adversa los resultados de los análisis posteriores, al incrementar el error en la varianza de los datos y sesgar significativamente los cálculos y estimaciones.

Bibliografía recomendada

Osborne, Jason W. (2010, marzo). «Data cleaning basics: Best practices in dealing with extreme scores». *Newborn and Infant Nursing Reviews* (vol. 10, n.º 1, págs. 37-43).

Estos valores pueden aparecer por diferentes razones, por lo que se aplican distintas soluciones en función del contexto. En algunos casos, estos datos serán legítimos y formarán parte de la muestra, por lo que no se deberá modificar el conjunto de datos y se contemplarán los *outliers* en el análisis.

En otros casos, se puede detectar una desviación sistemática en el grupo de valores extremos que se solucionará con una operación matemática sencilla. Por ejemplo, al integrar los conjuntos de datos procedentes de las diferentes tiendas de una cadena, nos podemos encontrar que, aunque la mayoría de las tiendas reportan el precio de sus productos en euros, la tienda que se encuentra en el aeropuerto de Ciudad de México ha reportado los precios en pesos mexicanos. Aunque estos precios se verán como *outliers* inicialmente, solo será necesario aplicar la conversión de pesos a euros en ese grupo de datos para que el conjunto sea coherente.

En los casos en los que los *outliers* sean errores en los datos complicados de corregir, generalmente se tratarán como valores perdidos, por lo que se optará por eliminar o corregir el registro mediante los métodos de imputación de datos mencionados en el apartado anterior (apartado 1.5).

Aunque la decisión sobre qué se considera un valor extremo ha sido un tema controvertido durante décadas, generalmente se considera que cuando un valor se encuentra alejado 3 desviaciones estándar con respecto a la media del conjunto es un *outlier*. Por ello, en muchos trabajos se utiliza la representación de los datos mediante gráficos de cajas (*boxplots*), con el objetivo de detectar dichos *outliers*.

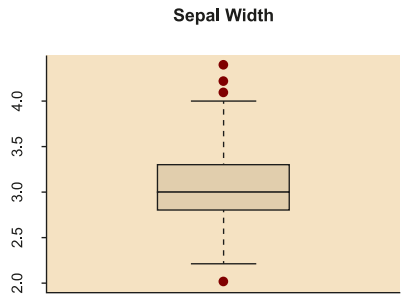
El siguiente ejemplo muestra la detección de valores extremos mediante esta técnica en R. En la gráfica resultante de la función `boxplot()` se identifican 4 *outliers*, representados en forma de círculos y cuyo valor puede recuperarse del resultado `out` (figura 8).

Ejemplo

```
>iris.bp<-boxplot(iris$Sepal.Width,main="Sepal Width")
```

```
>iris.bp$out
[1] 4.4 4.1 4.2 2.0
```

Figura 8. *Boxplot* de `Sepal.Width`, del conjunto de datos `iris`, donde se representan 4 *outliers*



Otros métodos que permiten identificar valores extremos en datos multidimensionales se basan en la distancia de Mahalanobis o la distancia de Cook. A partir de estas distancias, los valores extremos pueden identificarse al aplicar un umbral o al especificar un número fijo de *outliers*.

La distancia de Mahalanobis determina la similitud entre variables aleatorias multidimensionales, basándose en la correlación entre dichas variables. Por otro lado, la distancia de Cook estima el grado de influencia de cada uno de los puntos que forman el conjunto cuando se realiza un análisis de regresión por mínimos cuadrados.

En el siguiente ejemplo se busca identificar los valores extremos de un conjunto de datos incluyendo el peso y la altura de 16 individuos. Cuando se seleccionan como *outliers* aquellos puntos que se encuentran, unidimensionalmente, a más de 2 desviaciones estándar de la media, una de las muestras que se identifica como normal (por no presentar un valor de más de 2 desviaciones estándar en ninguno de los atributos analizados) parece ser un *outlier* cuando se analiza visualmente junto al resto de datos (figura 9). Esto se confirma al calcular las distancias de Mahalanobis de cada muestra y seleccionar los 2 primeros candidatos a *outliers*.

Bibliografía recomendada

Newton, Rae R.; Rudestam, Kjell E. (1999). *Your Statistical Consultant: Answers to Your Data Analysis Questions*. Thousand Oaks, CA: Sage Publications.

Mahalanobis, Prasanta C. (1936, enero). «On the generalized distance in statistics». *Proceedings of the National Institute of Science of India* (vol. II, n.º 1).

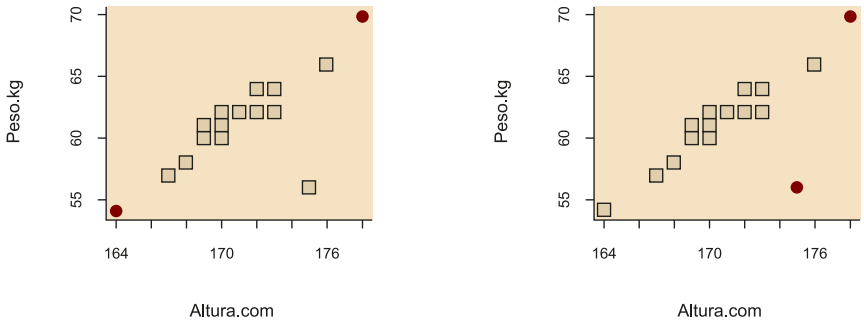
Cook, R. Dennis. (1977, febrero). «Detection of Influential Observations in Linear Regression». *Technometrics*, American Statistical Association (vol. 19, n.º 1, págs. 15-18).

Ejemplo

```
>ap <- data.frame(Altura.cm=c(164, 167, 168, 169, 169, 170, 170, 170, 171,
172, 172, 173, 173, 175, 176, 178), Peso.kg=c( 54,  57,  58,  60,  61,  60,
61,  62,  62,  64,  62,  62,  64,  56,  66,  70))

>#Criterio +/-2SD
>Altura.outlier <- abs(scale(ap$Altura.cm)) > 2
>Peso.outlier <- abs(scale(ap$Peso.kg)) > 2
>pch <- (Altura.outlier | Peso.outlier) * 16

>par(mfrow=c(1,2))
>plot(ap, pch=pch)

>#Criterio distancia Mahalanobis (los dos outliers más extremos)
>n.outliers   <- 2
>m.dist.order <- order(mahalanobis(ap, colMeans(ap), cov(ap)), decreasing=TRUE)
>is.outlier   <- rep(FALSE, nrow(ap))
>is.outlier[m.dist.order[1:n.outliers]] <- TRUE
>pch <- is.outlier * 16
```

```
>plot(ap, pch=pch)
```

Figura 9. Representación del conjunto de datos `ap` y *outliers* detectados (círculos rojos) según diferentes criterios. Izquierda: dos desviaciones estándar. Derecha: distancia de Mahalanobis



Finalmente, existen otros métodos más sofisticados para la detección de *outliers* que se basan en modelos estadísticos, supervisados o no supervisados, del conjunto de datos para tratar de detectar las anomalías o errores. Por ejemplo, mediante técnicas de *clustering* es posible identificar conjuntos de datos que se alejen significativamente de los valores esperados de la muestra.

Bibliografía recomendada

Han, Jiawei; Kamber, Micheline; Pei, Jian (2011). *Data Mining: Concepts and Techniques*. Waltham: Elsevier.

2. Análisis de datos

El **análisis** o **exploración de los datos** tiene como objetivo explicar las principales características de los mismos, para así tratar de responder a las preguntas planteadas en el marco de un proyecto de datos.

Dependiendo de la naturaleza de estos datos, así como de los objetivos del proyecto, se pueden aplicar diferentes tipos de análisis. Los siguientes apartados describen algunos de los más utilizados.

2.1. Análisis estadístico descriptivo

Las **estadísticas descriptivas** son estimaciones, valores calculados a partir de una muestra de datos, que describen o resumen las características intrínsecas de dicha muestra. La media, la mediana o la desviación estándar son estimaciones comúnmente utilizadas.

Bibliografía recomendada

Jarman, Kristin H. (2013). *The art of data analysis: how to answer almost any question using basic statistics*. Hoboken, NJ: Wiley.

En muchas ocasiones, esta etapa se realiza incluso previamente al proceso de limpieza de los datos ya que proporciona una visión general de los datos muy valiosa a la hora de identificar los procesos de limpieza y análisis más adecuados para el tipo de datos a estudiar.

El análisis estadístico descriptivo puede dividirse principalmente en dos tipos:

a) Las medidas de tendencia central: se incluyen las medidas que representan el centro de la distribución de datos, como la media, la mediana, la moda o el rango medio.

b) Las medidas de dispersión: es habitual calcular el rango, los cuartiles, el rango intercuartílico, la varianza o la desviación estándar.

Algunas funciones interesantes en R que permiten calcular algunas de estas medidas descriptivas son `mean()`, `median()`, `quantiles()`, `var()` o `sd()`. Asimismo, la función `summary()` proporciona un resumen de cada uno de los atributos del conjunto, incluyendo el mínimo, el máximo, la media, la mediana, el primer y tercer cuartiles, así como el número de NA (siglas de *not avalaible*). A continuación, se muestra un ejemplo con los datos de `iris` modificados para contener NA:

Ejemplo

```
>summary(iris.mis)

Sepal.Length    Sepal.Width        Petal.Length    Petal.Width      Species
```

```
 Min.   :4.400   Min.   :2.000   Min.   :1.000   Min.   :0.100   setosa  :45
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300   versicolor:48
 Median :5.800   Median :3.000   Median :4.200   Median :1.300   virginica :47
 Mean   :5.873   Mean   :3.058   Mean   :3.706   Mean   :1.209   NA's    :10
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
 NA's   :19      NA's   :18       NA's   :17    NA's   :11
```

Otra función interesante en R que permite mostrar de forma compacta la estructura de un conjunto de datos es `str()`, ya que muestra el número de observaciones y atributos o variables del conjunto, así como sus tipos. El siguiente ejemplo describe las dimensiones del conjunto de datos `iris.mis`, así como los tipos de variables que contiene (cuatro numéricos y un factor con 3 niveles o categorías).

Ejemplo

```
>str(iris.mis)

'data.frame':    150 obs. of  5 variables:
$ Sepal.Length: num  5.1 4.9 NA 4.6 5 NA 4.6 5 4.4 4.9 ...
$ Sepal.Width : num  NA 3 NA 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
$ Petal.Length: num  1.4 NA 1.3 1.5 NA 1.7 1.4 1.5 1.4 1.5 ...
$ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 NA 0.2 0.1 ...
$ Species  : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 NA 1 1 1 1 1 NA …
```

2.2. Análisis estadístico inferencial

Este tipo de análisis tiene por objetivo modelar los datos a través de una distribución conocida. Partiendo de la premisa que el conjunto de datos estudiado representa una fracción de la totalidad de una población, su objetivo es inferir cómo es esa población, asumiendo un grado de error en las estimaciones por el hecho de disponer de una muestra reducida de los datos.

Los siguientes apartados describen algunos ejemplos de análisis de este tipo, como son la comparación de grupos mediante los contrastes de hipótesis, las regresiones o las correlaciones.

2.2.1. Comparación de uno o dos grupos

En el apartado sobre limpieza de los datos se hacía referencia a la importancia del *data screening* para verificar características importantes en los datos a la hora de identificar los métodos de análisis más adecuados (ver apartado 2.2). Por ejemplo, saber si los datos siguen una distribución normal, así como si presentan homocedasticidad, será fundamental para poder aplicar pruebas por contraste de hipótesis de tipo paramétrico.

Los siguientes apartados presentan varios métodos basados en el análisis estadístico de los datos, diseñados para comprobar la normalidad y la homocedasticidad, así como para comparar pares de grupos de datos.

Comprobación de la normalidad

Con el objetivo de verificar la suposición de la normalidad, algunas de las pruebas más habituales son los tests de Kolmogorov-Smirnov y de Shapiro-Wilk. Aunque ambos comparan la distribución de los datos con una distribución normal, el test de Shapiro-Wilk se considera uno de los métodos más potentes para contrastar la normalidad. Asumiendo como hipótesis nula que la población está distribuida normalmente, si el p-valor es menor al nivel de significancia, generalmente $\alpha = 0{,}05$, entonces la hipótesis nula es rechazada y se concluye que los datos no cuentan con una distribución normal. Si, por el contrario, el p-valor es mayor a α, se concluye que no se puede rechazar dicha hipótesis y se asume que los datos siguen una distribución normal.

El siguiente fragmento de código en R muestra cómo se pueden aplicar estas pruebas, mediante las funciones `ks.test()` y `shapiro.test()`, respectivamente.

Ejemplo

```
>ks.test(iris$Sepal.Length, pnorm, mean(iris$Sepal.Length), sd(iris$Sepal.Length))

 One-sample Kolmogorov-Smirnov test

data:  iris$Sepal.Length
D = 0.088654, p-value = 0.1891
alternative hypothesis: two-sided

>shapiro.test(iris$Sepal.Length)

 Shapiro-Wilk normality test

data:  iris$Sepal.Length
W = 0.97609, p-value = 0.01018
```

En el ejemplo se presenta un caso controvertido en el que se obtienen resultados diferentes para cada una de las pruebas. Mientras que según Kolmogorov-Smirnov los datos siguen una distribución normal, el test de Shapiro-Wilk rechaza la hipótesis nula y considera que no es así.

El **teorema central del límite** se aplica a la distribución de la media de la muestra de un conjunto de datos. La media de una muestra de cualquier conjunto de datos es cada vez más normal a medida que aumenta la cantidad de observaciones. Así, a medida que aumenta el tamaño de la muestra N, la distribución de la media de la muestra se parece cada vez más a una distribución normal con una (verdadera) media de la población μ y varianza σ^2/N.

Bibliografía recomendada

Jarman, Kristin H. (2013). *The art of data analysis: how to answer almost any question using basic statistics*. Hoboken, NJ: Wiley.

Dado que la prueba de Shapiro-Wilk se considera más robusta, una posición más conservadora concluiría que los datos no siguen una distribución normal. No obstante, si el conjunto de datos se compone de un número de registros suficientemente grande, por el teorema central del límite, se podría considerar que los datos siguen una distribución normal.

Comprobación de la homocedasticidad

Asimismo, algunas pruebas estadísticas requieren la comprobación previa de la homocedasticidad en los datos, es decir, de la igualdad de varianzas entre los grupos que se van a comparar. Entre las pruebas más habituales se encuentra el test de Levene, que se aplica cuando los datos siguen una distribución normal, así como el test de Fligner-Killeen, que se trata de la alternativa no paramétrica, utilizada cuando los datos no cumplen con la condición de normalidad. En ambas pruebas, la hipótesis nula asume igualdad de varianzas en los diferentes grupos de datos, por lo que p-valores inferiores al nivel de significancia indicarán heterocedasticidad.

El siguiente código en R muestra un ejemplo del funcionamiento de estas pruebas en la base de datos `InsectSprays`, mediante las funciones `leveneTest()` del paquete `car` y `fligner.test()`, respectivamente.

Ejemplo

```
> leveneTest(count ~ spray, data = InsectSprays)
Levene's Test for Homogeneity of Variance (center = median)
      Df F value   Pr(>F)
group  5  3.8214 0.004223 **
      66
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> fligner.test(count ~ spray, data = InsectSprays)

    Fligner-Killeen test of homogeneity of variances

data:  count by spray
Fligner-Killeen:med chi-squared = 14.483, df = 5, p-value = 0.01282
```

Dado que ambas pruebas resultan en un p-valor inferior al nivel de significancia (< 0,05), se rechaza la hipótesis nula de homocedasticidad y se concluye que la variable `count` presenta varianzas estadísticamente diferentes para los diferentes grupos de `spray`.

Comparación entre dos grupos de datos

Cuando la normalidad y la homocedasticidad se cumplan (p-valores mayores al nivel de significancia), se podrán aplicar pruebas por contraste de hipótesis de tipo paramétrico, como la prueba t de Student. En los casos en los que no se cumplan, se deberán aplicar pruebas no paramétricas como Wilcoxon (cuando se comparen datos dependientes) o Mann-Whitney (cuando los grupos de datos sean independientes).

En la prueba t de Student, la hipótesis nula asume que las medias de los grupos de datos son las mismas, mientras que en las pruebas no paramétricas se asume que las distribuciones de los grupos de datos son las mismas. Por lo tanto,

solo si el p-valor resultante de la prueba es menor al nivel de significancia se rechazará la hipótesis nula y se concluirá que existen diferencias estadísticamente significativas entre los grupos de datos analizados.

En R, la prueba t de Student se aplica mediante la función `t.test()`. El siguiente código comprueba la normalidad y homocedasticidad de los datos `sleep` en R, para aplicar posteriormente la prueba paramétrica t de Student.

Ejemplo

```
>shapiro.test(sleep$extra)
 Shapiro-Wilk normality test

data:  sleep$extra
W = 0.94607, p-value = 0.3114

>leveneTest(extra ~ group, data = sleep)
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group  1  0.2482 0.6244
      18

>t.test(extra ~ group, data = sleep)
 Welch Two Sample t-test

data:  extra by group
t = -1.8608, df = 17.776, p-value = 0.07939
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -3.3654832  0.2054832
sample estimates:
mean in group 1 mean in group 2
           0.75            2.33
```

El p-valor resultante de la prueba t de Student es mayor al nivel de significancia, esto significa que no se observan diferencias estadísticamente significativas entre los grupos de datos de `sleep` para la variable `extra`.

Por otro lado, las pruebas de Wilcoxon y Mann-Whitney se aplican indistintamente mediante la función `wilcox.test()`. El siguiente ejemplo compara las distribuciones de `airquality` mediante esta prueba, tras comprobar que no cumple las suposiciones requeridas por los test paramétricos.

Ejemplo

```
> shapiro.test(airquality$Ozone)

    Shapiro-Wilk normality test

data:  airquality$Ozone
W = 0.87867, p-value = 2.79e-08

> fligner.test(Ozone ~ Month, data = airquality)

    Fligner-Killeen test of homogeneity of variances

data:  Ozone by Month
Fligner-Killeen:med chi-squared = 19.341, df = 4, p-value = 0.0006736

> wilcox.test(Ozone ~ Month, data = airquality, subset = Month %in% c(5, 8))
```

```
    Wilcoxon rank sum test with continuity correction

data:  Ozone by Month
W = 127.5, p-value = 0.0001208
alternative hypothesis: true location shift is not equal to 0
```

En este caso, sí se observan diferencias estadísticamente significativas en la calidad del aire en términos del ozono (`airquality$Ozone`), entre los meses de mayo y agosto.

Por último, en ocasiones se querrá comparar si existen diferencias significativas en una variable categórica entre los grupos definidos por otra variable categórica. En ese caso, se puede aplicar el test de χ^2 en R, mediante la función `chisq.test()`, como muestra el siguiente ejemplo. A partir de las frecuencias de cada sabor de helado para cada uno de los grupos, se observa que hombres y mujeres muestran diferencias significativas en sus gustos.

Ejemplo

```
> men = c(100, 120, 60)
> women = c(350, 200, 90)
> ice.cream.survey = as.data.frame(rbind(men, women))
> names(ice.cream.survey) = c('chocolate', 'vanilla', 'strawberry')
> ice.cream.survey

   chocolate vanilla strawberry
men      100     120          60
women    350     200          90
> chisq.test(ice.cream.survey)

    Pearson's Chi-squared test

data:  ice.cream.survey
X-squared = 28.362, df = 2, p-value = 6.938e-07
```

2.2.2. Comparación entre más de dos grupos

El análisis de varianza unidireccional, también conocido como ANOVA (en inglés, *analysis of variance*) de un solo factor, es una extensión de la prueba t de Student, con el objetivo de comparar las medias entre más de dos grupos de datos.

El siguiente ejemplo muestra el resultado de aplicar un test de ANOVA en R mediante la función `aov()`, al comparar el `Sepal.Width` del conjunto de datos `iris`, entre las diferentes especies de flores (`Species`). La función `summary.aov()` resume dicho resultado, permitiendo concluir que las diferentes especies muestran anchuras del sépalo estadísticamente diferentes.

Ejemplo

```
> shapiro.test(iris$Sepal.Width)

    Shapiro-Wilk normality test
```

```
data:  iris$Sepal.Width
W = 0.98492, p-value = 0.1012

> leveneTest(Sepal.Width ~ Species, data = iris)
Levene's Test for Homogeneity of Variance (center = median)
      Df F value Pr(>F)
group   2  0.5902 0.5555
      147

> res.aov <- aov(Sepal.Width ~ Species, data = iris)
> summary(res.aov)
             Df Sum Sq Mean Sq F value Pr(>F)
Species       2  11.35   5.672   49.16 <2e-16 ***
Residuals   147  16.96   0.115
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Por otro lado, la alternativa no paramétrica a los contrastes de hipótesis de más de 2 grupos, es el test de Kruskal-Wallis. En R, este se aplica mediante la función `kruskal.test()`, como muestra el siguiente ejemplo.

Ejemplo

```
> shapiro.test(airquality$Ozone)

    Shapiro-Wilk normality test

data:  airquality$Ozone
W = 0.87867, p-value = 2.79e-08

> fligner.test(Ozone ~ Month, data = airquality)

    Fligner-Killeen test of homogeneity of variances

data:  Ozone by Month
Fligner-Killeen:med chi-squared = 19.341, df = 4, p-value = 0.0006736

> kruskal.test(Ozone ~ Month, data = airquality)

    Kruskal-Wallis rank sum test

data:  Ozone by Month
Kruskal-Wallis chi-squared = 29.267, df = 4, p-value = 6.901e-06
```

Dado que el p-valor obtenido es menor al nivel de significancia, se puede concluir que el nivel de ozono muestra diferencias significativas para los diferentes meses del año.

2.2.3. Regresión

La **regresión lineal** es un modelo matemático que tiene como objetivo aproximar la relación de dependencia lineal entre una variable dependiente y una (o una serie) de variables independientes.

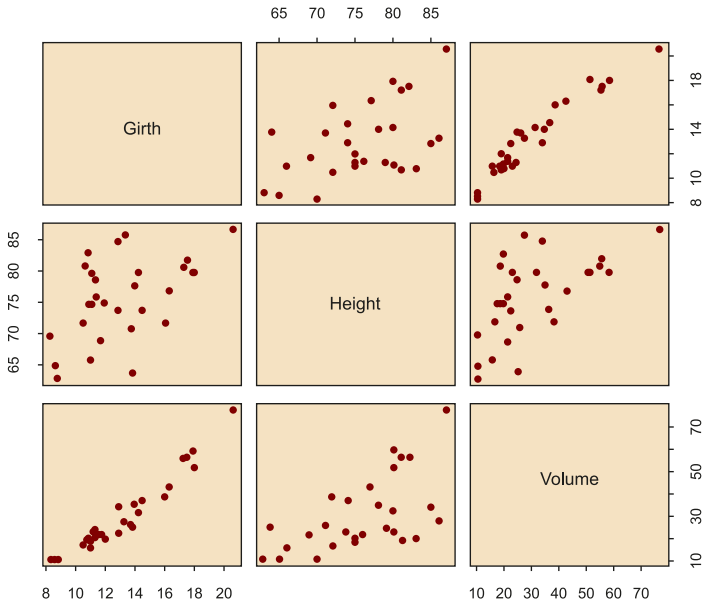
En R, la regresión lineal se aplica mediante la función `lm()`. Esta puede ser simple o múltiple en función de las variables independientes que se incluyan en la fórmula que se introduce como argumento.

El siguiente código en R muestra un ejemplo de cada tipo, para el conjunto de datos `trees`. En primer lugar, se estima un modelo simple del volumen, a partir del perímetro (`Girth`) tras intuir visualmente (figura 10) cierta relación lineal entre estas dos variables. Posteriormente, se implementa un modelo múltiple del volumen, a partir del perímetro y la altura de los árboles (`Girth` y `Height`). Gracias a la función `summary()` se analizan en detalle los resultados de cada uno de los modelos.

Ejemplo

```
> plot(trees)
```

Figura 10. Representación de cada par de variables del conjunto de datos `trees`




```
> m1 = lm(Volume~Girth,data=trees)
> summary(m1)

Call:
lm(formula = Volume ~ Girth, data = trees)

Residuals:
   Min      1Q  Median      3Q     Max
-8.065  -3.107   0.152   3.495   9.587

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -36.9435     3.3651  -10.98 7.62e-12 ***
Girth         5.0659     0.2474   20.48  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.252 on 29 degrees of freedom
Multiple R-squared:  0.9353,    Adjusted R-squared:  0.9331
F-statistic: 419.4 on 1 and 29 DF,  p-value: < 2.2e-16


> m2 = lm(Volume~Girth+Height,data=trees)
> summary(m2)
```

```
Call:
lm(formula = Volume ~ Girth + Height, data = trees)

Residuals:
     Min      1Q  Median      3Q     Max
-6.4065 -2.6493 -0.2876  2.2003  8.4847

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -57.9877     8.6382  -6.713 2.75e-07 ***
Girth         4.7082     0.2643  17.816  < 2e-16 ***
Height        0.3393     0.1302   2.607   0.0145 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.882 on 28 degrees of freedom
Multiple R-squared:  0.948,    Adjusted R-squared:  0.9442
F-statistic:   255 on 2 and 28 DF,  p-value: < 2.2e-16
```

Siendo el coeficiente de determinación (R^2 o *R-squared*) una medida de calidad del modelo que toma valores entre 0 y 1, se comprueba cómo el volumen y el perímetro se correlacionan fuertemente, dando lugar a un *R-squared* de 0,9353. Al introducir la altura, este *R-squared* mejora hasta 0,948 ya que también se correlaciona con el volumen de forma significativa, aunque en menor medida.

Asimismo, la función `lm()` permite implementar modelos polinómicos más complejos, como en el siguiente ejemplo:

Ejemplo

```
> m3 = lm(Volume~Girth+I(Girth^2),data=trees)
> summary(m3)

Call:
lm(formula = Volume ~ Girth + I(Girth^2), data = trees)

Residuals:
    Min      1Q  Median      3Q     Max
-5.4889 -2.4293 -0.3718  2.0764  7.6447

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.78627   11.22282   0.961 0.344728
Girth       -2.09214    1.64734  -1.270 0.214534
I(Girth^2)   0.25454    0.05817   4.376 0.000152 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.335 on 28 degrees of freedom
Multiple R-squared:  0.9616,    Adjusted R-squared:  0.9588
F-statistic: 350.5 on 2 and 28 DF,  p-value: < 2.2e-16
```

Se puede observar que el término que relaciona el volumen con el perímetro de forma cuadrática resulta ser el más significativo, mejorando el *R-squared* hasta 0,9616.

Por último, si una vez estimado el modelo de regresión quisiéramos utilizarlo para predecir el resultado en nuevas muestras de datos, se utilizaría la función `predict()` de la siguiente manera:

Ejemplo

```
> pred.frame<-data.frame(Girth=seq(10,16,2))
> predict(m3,newdata=pred.frame)
    1     2     3     4
15.31863 22.33400 31.38568 42.47365
```

Por otro lado, la **regresión logística** es un tipo de análisis de regresión utilizado para predecir el resultado de una variable dicotómica dependiente, en función de una serie de variables independientes o predictoras. Dado que este modelo estima las probabilidades de ocurrencia, en lugar de utilizar un modelo aditivo que podría predecir valores fuera del rango (0,1) utiliza una escala transformada basada en una función logística.

Así, el modelo lineal para probabilidades transformadas se define como:

$$logit(p) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k \qquad (4)$$

donde $logit(p) = log\left(\frac{p}{1-p}\right)$ y p representa la probabilidad de ocurrencia de una de las categorías.

En R, este tipo de modelos se estiman mediante la función `glm()`, especificando la familia como binomial. El siguiente código muestra un ejemplo de aplicación sobre el conjunto de datos `BreastCancer` del paquete `mlbench`, donde se combinan las variables `Cell.size` y `Cell.shape` para estimar si un tumor es maligno o benigno.

Ejemplo

```
> data(BreastCancer, package="mlbench")
> bc <- BreastCancer[complete.cases(BreastCancer),]
> m4<- glm(Class ~ Cell.size+Cell.shape,data=bc, family="binomial")
> summary(m4)

Call:
glm(formula = Class ~ Cell.size + Cell.shape, family = "binomial",
 data = bc)

Deviance Residuals:
  Min        1Q     Median    3Q    Max
-2.6080  -0.0868  -0.0868   0.0000   3.3416

Coefficients:
              Estimate Std. Error z value   Pr(>|z|)
(Intercept)     9.57037  819.07884   0.012     0.991
Cell.size.L    11.57767  971.41961   0.012     0.990
Cell.size.Q    -6.17166 1905.62757  -0.003     0.997
Cell.size.C     8.20862 1050.06309   0.008     0.994
Cell.size^4    18.04156 1699.37615   0.011     0.992
Cell.size^5     4.55711 1084.68606   0.004     0.997
Cell.size^6    -8.65383 1474.48372  -0.006     0.995
Cell.size^7     1.55134 1538.21302   0.001     0.999
Cell.size^8    10.18306 1226.09621   0.008     0.993
Cell.size^9     0.08582 2675.24882   0.000     1.000
Cell.shape.L   17.78829 2558.23603   0.007     0.994
Cell.shape.Q    8.88783 1476.66754   0.006     0.995
Cell.shape.C    5.64269 1282.10652   0.004     0.996
```

```
Cell.shape^4   -1.99598 2612.89511  -0.001     0.999
Cell.shape^5   -5.73970 3110.33104  -0.002     0.999
Cell.shape^6   -5.75511 2642.37488  -0.002     0.998
Cell.shape^7   -3.90172 1695.00477  -0.002     0.998
Cell.shape^8   -1.51847  805.59705  -0.002     0.998
Cell.shape^9   -0.82841  251.11006  -0.003     0.997

(Dispersion parameter for binomial family taken to be 1)

 Null deviance: 884.35  on 682  degrees of freedom
Residual deviance: 180.13  on 664  degrees of freedom
AIC: 218.13

Number of Fisher Scoring iterations: 19
```

En este caso, la bondad del modelo se evaluará mediante la medida AIC (criterio de información de Akaike, por sus siglas en inglés *Akaike Information Criterion*). Dado que esta medida tiene en cuenta tanto la bondad del ajuste (el error) como la complejidad del modelo, cuando se comparen varios modelos candidatos se seleccionará aquel que resulte en el menor AIC.

2.2.4. Correlación

El **coeficiente de correlación** es una medida de la asociación entre dos variables. Este puede tomar valores entre -1 y 1, donde los extremos indican una correlación perfecta y el 0 indica la ausencia de correlación. El signo es negativo cuando valores elevados de una variable se asocian con valores pequeños de la otra, y el signo es positivo cuando ambas variables tienden a incrementar o disminuir simultáneamente.

El coeficiente de correlación de **Pearson** es el más utilizado entre variables relacionadas linealmente. No obstante, para poder aplicarse, requiere que la distribución de ambas variables sea normal, así como que se cumpla el criterio de homocedasticidad.

Así, la correlación de **Spearman** aparece como una alternativa no paramétrica que mide el grado de dependencia entre dos variables. Este método no conlleva ninguna suposición sobre la distribución de los datos, aunque las variables a comparar deben medirse al menos en una escala ordinal.

En R, la función `cor()` permite calcular la correlación entre las variables que componen un conjunto de datos. Por ejemplo:

Ejemplo

```
> cor(trees)

           Girth Height   Volume
Girth  1.0000000 0.5192801 0.9671194
Height 0.5192801 1.0000000 0.5982497
Volume 0.9671194 0.5982497 1.0000000
```

Sin embargo, el resultado anterior no da ninguna indicación sobre si la correlación es significativamente diferente de cero. Para ello, es necesario utilizar la función `cor.test()` y especificar las variables a comparar.

Ejemplo

```
> cor.test(trees$Volume,trees$Height)

    Pearson's product-moment correlation

data:  trees$Volume and trees$Height
t = 4.0205, df = 29, p-value = 0.0003784
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.3095235 0.7859756
sample estimates:
   cor
0.5982497
```

Como se observa del resultado anterior, `cor.test()` analiza por defecto la correlación de Pearson, aunque también permite analizar otras correlaciones como la de Spearman, la cual será más indicada cuando los datos no sigan una distribución normal.

Ejemplo

```
> cor.test(trees$Volume,trees$Height, method="spearman")

    Spearman's rank correlation rho

data:  trees$Volume and trees$Height
S = 2089.6, p-value = 0.0006484
alternative hypothesis: true rho is not equal to 0
sample estimates:
     rho
0.5787101
```

En ambos casos el p-valor es significativo y el coeficiente de correlación es mayor a 0,57. No obstante, podemos comprobar como la condición de normalidad no se cumple para la variable `Volume`, por lo que el test más adecuado en este caso será el de Spearman. Se observa, por tanto, una correlación de 0,579 entre `Volume` y `Height`. Sería erróneo afirmar que esta correlación es de 0,598 ya que nos estaríamos basando en el resultado del test de Pearson, que supone normalidad en los datos.

2.3. Análisis de supervivencia

El **análisis de la supervivencia** trata generalmente con datos que no se encuentran distribuidos normalmente. Asimismo, estos datos son a menudo censurados, es decir, no se conoce su supervivencia exacta ya que va más allá del periodo de estudio. Esta técnica es ampliamente utilizada en campos como la biología y la medicina, pero también en ingeniería, para el estudio de la fiabilidad de ciertas aplicaciones (Dalgaard, 2008).

El **estimador de Kaplan-Meier** es uno de los métodos más utilizados ya que se trata de un estimador no paramétrico de la función de supervivencia que tiene en cuenta la censura.

En R, el paquete `survival` permite realizar este tipo de análisis. Mediante la función `Surv()` se pueden indicar los datos censurados para posteriormente implementar un estimador de Kaplan-Meier mediante la función `survfit()`.

El siguiente ejemplo utiliza el conjunto de datos `melanoma`, del paquete `ISwR`. En este, la variable `status` es un indicador del estado del paciente al final del estudio:

1) muerto por melanoma maligno;

2) vivo el 1 de enero de 1978; y

3) muerto por otras causas.

Además, `days` es el tiempo de observación en días, `ulc` indica si el tumor fue ulcerado, `thick` es el grosor del tumor y `sex` contiene la información sobre el sexo del paciente (1 para las mujeres y 2 para los hombres). Dado que los estados 2 y 3 se consideran censurados, la función de supervivencia se modela a partir de esta información (los valores censurados se representan con el signo +).

Ejemplo

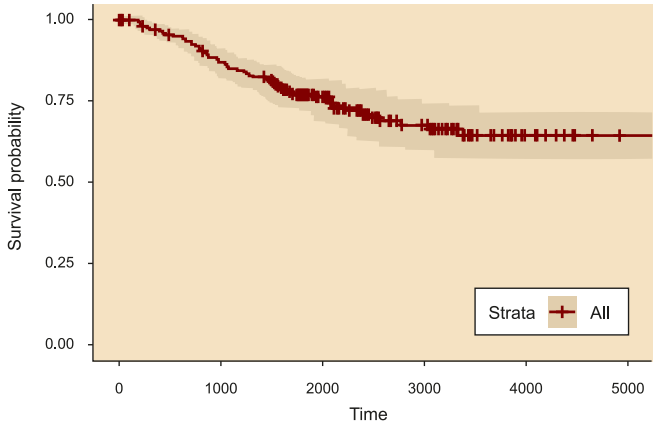
```
> data(melanom)
> attach(melanom)

> Surv(days, status==1)
[1]   10+   30+   35+   99+  185   204   210   232   232+  279   295   355+  386   426   469
493+  529   621   629
[20]  659   667   718   752   779   793   817   826+  833   858   869   872   967   977   982
1041  1055  1062  1075
[39] 1156  1228  1252  1271  1312  1427+ 1435  1499+ 1506  1508+ 1510+ 1512+ 1516  1525+ 1542+
1548  1557+ 1560  1563+
[58] 1584  1605+ 1621  1627+ 1634+ 1641+ 1641+ 1648+ 1652+ 1654+ 1654+ 1667  1678+ 1685+ 1690
1710+ 1710+ 1726  1745+
[77] 1762+ 1779+ 1787+ 1787+ 1793+ 1804+ 1812+ 1836+ 1839+ 1839+ 1854+ 1856+ 1860+ 1864+ 1899+
1914+ 1919+ 1920+ 1927+
[96] 1933  1942+ 1955+ 1956+ 1958+ 1963+ 1970+ 2005+ 2007+ 2011+ 2024+ 2028+ 2038+ 2056+ 2059+
2061  2062  2075+ 2085+
[115] 2102+ 2103  2104+ 2108  2112+ 2150+ 2156+ 2165+ 2209+ 2227+ 2227+ 2256  2264+ 2339+ 2361+
2387+ 2388  2403+ 2426+
[134] 2426+ 2431+ 2460+ 2467  2492+ 2493+ 2521+ 2542+ 2559+ 2565  2570+ 2660+ 2666+ 2676+ 2738+
2782  2787+ 2984+ 3032+
[153] 3040+ 3042  3067+ 3079+ 3101+ 3144+ 3152+ 3154+ 3180+ 3182+ 3185+ 3199+ 3228+ 3229+ 3278+
3297+ 3328+ 3330+ 3338
[172] 3383+ 3384+ 3385+ 3388+ 3402+ 3441+ 3458+ 3459+ 3459+ 3476+ 3523+ 3667+ 3695+ 3695+ 3776+
3776+ 3830+ 3856+ 3872+
[191] 3909+ 3968+ 4001+ 4103+ 4119+ 4124+ 4207+ 4310+ 4390+ 4479+ 4492+ 4668+ 4688+ 4926+ 5565+
```

Además, el estimador de Kaplan-Meier se puede aplicar para el conjunto de datos completo o dividiendo el análisis en grupos de datos. El siguiente ejemplo muestra el resultado de aplicar ambas aproximaciones, tomando el conjunto de datos completo y posteriormente separando la estimación para hombres y mujeres. La función `ggsurvplot()` del paquete `survminer` permite graficar el resultado, donde las líneas verticales representan los datos censurados.

Ejemplo

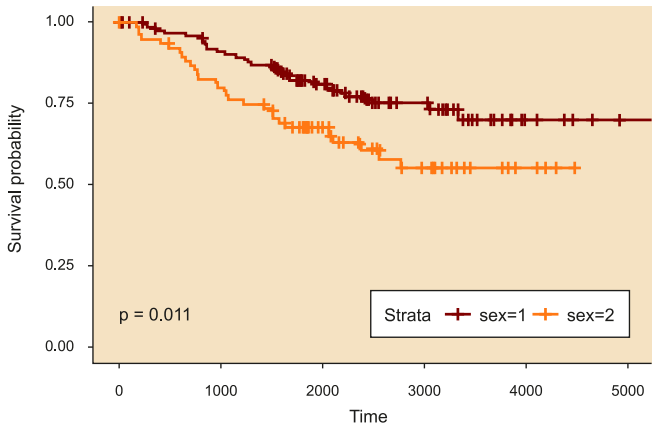
```
> surv.all <- survfit(Surv(days,status==1)~1)
ggsurvplot(surv.all,Surv(days,status==1),pval=TRUE)
```

Figura 11. Kaplan-Meier (con intervalo de confianza) para los datos de `melanoma`




```
> surv.bysex <- survfit(Surv(days,status==1)~sex)
ggsurvplot(surv.bysex,Surv(days,status==1),pval=TRUE)
```

Figura 12. Kaplan-Meier para los datos de `melanoma`, cuando se separa la estimación para hombres (naranja) y mujeres (rojo)



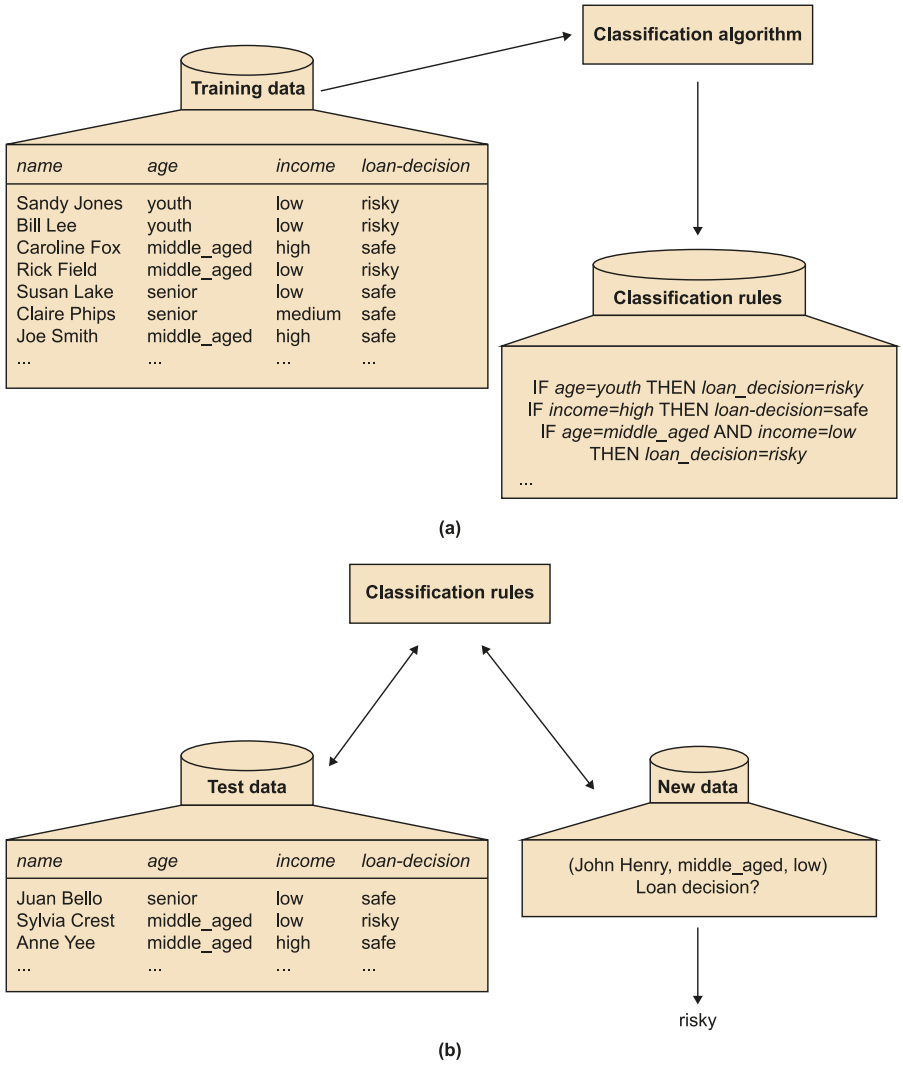
En la figura 12 se observa cómo la supervivencia al melanoma tiende a ser menor en el grupo de hombres (`sex = 2`), siendo esta diferencia estadísticamente significativa (p-valor = 0,011).

2.4. Modelos supervisados

El **aprendizaje supervisado** estima una función o modelo a partir de una serie de datos de entrenamiento, con el objetivo de predecir posteriormente el resultado de nuevos datos desconocidos. Los conjuntos de datos de entrenamiento están formados por pares de objetos que representan los datos de entrada y los resultados deseados. Estos resultados pueden ser un valor numérico, como en los problemas de regresión tratados en el apartado 2.2.3, o una etiqueta de clase, como en los de clasificación, que serán el objeto de este apartado.

En todo problema de **clasificación**, el conjunto de datos se dividirá por tanto en los subconjuntos de entrenamiento (*training*) y de prueba o test (*testing*). Gracias a los primeros, se entrenará un modelo de clasificación de forma que se definan una serie de reglas de clasificación. A continuación, gracias a los datos de test, se estimará la exactitud (*accuracy*) del modelo, de manera que, si esta es aceptable, las reglas de clasificación definidas podrán ser utilizadas en nuevos datos de entrada con las mismas características, con el objetivo de predecir su resultado. La figura 13 representa esquemáticamente este proceso.

Figura 13. Etapas implicadas en los procesos de clasificación: a) etapa de entrenamiento y b) etapa de clasificación (Jiawei *et al.*, 2011)



Los siguientes apartados describen los principales métodos de partición de los datos que permiten definir los grupos de entrenamiento y test, así como los modelos de clasificación y las medidas de análisis del rendimiento de dichos modelos más utilizados.

2.4.1. Partición de los datos

Existen varios métodos para clasificar los datos originales en entrenamiento y test: el método de exclusión (*holdout*), el método de submuestreo aleatorio (*random subsampling*) y la validación cruzada (*cross-validation*).

Bibliografía recomendada

Han, Jiawei; Kamber Micheline; Pei, Jian (2011). *Data Mining: Concepts and Techniques*. Waltham: Elsevier.

En el método de **exclusión**, los datos se dividen aleatoriamente en dos conjuntos independientes, el de entrenamiento y el de test. Típicamente, dos tercios de los datos se asignan al conjunto de entrenamiento, y el tercio restante se reserva para testear el modelo. En general, esta estimación es pesimista ya que solo utiliza una parte de los datos originales para diseñar el modelo.

El método de **submuestreo** aleatorio es una variación del método anterior, ya que aplica la misma técnica *k* veces para posteriormente estimar la precisión global del modelo como el promedio de las precisiones obtenidas de cada iteración.

En el método de la **validación cruzada** de tipo *k-fold* los datos originales se dividen aleatoriamente en *k* subconjuntos (*folds*) mutuamente exclusivos y de tamaños similares. El entrenamiento y testeo se realizan *k* veces, a partir de todas las combinaciones posibles de *k-1* subconjuntos para entrenamiento y dejando el subconjunto restante para testear el modelo. En este caso, a diferencia de los métodos anteriores, cada muestra se utiliza el mismo número de veces para entrenar y solo una vez para testear. La exactitud se calcula como el número total de clasificaciones correctas en las *k* iteraciones, dividido por el número total de muestras en el conjunto de datos original.

El *leave-one-out* es un caso especial de validación cruzada de tipo *k-fold* donde *k* se ajusta al número de muestras del conjunto de datos original. En cada iteración, solo omite una de las muestras en la fase de entrenamiento, para posteriormente utilizarla en el testeo del modelo. La validación cruzada de tipo *k-fold* es una estimación pesimista y sesgada del rendimiento del modelo ya que, generalmente, mejorará cuando se amplíe el conjunto de entrenamiento. Aunque la validación cruzada de tipo *leave-one-out* reduce significativamente este sesgo al utilizar prácticamente la totalidad del conjunto de datos en la etapa de entrenamiento, tiende a presentar una alta varianza (se pueden obtener estimaciones muy diferentes al variar la muestra seleccionada para la etapa de test). Así, como el error del estimador dependerá finalmente tanto del sesgo como de la varianza, el uso de un método u otro dependerá del contexto.

En la validación cruzada estratificada, los *folds* se estratifican de modo que la distribución de clases en cada *fold* sea aproximadamente el mismo que en el conjunto de datos original. Generalmente se recomienda utilizar una validación cruzada estratificada con 10 *folds*, incluso si la potencia de cálculo permite utilizar más *folds*, debido a su sesgo y varianza relativamente bajos.

En R, el método de exclusión o *holdout* se puede aplicar mediante la función `holdout()` del paquete `rminer`. Retomando el ejemplo del cáncer de mama (`BreastCancer`), el siguiente código divide los datos originales en 2/3 para entrenamiento y 1/3 para testeo.

Ejemplo

```
> h<-holdout(bc$Class,ratio=2/3,mode="stratified")
> data_train<-bc[h$tr,]
> data_test<-bc[h$ts,]
> print(table(data_train$Class))

benign malignant
 296    159

> print(table(data_test$Class))
```

```
benign malignant
 148     80
```

Dado que el ejemplo aplica una partición de los datos estratificada, las proporciones entre tumores benignos y malignos se mantienen en ambos conjuntos de datos, dejando 2/3 para el entrenamiento (296 tumores benignos y 159 malignos), y el resto para la etapa de test (148 tumores benignos y 89 malignos).

Otro paquete interesante en R es `caret`, ya que permite utilizar gran cantidad de métodos de entrenamiento y clasificación. La función `trainControl()` permite especificar las características del proceso de entrenamiento y testeo. Los siguientes ejemplos de código muestran algunas posibilidades de esta herramienta. En el primer caso se aplica un *leave-one-out*; el segundo hace referencia a una validación cruzada de tipo *10-fold* y el tercero indica la repetición 10 veces de una validación cruzada de tipo *4-fold*.

Ejemplo

```
> train_control1<- trainControl(method="LOOCV")
> train_control2<- trainControl(method="cv", number=10)
> train_control3<- trainControl(method="repeatedcv", number=4, repeats=10)
```

2.4.2. Medidas del rendimiento

A la hora de evaluar la bondad o el rendimiento del modelo de clasificación, existen varias medidas que permiten interpretar el resultado de aplicar dicho modelo al subconjunto de testeo.

Entendiendo los valores positivos como aquellos registros que pertenecen a la clase de interés (por ejemplo, pacientes que no responden a un tratamiento) y negativos el resto (pacientes respondiendo al tratamiento), se definen los siguientes términos, representados en la matriz de confusión de la figura 14:

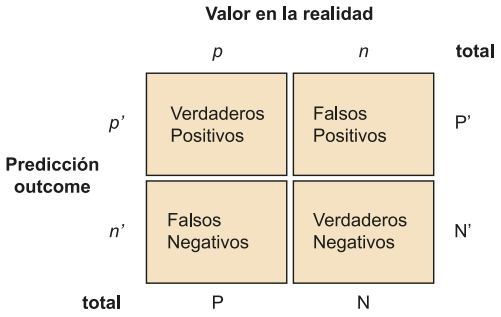
- **Verdaderos positivos (*VP*)**: son los registros positivos correctamente clasificados. Por ejemplo, los pacientes que no responden al tratamiento que han sido clasificados correctamente como «no respondedores».

- **Falsos positivos (*FP*)**: hacen referencia a los registros negativos que fueron incorrectamente clasificados como positivos, es decir, los pacientes que respondieron al tratamiento, pero fueron clasificados como «no respondedores».

- **Verdaderos negativos (*VN*)**: son los registros negativos correctamente clasificados. Son los pacientes que respondieron al tratamiento y fueron clasificados como «respondedores».

- **Falsos negativos (*FN*)**: registros positivos clasificados como negativos. En el ejemplo, aquellos pacientes que no responden al tratamiento, pero son clasificados como «respondedores».

Figura 14. Matriz de confusión



Fuente: *Wikipedia*.

La **exactitud** (*accuracy*) de un clasificador hace referencia al conjunto de registros correctamente clasificados y se calcula como:

$$Exactitud = \frac{VP + VN}{P + N} \qquad (5)$$

No obstante, esta medida no informa de la capacidad del modelo para clasificar los registros positivos y negativos por separado. Para ello, la **sensibilidad** (o tasa de verdaderos positivos) cuantifica la proporción de registros positivos correctamente identificados, mientras que la **especificidad** (o tasa de verdaderos negativos) mide la proporción de registros negativos que se identifican correctamente. Estas medidas se definen como:

$$Sensibilidad = \frac{VP}{P} \qquad (6)$$

$$Especificidad = \frac{VN}{N} \qquad (7)$$

Otra medida ampliamente utilizada es la precisión, que expresa la proporción de registros clasificados como positivos que efectivamente lo son, y se calcula:

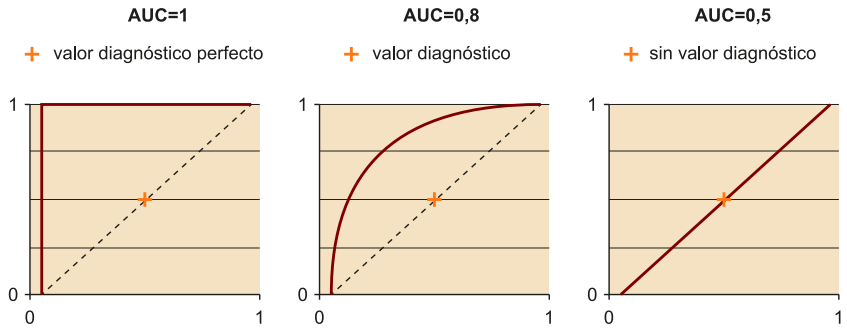
$$Precisión = \frac{VP}{VP + FP} \qquad (8)$$

Las curvas ROC (*receiver operating characteristic*, por sus siglas en inglés) son otra herramienta visual para comparar modelos de clasificación. En R, el paquete `pROC` permite visualizar y trabajar con estas curvas. Las ROC muestran el compromiso entre la tasa de verdaderos positivos (TVP), equivalente a la sensibilidad, y la tasa de falsos positivos (TFP), equivalente a 1-Especificidad. Dado un modelo y un subconjunto de datos para testeo, la TVP es la propor-

ción de registros positivos que han sido correctamente etiquetados por el modelo, mientras que la TFP es la proporción de registros negativos que han sido erróneamente etiquetados como positivos.

A partir de la representación de la curva ROC, es habitual medir su área o AUC (*area under the curve*, por sus siglas en inglés). Este parámetro proporciona información sobre la calidad del modelo, siendo menos preciso a medida que el AUC se acerca a 0,5 y mostrando una exactitud perfecta cuando es 1. La figura 15 muestra tres ejemplos de curvas ROC junto con el valor de su AUC.

Figura 15. AUC obtenido para diferentes curvas ROC



Fuente: *Wikipedia*.

2.4.3. Métodos de clasificación

Existen gran cantidad de métodos de clasificación, diseñados para adaptarse a diferentes tipos de datos. Por ello, dependerá del contexto la idoneidad de cada método para un problema en particular. Algunos ejemplos son los árboles de decisión, los modelos bayesianos o los basados en reglas (*rule-based models*). Asimismo, técnicas más sofisticadas de clasificación incluyen las máquinas de soporte vectorial, los *random forests* o las redes neuronales. En (Jiawei *et al.*, 2011) se puede encontrar más información sobre cada uno de estos métodos.

Aunque muchos de estos métodos cuentan con su propio paquete en R, `caret` incluye más de 200 posibilidades, especificadas en este enlace:

Enlace de interés

Modelos de clasificación disponibles en el paquete `caret` de R: http://topepo.github.io/caret/available-models.html

El siguiente fragmento de código muestra un ejemplo de aplicación de un *random forest* al conjunto de datos sobre cáncer de mama (`BreastCancer`), mediante una validación cruzada con 4 *folds*. Con el objetivo de analizar la bondad del modelo en un conjunto de datos completamente desconocido, se dividen los datos en entrenamiento y test para solo aplicar el entrenamiento mediante validación cruzada al primer subconjunto. Posteriormente, mediante la función `predict()` se predice el resultado de los datos del subcon-

junto de test y se representan las diferentes medidas de bondad del modelo, mediante la función `confusionMatrix()`, especificando como positivos los casos de cáncer maligno.

Ejemplo

```
> data(BreastCancer, package="mlbench")
> bc <- BreastCancer[complete.cases(BreastCancer),-1]

> h<-holdout(bc$Class,ratio=2/3,mode="stratified")
> data_train<-bc[h$tr,]
> data_test<-bc[h$ts,]

> train_control<- trainControl(method="cv", number=4)
> mod<-train(Class~., data=data_train, method="rf", trControl = train_control)

> pred <- predict(mod, newdata=data_test)
> confusionMatrix(pred,data_test$Class,positive="malignant")

Confusion Matrix and Statistics

          Reference
Prediction   benign malignant
  benign        142         6
  malignant       6        74

               Accuracy : 0.9474
                 95% CI : (0.9099, 0.9725)
 No Information Rate : 0.6491
 P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.8845
 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9250
            Specificity : 0.9595
         Pos Pred Value : 0.9250
         Neg Pred Value : 0.9595
             Prevalence : 0.3509
         Detection Rate : 0.3246
   Detection Prevalence : 0.3509
   Balanced Accuracy : 0.9422

    'Positive' Class : malignant
```

2.5. Modelos no supervisados

El **aprendizaje no supervisado** consiste en adaptar un modelo a las observaciones dadas ya que, a diferencia del aprendizaje supervisado, no se tiene un conocimiento *a priori* de los datos. Aunque existen diferentes métodos de aprendizaje no supervisado, los más utilizados son los basados en el agrupamiento o *clustering*.

El agrupamiento o *clustering* es el proceso mediante el cual se agrupa un conjunto de datos en múltiples subgrupos o clústeres, de modo que los registros dentro de un mismo clúster tengan una alta similitud y sean muy diferentes a los registros de otros clústeres. Estas diferencias y similitudes se evalúan en base a los valores de los atributos que describen cada registro y a menudo im-

Bibliografía recomendada

Han, Jiawei; Kamber, Micheline; Pei, Jian (2011). *Data Mining: Concepts and Techniques*. Waltham: Elsevier.

plican medidas de distancia. Este tipo de métodos puede asimismo dividirse en las siguientes categorías: métodos de partición, métodos jerárquicos y métodos basados en la densidad.

Los **métodos de partición**, dado un conjunto de datos con *n* objetos o registros, construyen *k* particiones en los datos, donde cada partición representa un clúster que debe contener al menos un objeto. Los métodos *k-means* y *k-medoids* son ejemplos populares de este tipo. En *k-means*, cada objeto pertenece al clúster cuyo valor medio es más cercano. Así, mientras en *k-means* cada clúster queda representado por el valor medio, en *k-medoids* es el objeto más cercano al centro el que representa cada clúster, razón por la que esta segunda aproximación es más robusta ante el ruido y los *outliers*.

En R, el método *k-means* se puede aplicar mediante la función `kmeans()`. El siguiente ejemplo muestra el resultado de aplicar esta función sobre la base de datos `iris`, para *k = 3*. La última tabla compara el resultado esperado con el obtenido por el método aplicado, donde se puede observar que la clase «setosa» se separa adecuadamente, mientras que «versicolor» y «virginica» muestran cierto solapamiento.

Ejemplo

```
> iris.cl<-iris
> iris.cl$Species<-NULL
> kmeans.res<-kmeans(iris.cl,3)
> table(iris$Species,kmeans.res$cluster)

             1  2  3
  setosa      0  0 50
  versicolor 48  2  0
  virginica  14 36  0
```

El método *k-medoids* suele aplicarse mediante el algoritmo PAM (*partitioning around medoids*, por sus siglas en inglés), disponible en el paquete `cluster` a través de la función `pam()`. Asimismo, el paquete `fpc` proporciona la función `pamk()` donde no es necesario fijar un valor para *k*. El siguiente ejemplo muestra el funcionamiento de ambas funciones.

Ejemplo

```
> kmedoids.res1<-pam(iris.cl,3)
> table(iris$Species,kmedoids.res1$cluster)

             1  2  3
  setosa     50  0  0
  versicolor  0 48  2
  virginica   0 14 36

> kmedoids.res2<-pamk(iris.cl)
> table(iris$Species,kmedoids.res2$pamobject$clustering)

             1  2
  setosa     50  0
  versicolor  1 49
  virginica   0 50
```

Del segundo resultado, se puede confirmar el solapamiento entre las clases «versicolor» y «virginica» ya que, al no fijar el número de clústeres a 3, el algoritmo ha considerado que los datos pueden describirse solo con 2 grupos.

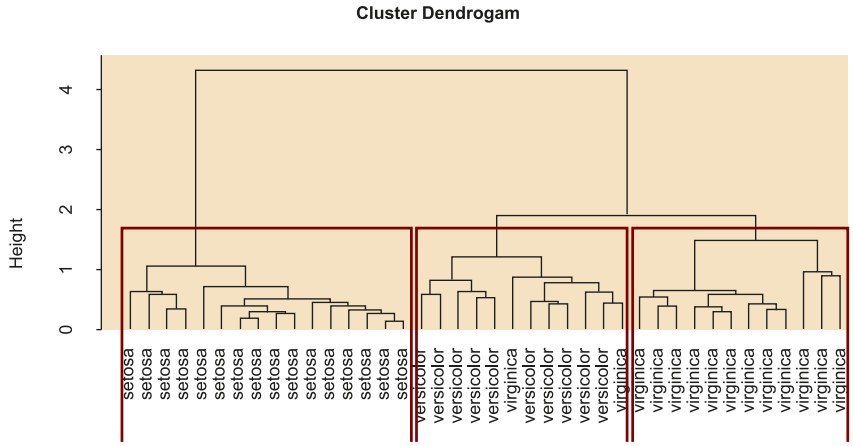
Por otro lado, los **métodos jerárquicos** aplican una descomposición jerárquica del conjunto de datos origen. Estos pueden ser de tipo ascendente (*bottom-up)* o descendente (*top-down*), en función de cómo se produzca dicha descomposición jerárquica. En el primer caso, todos los objetos forman grupos separados al inicio del proceso. Posteriormente, los grupos similares entre sí se van fusionando hasta formar un único clúster (nivel superior de la jerarquía) o hasta llegar a una condición que termine el proceso. Contrariamente, en los métodos de tipo *top-down*, todos los objetos se encuentran en un mismo clúster y, a medida que avanza el proceso, se van dividiendo en clústeres más pequeños, hasta que cada objeto forma un clúster o se llega a una condición de terminación.

En R, este tipo de *clustering* se aplica mediante la función `hclust()`, tal y como se muestra en el ejemplo siguiente, donde se analiza una submuestra del conjunto de datos `iris`, con el objetivo de facilitar la representación e interpretación del resultado (figura 16).

Ejemplo

```
> id<-sample(1:dim(iris.cl)[1],40)
> irisSample<-iris.cl[id,]
> hc<-hclust(dist(irisSample),method="ave")
> hc

Call:
hclust(d = dist(iris.cl), method = "ave")

Cluster method   : average
Distance       : euclidean
Number of objects: 150

> plot(hc, hang = -1, labels = iris$Species[id])
> rect.hclust(hc, k = 3)
```

Figura 16. Resultado del *clustering* jerárquico



Finalmente, los **métodos basados en la densidad** siguen extendiendo un clúster dado mientras la densidad (número de objetos) en el «vecindario» supere algún umbral. Por ejemplo, para cada registro contenido en un clúster dado, la vecindad de un radio dado debe contener al menos un número mínimo de puntos. Dado que aproximaciones como *k-means* suelen definir clústeres con forma esférica y de tamaños similares, estos métodos se utilizan generalmente para filtrar ruido o valores extremos.

En R, pueden aplicarse mediante la función `dbscan()` del paquete `fpc`, especificando los valores de `eps` (tamaño del vecindario) y `MinPts` (número mínimo de puntos).

Ejemplo

```
> ds <- dbscan(iris.cl, eps = 0.42, MinPts = 5)
table(ds$cluster, iris$Species)

      setosa versicolor virginica
  0   2          10         17
  1   48         0          0
  2   0          37         0
  3   0          3         33
```

En el ejemplo, tras fijar un tamaño del vecindario de 0,42 y un mínimo de 5 puntos, se clasifican correctamente 48 plantas como «setosa», 37 como «versicolor» y 33 como «virginica». El resto de plantas fueron clasificadas erróneamente en otros clústeres.

Aunque no se incluye un ejemplo al respecto, como en los métodos de regresión y clasificación, se podría predecir el clúster al que pertenecería un nuevo conjunto de datos mediante la función `predict()`.

3. Visualización de los datos

La **visualización de los datos** tiene como objetivo comunicar la información contenida en los datos de forma clara y eficaz mediante la representación gráfica. Esta etapa aprovecha la capacidad del sistema visual humano para la detección de patrones y tendencias.

Aunque algunos métodos de representación se han ido citando a lo largo del módulo didáctico, este apartado pretende ser un resumen de aquellas técnicas de representación más comunes en las fases de limpieza y análisis de los datos.

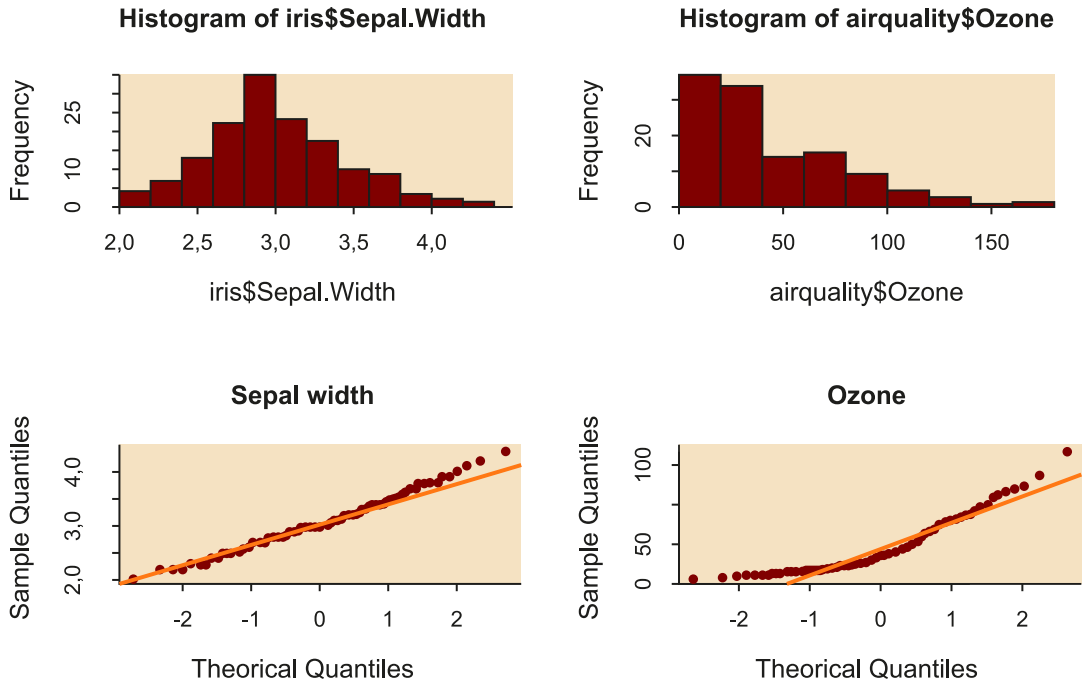
Por un lado, para el análisis de la normalidad, es muy habitual el uso de **histogramas** con el objetivo de observar visualmente si los datos parecen seguir una distribución normal. Otro método ampliamente utilizado en el análisis de la normalidad son los **gráficos Q-Q** o gráficos de cuantiles teóricos. Estos comparan los cuantiles de la distribución observada con los cuantiles teóricos de una distribución normal, por lo que cuanto más se aproximan los datos a una normal, más alineados se muestran sus puntos a la recta.

El siguiente ejemplo muestra el histograma y el gráfico Q-Q de la variable `Sepal.Width` de los datos `iris` y de la variable `Ozone` de los datos `airquality`, mediante el uso de las funciones `hist()`, `qqnorm()` y `qqline()`.

Ejemplo

```
> par(mfrow=c(2,2))
>
> hist(iris$Sepal.Width)
> hist(airquality$Ozone)
>
> qqnorm(iris$Sepal.Width, main="Iris")
> qqline(iris$Sepal.Width,col=2)
> qqnorm(airquality$Ozone, main="AirQuality")
> qqline(airquality$Ozone,col=2)
```

Figura 17. Histograma y gráfico Q-Q para las variables Sepal.Width y Ozone



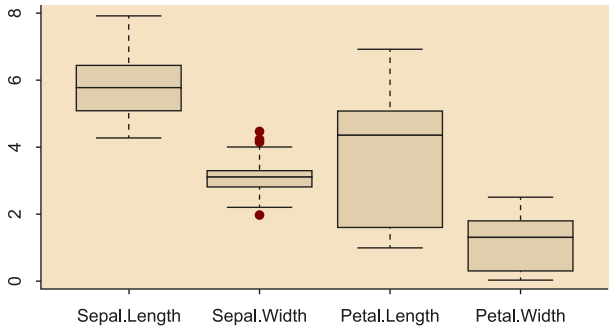
Aunque la normalidad deberá comprobarse mediante pruebas estadísticas, se puede intuir que el ancho del sépalo (`Sepal.Width`) parece seguir una distribución normal, mientras que no será así para el ozono (`Ozone`).

En el análisis de valores extremos, también es habitual el uso de boxplots para identificar *outliers* de forma visual. Aunque la figura 8 ya muestra un ejemplo, a continuación se muestra el *boxplot* para todas las variables del conjunto de datos `iris`, donde solo se identifican valores extremos para la variable `Sepal.Width`, representados con círculos (figura 18).

Ejemplo

```
> boxplot(iris[,-5])
```

Figura 18. *Boxplots* para las variables del conjunto de datos `iris`

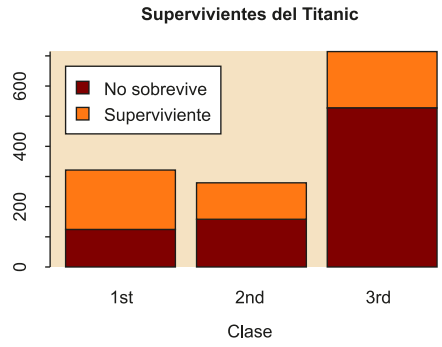


Los **gráficos de barras** (*barplots*) son particularmente útiles cuando se trabaja con datos cualitativos (Jarman, 2013). El siguiente ejemplo representa la cantidad de supervivientes y no supervivientes del Titanic (conjunto de datos `TitanicSurvival`) en cada una de las categorías de `Class`, mediante el uso de la función `barplot()`.

Ejemplo

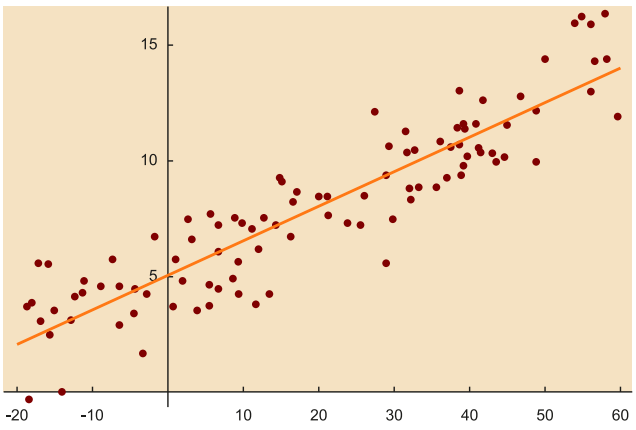
```
> titanic.data<-table(TitanicSurvival[,c(1,4)])
> barplot(titanic.data, main = "Supervivientes del Titanic", xlab = "Clase",col
= c("cadetblue4","aquamarine"))
> legend("topleft", c("No sobrevive","Superviviente"), fill =
c("cadetblue4","aquamarine"))
```

Figura 19. Gráficos de barras para representar la cantidad de supervivientes y no supervivientes en función de la clase, en el conjunto de datos `TitanicSurvival`



Otros gráficos pueden ayudar a interpretar los resultados de diferentes modelos como las regresiones (figura 20), los análisis de supervivencia (ver apartado 2.3), los árboles de decisión o la importancia de las variables en un modelo de clasificación.

Figura 20. Ejemplo de regresión lineal



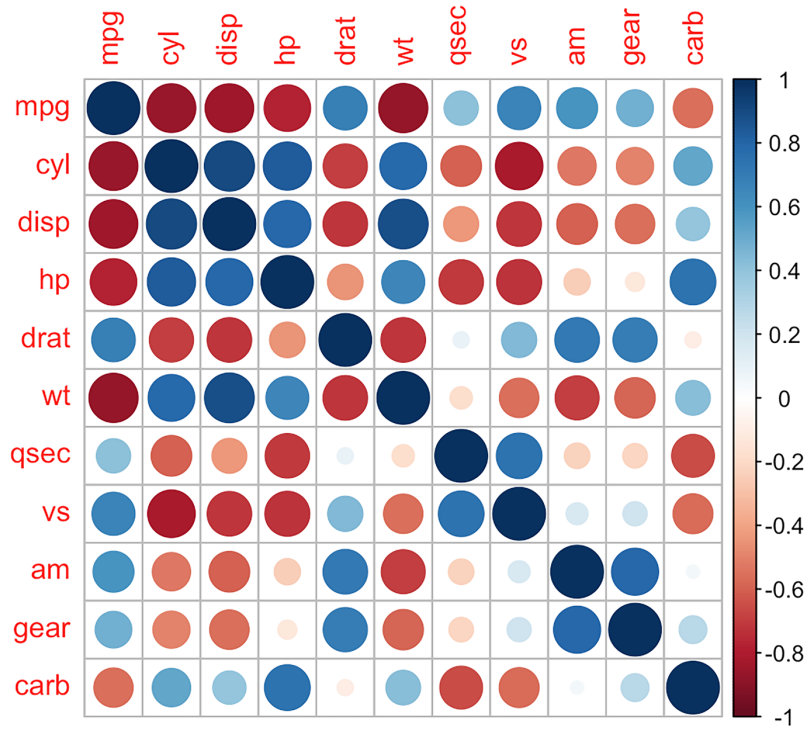
Fuente: *Wikipedia*.

La función `corrplot()`, del paquete `corrplot`, por ejemplo, permite representar gráficamente las correlaciones entre pares de variables en un conjunto de datos. Aunque esta función ofrece gran cantidad de posibilidades para la representación de dichas correlaciones, en el ejemplo se utiliza el método de los círculos, para el conjunto de datos `mtcars`.

Ejemplo

```
> corr.res<-cor(mtcars)
orrplot(corr.res,method="circle")
```

Figura 21. Gráfico de correlaciones para el conjunto de datos `mtcars`



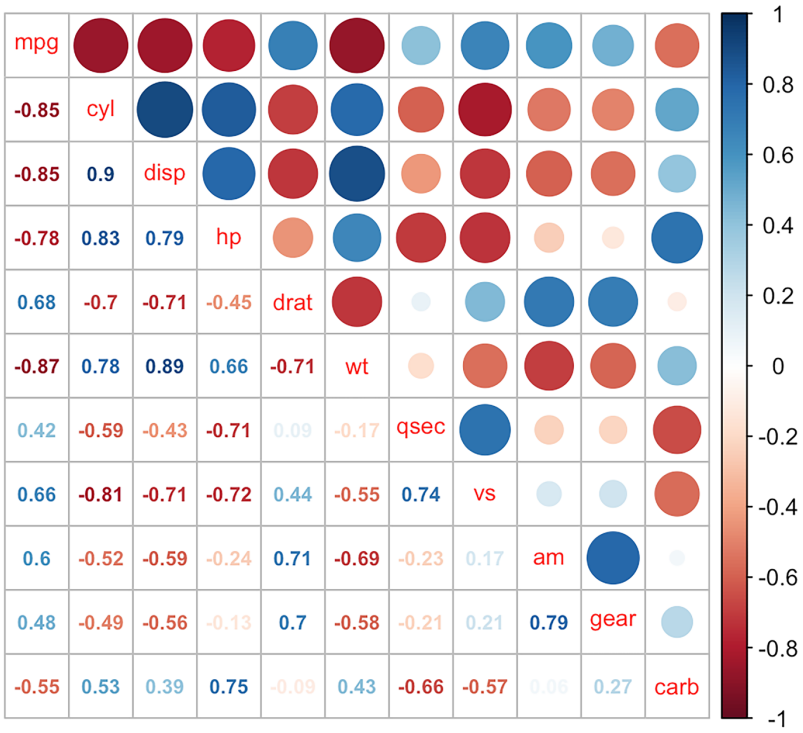
Como indica la barra lateral, el gráfico representa en rojo las correlaciones negativas y en azul las positivas. Asimismo, el tamaño e intensidad de cada uno de los círculos indica el grado de correlación entre los pares de variables.

Por otro lado, la función `corrplot.mixed()` permite mezclar dos métodos de representación, como son los círculos y los valores numéricos. El siguiente ejemplo muestra el resultado para el conjunto de datos `mtcars`.

Ejemplo

```
> corrplot.mixed(corr.res,upper="circle",number.cex=.7,tl.cex=.8)
```

Figura 22. Gráfico mixto de correlaciones para el conjunto de datos `mtcars`



Resumen

En este módulo didáctico se han revisado los aspectos fundamentales relacionados con la limpieza y el análisis de los datos. En primer lugar, se ha presentado la utilidad y potencial de la limpieza de datos o *data cleaning*, particularmente útil cuando se pretenden realizar análisis posteriores con el objetivo de detectar tendencias y patrones en los datos.

Así, en el apartado "Limpieza de datos", se han revisado las principales etapas de limpieza como la integración, la selección y la reducción de los datos, tanto en términos de dimensionalidad como de cantidad, destacando el método ACP como técnica comúnmente utilizada para reducir la dimensionalidad de los datos. Posteriormente, se han presentado algunos métodos de conversión ampliamente utilizados, como los diferentes tipos de normalización para reescalar los datos, las transformaciones de Box-Cox para mejorar su normalidad y homocedasticidad, así como la discretización. Para terminar este apartado, se han introducido varias técnicas de detección y corrección de datos perdidos, así como de valores extremos.

En el apartado "Análisi de datos" se han revisado los principales métodos de análisis de los datos. Se ha revisado la estadística descriptiva e inferencial, así como las pruebas por contraste de hipótesis paramétricas y no paramétricas. Tras presentar las pruebas de Kolmogorov-Smirnov y Shapiro para comprobar la normalidad de los datos, y las pruebas de Levene y Fligner-Killeen para comprobar la homocedasticidad, se han introducido las pruebas t de Student y Wilcoxon para comparar dos grupos de datos y, finalmente, ANOVA y Kruskal-Wallis para las comparaciones entre más de dos grupos. Posteriormente, se han presentado varios modelos de regresión y correlaciones, así como los análisis de supervivencia mediante el estimador de Kaplan-Meier, algunos modelos supervisados como el método de clasificación y modelos no supervisados como los métodos de agrupamiento o *clustering*.

Finalmente, en el apartado "Visualización de los datos" se han resumido algunas funciones útiles para la visualización de los datos, entendida como una etapa adicional y complementaria al análisis de los datos.

Ejercicios de autoevaluación

1. Enumera y describe brevemente las 6 etapas principales de la limpieza de datos.

2. Explica la diferencia entre la reducción de la dimensionalidad y de la cantidad. Pon un ejemplo de cada tipo y describe brevemente en qué consiste.

3. Explica la diferencia entre los conceptos de normalización y normalidad.

4. ¿En qué consiste la regresión logística y en qué casos resulta más adecuada que una regresión lineal?

5. Explica las similitudes y diferencias entre las correlaciones de Pearson y Spearman.

6. ¿Qué son los datos censurados en el análisis de supervivencia? Pon un ejemplo.

7. Enumera y explica brevemente los diferentes tipos de particiones de los datos para entrenamiento y test en clasificación.

8. A partir de los datos `ToothGrowth`, disponibles en R, analiza:

a) Si la variable `len` sigue una distribución normal.

b) Si la varianza de la variable `len` se mantiene constante para los grupos definidos por `supp`.

c) Compara los valores de la variable `len` entre los grupos de datos definidos por `supp` mediante la prueba por contraste de hipótesis más adecuada. Interpreta el resultado.

9. A partir de los datos `iris`, disponibles en R:

a) Analiza si las variables `Sepal.Length, Sepal.Width, Petal.Length` y `Petal.Width` siguen una distribución normal.

b) Calcula la correlación entre pares de variables.

c) Representa gráficamente dichas correlaciones, mostrando los valores numéricos en el triángulo inferior de la figura, así como la representación mediante elipses en el triángulo superior. Interpreta el resultado.

Solucionario

1. Las etapas principales de la limpieza de datos son:

a) Integración: esta etapa consiste en la combinación de datos procedentes de diferentes fuentes, con el fin de crear una estructura coherente y única, que contenga mayor cantidad de información.

b) Selección: se basa en el filtrado de los datos que se encuentran dentro de un fichero o base de datos con el objetivo de seleccionar solo los datos de interés. Para ello, se aplican criterios de búsqueda que nos permitan discernir, en el conjunto de datos, aquellos que son realmente necesarios para el análisis posterior.

c) Reducción: consiste en aplicar ciertos algoritmos de tratamiento con el objetivo de obtener una representación reducida de los datos, manteniendo la integridad de la muestra original. Así, los análisis aplicados sobre la muestra de datos reducida producirán los mismos (o muy similares) resultados que si se aplicaran sobre la muestra total.

d) Conversión: son una serie de técnicas que tienen como objetivo transformar o modificar el formato original de los datos a un formato más plano y entendible, con el objetivo de que el análisis posterior sea más eficiente y/o los resultados obtenidos sean más fácilmente interpretables.

e) Gestión de datos perdidos: representa uno de los problemas más comunes encontrados en bases de datos. Pueden surgir por diferentes causas por lo que, dependiendo de su naturaleza, existen diferentes soluciones para resolver este inconveniente. Por ejemplo, completar manualmente los registros faltantes, reemplazar el conjunto de valores perdidos por una misma constante o etiqueta, reemplazar por una misma medida de tendencia central, o la implementación de métodos probabilistas para predecir los valores perdidos.

f) Análisis de valores extremos: consiste en identificar aquellos datos que se encuentran muy alejados de la distribución normal de una variable o población. Estas observaciones se desvían tanto del resto que levantan sospechas sobre si fueron generadas mediante el mismo mecanismo. Como los datos perdidos, pueden aparecer por diferentes causas, por lo que, dependiendo de su naturaleza, se aplican distintos criterios para evitar que afecten negativamente en el proceso de análisis.

2. Aunque ambos métodos tienen como objetivo reducir la cantidad de datos de un conjunto datos de forma que se mantenga la integridad de la muestra original, la **reducción de la dimensionalidad** permite reducir el número de atributos, mientras que la **reducción de la cantidad** reduce el número de muestras bajo consideración.

Un ejemplo de método para reducir la dimensionalidad es el **análisis de componentes principales** (ACP). Esta técnica permite describir un conjunto de datos de n atributos, en términos de m nuevas variables no correlacionadas, o componentes principales, donde $m < n$. Estas componentes se ordenan según la cantidad de varianza de los datos originales que describen.

Por otro lado, un ejemplo de método para reducir la cantidad de los datos es el *sampling*, ya que permite que un gran conjunto de datos sea representado por un subconjunto de datos mucho más pequeño, seleccionados de forma aleatoria.

3. La **normalización** es un tipo de conversión de los datos que permite reducir el sesgo causado por la combinación de valores medidos a diferentes escalas al ajustarlos a una escala común, típicamente entre (-1,1) o entre (0,1). Dependiendo del contexto, esta normalización se puede aplicar mediante diferentes métodos, siendo la normalización min-max y la normalización z-score los más comunes

La **normalidad** hace referencia a la propiedad de un conjunto de datos de seguir una distribución normal. Es importante destacar que cuando unos datos se normalizan, se está cambiando su escala de representación, pero esto no tiene por qué mejorar su normalidad.

4. La **regresión logística** es un tipo de análisis de regresión utilizado para predecir el resultado de una variable dicotómica dependiente, en función de una serie de variables independientes o predictoras. Dado que este modelo estima las probabilidades de ocurrencia, en lugar de utilizar un modelo aditivo que podría predecir valores fuera del rango (0,1), utiliza una escala transformada basada en una función logística.

Por lo tanto, cuando la variable resultado (dependiente) solo pueda tomar dos valores, la regresión logística será más adecuada que la **regresión lineal**.

5. Ambos coeficientes de correlación son una medida de la asociación entre dos variables. Estos pueden tomar valores entre -1 y 1, donde los extremos indican una correlación perfecta y el 0 indica la ausencia de correlación. Su signo será negativo cuando valores elevados de una variable se asocien con valores pequeños de la otra, y será positivo cuando ambas variables tiendan a incrementar o disminuir simultáneamente.

El coeficiente de **correlación de Pearson** es el más utilizado entre variables relacionadas linealmente. No obstante, para poder aplicarse, requiere que la distribución de ambas variables sea normal, así como que se cumpla el criterio de homocedasticidad.

Por otro lado, la **correlación de Spearman** aparece como una alternativa no paramétrica que mide el grado de dependencia entre dos variables. Este método no conlleva ninguna suposición sobre la distribución de los datos, aunque las variables a comparar deben medirse al menos en una escala ordinal.

6. En el análisis de supervivencia, se dice que los datos de un conjunto están **censurados** cuando no se conoce su supervivencia exacta ya que va más allá del periodo de estudio.

Por ejemplo, en un estudio sobre la supervivencia de una serie de pacientes tras aplicar un tratamiento en particular, si los datos se recogen tan solo un año después de la aplicación de dicho tratamiento, aquellos pacientes que sigan vivos después del año serán datos censurados. Al no recoger datos más allá de este periodo no podremos saber su supervivencia exacta.

7. En el **método de exclusión** (*holdout*), los datos se dividen aleatoriamente en dos conjuntos independientes, el de entrenamiento y el de test. Típicamente, dos tercios de los datos se asignan al conjunto de entrenamiento y el tercio restante se reserva para testear el modelo.

El **submuestreo aleatorio** (*random subsampling*) es una variación del método anterior, ya que aplica la misma técnica *k* veces para posteriormente estimar la precisión global del modelo como el promedio de las precisiones obtenidas de cada iteración.

En la **validación cruzada** (*cross-validation*) de tipo *k-fold* los datos originales se dividen aleatoriamente en *k* subconjuntos (*folds*) mutuamente exclusivos y de tamaños similares. El entrenamiento y testeo se realizan *k* veces, a partir de todas las combinaciones posibles de *k-1* subconjuntos para entrenamiento y dejando el subconjunto restante para testear el modelo. En este caso, la exactitud se calcula como el número total de clasificaciones correctas en las *k* iteraciones, dividido por el número total de muestras en el conjunto de datos original.

El *leave-one-out* es un caso especial de validación cruzada de tipo *k-fold* donde *k* se ajusta al número de muestras del conjunto de datos original. En cada iteración, solo omite una de las muestras en la fase de entrenamiento, para posteriormente utilizarla en el testeo del modelo.

Por último, en la **validación cruzada estratificada**, los *folds* se estratifican de modo que la distribución de clases en cada *fold* sea aproximadamente el mismo que en el conjunto de datos original.

8. a) Se aplica el test de Shapiro y se comprueba que los datos siguen una distribución normal (p-valor > 0,05).

```
> shapiro.test(ToothGrowth$len)

    Shapiro-Wilk normality test

data:  ToothGrowth$len
W = 0.96743, p-value = 0.1091
```

b) Dado que los datos siguen una distribución normal, se aplica el test de Levene y se comprueba que también presentan homocedasticidad (p-valor > 0,05).

```
> leveneTest(ToothGrowth$len ~ ToothGrowth$supp)
```

```
Levene's Test for Homogeneity of Variance (center = median)
       Df F value Pr(>F)
group    1  1.2136 0.2752
       58
```

c) Dado que se cumplen las condiciones de normalidad y homocedasticidad, se aplica la prueba t de Student, comprobando que no existen diferencias significativas entre los grupos de datos (p-valor > 0,05). Por lo tanto, el método de suministro de la vitamina C parece no afectar en el crecimiento dental.

```
> t.test(ToothGrowth$len ~ ToothGrowth$supp)

    Welch Two Sample t-test

data:  ToothGrowth$len by ToothGrowth$supp
t = 1.9153, df = 55.309, p-value = 0.06063
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.1710156  7.5710156
sample estimates:
mean in group OJ mean in group VC
     20.66333      16.96333
```

9. a) Se aplica el test de Shapiro y se comprueba que solo `Sepal.Width` sigue una distribución normal (p-valor > 0,05).

```
> shapiro.test(iris$Sepal.Length)

    Shapiro-Wilk normality test

data:  iris$Sepal.Length
W = 0.97609, p-value = 0.01018

> shapiro.test(iris$Sepal.Width)

    Shapiro-Wilk normality test

data:  iris$Sepal.Width
W = 0.98492, p-value = 0.1012

> shapiro.test(iris$Petal.Length)

    Shapiro-Wilk normality test
```

```
data:  iris$Petal.Length
W = 0.87627, p-value = 7.412e-10


> shapiro.test(iris$Petal.Width)


    Shapiro-Wilk normality test


data:  iris$Petal.Width
W = 0.90183, p-value = 1.68e-08
```

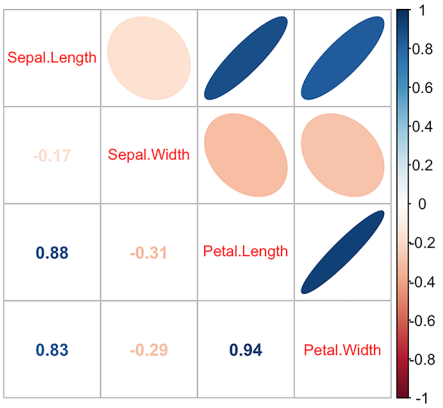
b) Dado que los datos en general no siguen una distribución normal, se aplica el test de Spearman para calcular las correlaciones entre pares de variables.

```
> corr.res<-cor(iris[,-5], method="spearman")
> corr.res
             Sepal.Length  Sepal.Width Petal.Length Petal.Width
Sepal.Length    1.0000000   -0.1667777 0.8818981   0.8342888
Sepal.Width    -0.1667777   1.0000000  -0.3096351  -0.2890317
Petal.Length    0.8818981   -0.3096351 1.0000000   0.9376668
Petal.Width     0.8342888   -0.2890317 0.9376668   1.0000000
```

c) El siguiente gráfico mixto representa las correlaciones entre pares de variables.

```
> corrplot.mixed(corr.res,upper="ellipse",number.cex=.9,tl.cex=.8)
```

Figura 23. Gráfico mixto de correlaciones para el conjunto de datos `iris`



Como indica la barra lateral, el gráfico representa en rojo las correlaciones negativas y en azul las positivas. Asimismo, la anchura e intensidad de cada una de las elipses indica el grado de correlación entre los pares de variables, siendo las elipses más estrechas e intensas las que muestran correlaciones más elevadas.

Así, la correlación más importante se observa entre `Petal.Length` vs. `Petal.Width` (0,94), seguida de `Petal.Length` vs. `Septal.Length` (0,88) y `Petal.Width` vs. `Sepal.Length` (0,83). El resto de pares de variables presentan correlaciones negativas poco significativas.

Glosario

ACP *m* Véase ***principal component analysis***.

AIC Véase **criterio de información de Akaike**.

analysis of variance *m* Extensión de la prueba t con el objetivo de comparar las medias entre más de dos grupos de datos. sigla **ANOVA**.

ANOVA *m* Véase ***analysis of variance***.

area under the curve *f* Área bajo la curva ROC.
sigla **AUC**.

AUC *f* Véase ***area under the curve***.

criterio de información de Akaike *m* Criterio para medir la bondad de un modelo.
sigla **AIC**.

clustering Es un tipo de técnica de aprendizaje no supervisado que divide los registros de datos en grupos, o clústeres, de modo que los registros dentro de un mismo clúster sean similares entre sí y diferentes a los registros de otros clústeres. La similitud se define generalmente en términos de cuán cerca se encuentran los registros en el espacio, basándose en una función de distancia.

coeficiente de determinación *m* (R^2 o *R-squared*) Medida de calidad del modelo que toma valores entre 0 y 1, y la proporción de variación de los resultados que puede explicarse por el modelo.

curva ROC *f* Véase ***receiver operating characteristic***.

exactitud *f* Se refiere a cuán cerca del valor real se encuentra el valor medido. En clasificación, hace referencia a la proporción de muestras correctamente clasificadas para un clasificador en particular.

heterocedasticidad *f* Un modelo de regresión lineal presenta heterocedasticidad cuando la varianza de los errores no es constante en todas las observaciones realizadas.

histograma *m* Es una representación gráfica de una variable en forma de barras, donde la superficie de cada barra es proporcional a la frecuencia de los valores representados.

homocedasticidad *f* Un modelo predictivo presenta homocedasticidad cuando la varianza del error condicional a las variables explicativas es constante a lo largo de las observaciones.

IMC *m* Véase **índice de masa corporal**.

imputar Sustituir valores no informados en una observación.

índice de masa corporal *m* Concepto que relaciona la altura y el peso de un individuo.
sigla **IMC**.

kNN Véase ***k-Nearest neighbours***.

k-Nearest neighbours Permite predecir valores en conjuntos de datos multidimensionales formados por datos mixtos (continuos, discretos, ordinales y/o nominales).
sigla **kNN**.

NA Véase ***not available***.

not available Es un indicador de un dato vacío. sigla **NA**.

potencia estadística *f* La potencia de una prueba estadística o el poder estadístico es la probabilidad de que la hipótesis nula sea rechazada cuando la hipótesis alternativa es verdadera (es decir, la probabilidad de no cometer un error de Tipo II).

principal component analysis *m* Procedimiento estadístico que utiliza una transformación ortogonal para convertir un conjunto de observaciones o variables posiblemente correlacionadas en un conjunto de valores de variables linealmente no correlacionadas llamadas componentes principales.

sigla **ACP**.

receiver operating characteristic Es una representación gráfica de la sensibilidad frente a la especificidad para un sistema clasificador binario según se varía el umbral de discriminación. Otra interpretación de este gráfico es la representación de la razón o ratio de verdaderos positivos frente a la razón o ratio de falsos positivos también según se varía el umbral de discriminación (valor a partir del cual decidimos que un caso es un positivo).
sigla **ROC**.

SRSWOR *f* Siglas en inglés de muestra aleatoria simple sin sustitución.

SRSWR *f* Siglas en inglés de muestra aleatoria simple con sustitución.

tamaño del efecto *m* En estadística, el tamaño del efecto es una medida de la fuerza de un fenómeno (por ejemplo, el cambio en el resultado después de una intervención experimental). El tamaño del efecto calculado a partir de datos es una estadística descriptiva que transmite la magnitud estimada de una relación sin hacer ninguna declaración acerca de si la relación aparente en los datos refleja una verdadera relación en la población.

tasa de error de Tipo I *f* El error de Tipo I, también denominado error de tipo alfa (α) o falso positivo, es el error que se comete cuando el investigador rechaza la hipótesis nula siendo esta verdadera en la población.

tasa de error de Tipo II *f* El error de Tipo II, también llamado error de tipo beta (β) (β es la probabilidad de que exista este error) o falso negativo, se comete cuando el investigador no rechaza la hipótesis nula siendo esta falsa en la población.

Bibliografía

Cook, R. Dennis (1977, febrero). *«Detection of Influential Observations in Linear Regression»*. Technometrics, American Statistical Association (vol. 19, n.º 1, págs. 15-18).

Dale, Kyran (2016). *Data Visualization with Python and JavaScript: Scrape, Clean, Explore & Transform Your Data*. Sebastopol, CA: O'Reilly Media.

Dalgaard, Peter (2008). *Introductory statistics with R*. Berlín: Springer.

Han, Jiawei; Kamber, Micheline; Pei, Jian (2011). *Data Mining: Concepts and Techniques*. Waltham: Elsevier.

Jarman, Kristin H. (2013). *The art of data analysis: how to answer almost any question using basic statistics*. Hoboken, NJ: Wiley.

Mahalanobis, Prasanta C. (1936, enero). «*On the generalized distance in statistics*». Proceedings of the National Institute of Science of India (vol. II, n.º 1).

McKinney, Wes (2012). *Python for data analysis: Data wrangling with Pandas, NumPy, and Ipython*. Sebastopol, CA: O'Reilly Media.

Newton, Rae R.; Rudestam, Kjell E. (1999). *Your Statistical Consultant: Answers to Your Data Analysis Questions*. Thousand Oaks, CA: Sage Publications.

Osborne, Jason W. (2013). *Best practices in data cleaning: A complete guide to everything you need to do before and after collecting your data*. Thousand Oaks, CA: Sage Publications.

Osborne, Jason W. (2010, marzo). «*Data cleaning basics: Best practices in dealing with extreme scores*». Newborn and Infant Nursing Reviews (vol. 10, n.º 1, págs. 37-43).

Osborne, Jason W.; Kocher, Brady, Tillman, David (2012). «*Sweating the small stuff: do authors in APA journals clean data or test assumptions (and should anyone care if they do)*?» [conferencia]. En: Annual meeting of the Eastern Education Research Association (2012: Hilton Head, SC).

Squire, Megan (2015). *Clean Data*. Birmingham: Packt Publishing.

Stekhoven, Daniel J.; Peter Bühlmann (2011, enero). «*MissForest: Non-parametric missing value imputation for mixed-type data*». Bioinformatics (vol. 28, n.º 1, págs. 112-118).