

---

# Preprocesamiento de los datos

---

PID\_00276230

Esteban Vegas Lozano

---

Tiempo mínimo de dedicación recomendado: 5 horas



**Esteban Vegas Lozano**

La revisión de este recurso de aprendizaje UOC ha sido coordinada por la profesora: Teresa Sancho Vinuesa

Segunda edición: septiembre 2020  
© de esta edición, Fundació Universitat Oberta de Catalunya (FUOC)  
Av. Tibidabo, 39-43, 08035 Barcelona  
Autoria: Esteban Vegas Lozano  
Producción: FUOC  
Todos los derechos reservados

*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, mecánico, óptico, grabación, fotocopia, o cualquier otro, sin la previa autorización escrita del titular de los derechos.*

## Índice

<b>Introducción.....</b>	<b>5</b>
<b>1. Lectura de la información.....</b>	<b>7</b>
1.1. Carga de datos en diferentes formatos .....	7
1.2. Verificación de la transmisión de la información .....	8
1.2.1. Tipos de variables estadísticas .....	9
1.2.2. Revisión descriptiva de la matriz de datos .....	10
1.2.3. Variable categórica oculta como variable cuantitativa ..	11
1.2.4. Sistema de codificación de caracteres .....	12
1.3. Cambio de estructura de los datos .....	12
<b>2. Limpieza de los datos.....</b>	<b>14</b>
2.1. Errores sintácticos y normalización de variables .....	14
2.2. Inconsistencias en variables cuantitativas .....	20
2.3. Valores atípicos ( <i>outliers</i> ) .....	21
2.3.1. Valor centinela .....	21
2.3.2. Valor atípico propiamente .....	23
2.4. Valores perdidos ( <i>missing</i> ) .....	26
2.4.1. Imputación de valores .....	26
<b>3. Reducción de los datos.....</b>	<b>29</b>
3.1. Reducción del número de registros .....	29
3.2. Reducción del número de variables .....	31
3.2.1. Análisis de componentes principales .....	35
<b>Resumen.....</b>	<b>55</b>
<b>Bibliografía.....</b>	<b>57</b>



## Introducción

La base de cualquier estudio estadístico es la información recogida, es decir, los datos disponibles para realizar el modelo y obtener las conclusiones. Actualmente se dispone de una gran variedad de datos, con formatos diversos, que pueden ser recopilados en diferentes fuentes, como bases de datos, o captarlos de páginas web. En nuestra sociedad es fácil registrarlos «casi» todo. Por ejemplo, podemos tener un reloj digital que puede monitorizar las pulsaciones u otras funciones fisiológicas muy fácilmente. Estamos en la era de los datos (Mayer-Schönberger y Cukier, 2013) y, en particular, del *big data*.

El término *big data* se refiere a la gran cantidad de datos que superan la capacidad de software convencional para su almacenamiento, procesamiento y análisis. Se caracteriza por tener las 5 V:

- **Volumen:** La cantidad de datos generados y almacenados es de un orden de magnitud de petabytes ( $10^{15}$  bytes) o superior.
- **Velocidad:** Se refiere a la velocidad en que los datos son creados, almacenados y procesados en tiempo real.
- **Variedad:** Se tienen datos de diferentes formatos, tipos y fuentes. Pueden ser datos estructurados, como son las bases de datos, o no estructurados, como correos electrónicos, videos, imágenes o blogs.
- **Variabilidad:** Con tanta cantidad y variedad de datos pueden aparecer incoherencias en el conjunto, lo que provocaría problemas en el manejo y análisis de los datos.
- **Veracidad:** El grado de fiabilidad de la información recogida puede variar mucho entre las diferentes fuentes de información.

Por tanto, cada vez cobra más importancia la preparación de los datos antes de hacer el análisis estadístico (Pyle, 1999; Han y Kamber, 2001). Tener más información no implica mejores resultados. La preparación de los datos se puede dividir en:

- 1) **Lectura de los datos.** Pasar la información bruta recogida y cargarla en el software estadístico adecuadamente.
- 2) **Limpieza de los datos.** Corregir errores, inconsistencias y otros problemas que pueden aparecer en los datos.
- 3) **Reducción de los datos.** En ocasiones, el exceso de información puede empeorar el rendimiento del análisis, tanto a nivel de velocidad y recursos disponibles como a nivel de calidad del análisis. El problema puede ser debido a:

- tener muchos registros,
- tener muchas variables.

En el primer caso, hay que aplicar técnicas para seleccionar los registros sin que se eliminen registros importantes. En el segundo caso, hay que reducir el número de variables por medio, habitualmente, de técnicas multivariantes.

La preparación de los datos suele ser considerada la fase menos atractiva y más aburrida del diseño experimental; la mayoría de los científicos de datos prefieren la selección del modelo y su aplicación y validación. En cambio, es la etapa que se le dedica una gran parte del tiempo y esfuerzo en función de la calidad y variabilidad de los datos originales. Siempre hay que preparar los datos. Habitualmente las fuentes de información no son coherentes, completas y veraces, por lo que se hace imprescindible preparar los datos. No se debe saltar o reducir esta fase porque puede tener consecuencias muy graves en los resultados, como sesgos o incluso conclusiones equivocadas.

En este módulo se presentan las diversas partes del proceso de preparación de datos una vez que ya se ha recopilado la información. En particular, se focaliza en el proceso de limpieza de los datos.

## 1. Lectura de la información

En esta primera etapa se prepara la información bruta, que en el caso más simple está contenida en un fichero tabular, en un objeto del software estadístico con la estructura adecuada para su posterior análisis estadístico.

En nuestro caso se usa el software R. Algunas de las acciones son:

- Carga de datos en diferentes formatos.
- Verificación de la transmisión de la información.
- Cambio de la estructura de los datos.

### 1.1. Carga de datos en diferentes formatos

La información recopilada puede ser muy variada y estar en diferentes formatos. Por ejemplo, formato tabular con delimitadores, como es el caso de fichero del tipo csv, base de datos tipo SQL, documentos Excel, páginas web con formato XML, entre otros formatos.

En la tabla 1 se presenta información sobre algunos de los paquetes que se pueden usar en R para leer según la fuente de datos.

Tabla 1. Paquetes de R para leer ficheros según el formato

Fuente de datos	Paquete	Función	Comentarios
Fichero de texto con campo fijo	utils	read.fwf(). Crea un data frame	Los campos están en posiciones fijas en cada línea
Fichero de texto con delimitadores (tabulaciones, punto y coma...)	utils	read.table(). Lee el fichero y crea un objeto data frame read.csv() y read.csv2(). Casos particulares de read.table() para leer ficheros csv	Los campos están en formato tabular separado por delimitadores
Base de datos	RJDBC	dbConnect(). Establece conexión dbReadTable(). Lee tabla y crea un objeto data frame dbGetQuery(). Realiza una consulta a la base de datos por SQL	Necesita la máquina virtual de Java y el driver del JDBC
Hoja de cálculo Excel	gdata	read.xls()	Necesita perl. Existen otros paquetes
XML o JSON	XML o rjson	readHTMLList() readHTMLTable() fromJSON()	Son más difíciles de usar
SAS, SPSS, Minitab...	foreign	read.xport() (SAS) read.spss() (SPSS) read.mtp() (Minitab)	Hay funciones específicas para cada caso

Fuente de datos	Paquete	Función	Comentarios
Web scraping	rvest	html()	Filtrar y analizar ficheros html

En la mayoría de los casos, el tipo de objeto R que contiene los datos es un `data frame`, una estructura de R preparada para representar las observaciones como filas y las columnas como variables.

Un error que suele pasar cuando se trabaja con ficheros de texto con delimitadores es no poner el delimitador correcto y, por tanto, la carga de datos se hace de manera equivocada. Veamos el caso de ficheros con extensión csv (*comma-separated values*), es decir, los valores se separan por comas. Un documento Excel se puede exportar como extensión csv. En la configuración de Excel con idioma inglés se crea un fichero con el separador coma (,) para los valores y el punto (.) como separador decimal. Este es el formato nativo. En cambio, cuando la coma se usa como separador decimal, tal como está la configuración de Excel de idioma español, se crea un fichero con el separador punto y coma (;) para los valores y con la coma (,) como separador decimal. En la tabla 2 se muestra un ejemplo con ambas variantes del formato csv.

Tabla 2. Ejemplo de fichero con formato csv según idioma

Formato csv inglés	Formato csv español
5.1,3.5,1.4,0.2	5,1;3,5;1,4;0,2
4.9,3.3,1.4,0.2	4,9;3,3;1,4;0,2
4.7,3.2,1.3,0.2	4,7;3,2;1,3;0,2
4.6,3.2,1.5,0.2	4,6;3,2;1,5;0,2
5,3.6,1.4,0.2	5;3,6;1,4;0,2

Para esta situación, existe una función específica en R basada en la función `read.table()` para cada caso. La función `read.csv()` está preparada para leer ficheros con el formato nativo: la coma (,) como separador de valores y el punto (.) como separador decimal. En cambio, `read.csv2()` se usa cuando el separador de valores es el punto y coma (;) y la coma (,) como separador decimal.

## 1.2. Verificación de la transmisión de la información

Una vez que se han cargado los datos en el software estadístico, hay que verificar si la transmisión de la información se ha realizado de manera correcta. Una primera verificación básica es comprobar si el número de observaciones y de variables cargadas en el software estadístico coincide con el número de registros y campos de la base de datos (o archivo). Si pasa este primer filtro se pueden escoger varios registros al azar de la base de datos (o archivo) y comprobar que coincide con las observaciones cargadas en el software estadístico. Es una manera de verificar si la estructura de los datos es correcta.

Una vez hecha esta primera verificación se debe revisar si cada variable cargada en el software estadístico tiene el tipo de variable estadística adecuada.

### 1.2.1. Tipos de variables estadísticas

Se pueden clasificar las variables desde un punto de vista estadístico en función del tipo de contenido:

- Categóricas o cualitativas
  - Nominal
  - Ordinal
- Cuantitativas
  - Discreta
  - Continua

Una variable categórica o cualitativa representa las cualidades en un conjunto de categorías o clases. Esta puede ser de tipo ordinal o nominal, según si el conjunto de categorías puede ordenarse o no.

#### Ejemplo de variable categórica o cualitativa

Un ejemplo sencillo es la variable color, que puede tener como valor (o atributo) un color concreto, por ejemplo amarillo. Los valores de la variable color no tienen un criterio de orden establecido; por tanto, se dice que es una variable categórica nominal. Otros ejemplos de este tipo son el sexo: hombre o mujer, o el grupo sanguíneo: O (universal), A, B, AB. Si la variable cualitativa tiene un criterio de ordenación, entonces se dice que es cualitativa ordinal. Por ejemplo, en la variable nivel de estudios se puede plantear una ordenación de menor a mayor con estos posibles atributos: estudios primarios, estudios secundarios o estudios universitarios. No es necesario que los atributos estén igualmente separados. Por ejemplo, nivel de dolor: leve, moderado o fuerte.

Por otra parte, las variables cuantitativas tienen valores numéricos que siempre tienen un sentido de orden. En el caso de la variable cuantitativa discreta, sus valores presentan interrupciones en la escala de valores que puede tomar. No hay valores intermedios entre dos valores consecutivos. En cambio, la variable cuantitativa continua puede tomar cualquier valor dentro de un intervalo específico. Por eso, en el primer caso los valores son números naturales positivos o negativos, y en el segundo los valores tienen decimales.

#### Ejemplos de variables cuantitativas

Ejemplos de variables cuantitativas discretas son: el número de hijos, el número de productos vendidos en un mes o el número de accidentes en un mes; y en cuanto a las variables cuantitativas continuas: el peso, la altura, la facturación o el beneficio.

En algunas ocasiones, las variables cuantitativas continuas se presentan sin decimales; por ejemplo, el peso de una persona se puede recoger solo en kilogramos. Esto puede confundir y creer que la variable es discreta. Para evitar

esta confusión se debe recordar la naturaleza de la variable, en este caso el peso, y observar que puede medirse con una mayor precisión, hasta en gramos o unidades más pequeñas.

El software estadístico tiene una serie de tipos básicos de variables para contener la información. En la tabla 3 se muestran los tipos básicos de variables de R y su asociación con el tipo de variable estadística.

Tabla 3. Tipos básicos de variables de R

<b>Tipo básico R</b>	<b>Tipo variable estadística</b>
numeric	Datos numéricos: cuantitativa continua
integer	Valores enteros: cuantitativos discretos
factor	Cualitativos nominales
ordered	Cualitativos ordinales
character	Datos de tipo carácter
logical	Variable lógica, con dos posibles valores: TRUE/FALSE
raw	Datos binarios

Cuando R lee un conjunto de datos, deduce los tipos de variables a partir de los valores suministrados. Puede suceder que un dato como la edad, de tipo entero, asigne un tipo categórico (factor). Por tanto, hay que revisar el hecho de que, al leer los datos, se asigne el tipo básico de variable R correctamente. Para conocer el tipo de variable en R se usa la función `class()`. Por ejemplo, si el objeto R `mydata` contiene los datos en la estructura R `data frame`, un código R para mostrar los tipos de variables podría ser: `sapply(mydata, class)`. Aplica la función `class()` a cada variable de `mydata` y obtiene un vector de los tipos de variables R para cada variable.

### 1.2.2. Revisión descriptiva de la matriz de datos

Una manera sencilla de inspeccionar la información cargada es hacer una estadística descriptiva simple de cada variable que debe ser diferente según el tipo de variable. En el caso de variables cuantitativas se pueden mostrar valores mínimos, máximos y algunos cuartiles, en contraste con las variables cualitativas que cabría esperar, una tabla de frecuencia (absoluta o relativa) por categoría. Esta información se usa para:

- Revisar si el contenido de las variables cargadas por el software estadístico coincide con la fuente de información. Si todo ha ido bien, deben dar el mismo resultado.
- Obtener una primera descripción de los datos.

Con el software R es muy fácil obtener esta información; se usa la función `summary()`. Un ejemplo del resultado del objeto `mydata` con 40 observaciones y 5 variables es:

```
> summary(mydata)
    peso      sexo     grupo.sang     nivel.est      n.hijos
  Min.   :53.82  H:21   A : 6   primarios   :14   Min.   :0.00
  1st Qu.:62.52 M:19   AB:12  secundarios  :10   1st Qu.:0.75
  Median :71.65             B :11 universitarios:16 Median :1.50
  Mean   :71.66             O :11                               Mean   :2.00
  3rd Qu.:80.50
  Max.   :88.56
```

### 1.2.3. Variable categórica oculta como variable cuantitativa

Una variable cuantitativa puede enmascarar una variable categórica cuando se usa como atributos valores numéricos en variables cualitativas nominales. Por ejemplo, en algunas ocasiones valores con respuestas Sí o No de variables como *fuma* o *producto disponible* se codifica como 1, si es Sí (fuma o el producto es disponible) o 0, si es No (no fuma o producto no disponible). El software R asigna por defecto el tipo `numeric` a ese campo y por tanto, se necesita corregir posteriormente al tipo `factor`.

A continuación, se presenta un código R donde la variable *fuma* usa como atributo el valor 1 para indicar Sí y 0 para indicar No. La función `as.factor()` convierte a tipo factor donde 0 y 1 son tratados como atributos. La función `factor()` crea una variable del tipo factor donde se pueden indicar las etiquetas para cada nivel.

```
> # Presentar el contenido de la variable fuma
> mydata$fuma
[1] 1 1 1 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
> # ¿Qué tipo de variable es?
> class(mydata$fuma)
[1] "numeric"
> # Cambiar a variable factor
> mydata$fuma <- as.factor(mydata$fuma)
> mydata$fuma
[1] 1 1 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Levels: 0 1
> # ¿Qué tipo de variable es?
> class(mydata$fuma)
[1] "factor"
>
> # Cambiar a variable factor con etiquetas
> mydata$fuma <- factor(mydata$fuma, levels=c(0,1), labels=c("No", "Si"))
> mydata$fuma
```

```
[1] Si Si Si No Si No  

Levels: No Si  

> # ¿Qué tipo de variable es?  

> class(mydata$fuma)  

[1] "factor"
```

#### 1.2.4. Sistema de codificación de caracteres

Nosotros nos comunicamos con un alfabeto que depende del idioma. Por ejemplo, el alfabeto griego es diferente del alfabeto inglés. El ordenador, en cambio, se basa en un sistema binario con dos posibles valores, 0 o 1, que se agrupa en secuencias denominadas bytes. Así que, para comunicar estos dos tipos de códigos, se aplican los sistemas de codificación. Traducen cada carácter de un alfabeto a una secuencia de bytes. Por ejemplo, ASCII es un sistema de codificación que incluye las letras mayúsculas y minúsculas del alfabeto latín, los números arábigos del 0 al 9, signos de puntuación y algunos símbolos especiales como el control de carro o nueva línea. Fue uno de los primeros que existió. Actualmente hay muchos sistemas de codificación, pero Latin1 y UTF-8 son los mayoritarios. El primero suele estar en aplicaciones Windows y el segundo en sistemas operativos basados en Unix.

En R se puede conocer el sistema de codificación que aplica usando la función `Encoding()` con un carácter que no sea ASCII, como por ejemplo una letra acentuada.

```
> Encoding("í")
[1] "latin1"
```

Si la respuesta es «unknown», indica que el sistema de codificación usado es el del sistema operativo. El sistema de codificación local de R se puede averiguar con la función `Sys.getlocale ("LC_CTYPE")`.

Si los datos provienen de un sistema operativo diferente al que se analiza, posiblemente puede aparecer algún error de codificación. En castellano o catalán se puede apreciar que los caracteres no ASCII, como las letras acentuadas, son mostradas de manera extraña. Es un error molesto que puede provocar situaciones anómalas. Si el fichero se abre desde RStudio, para poder cambiar de *encoding* hay que seleccionar la opción *File/Reopen with Encoding*. Esto abrirá un menú con diferentes tipos de *encoding* donde hay que escoger el que se desee y marcar OK.

#### 1.3. Cambio de estructura de los datos

En algunas ocasiones la «forma» o estructura de los datos tal como se lee de las fuentes de información no es la adecuada para hacer su análisis. Una situación común es pasar de un formato largo y estrecho a un formato corto y ancho. Un formato largo y estrecho aparece cuando la información recogida de un mismo

individuo está dispuesta en varios registros. Por ejemplo, recoge información del mismo individuo en condiciones diferentes, o momentos diferentes. En el siguiente ejemplo se presentan los valores obtenidos de cuatro individuos en tres condiciones diferentes, donde hay un registro por cada individuo y condición.

```
> mydata.largo
  individuo sexo condicion valor
  1          1     M      cond1   8.1
  2          1     M      cond2  11.2
  3          1     M      cond3  12.3
  4          2     M      cond1   6.3
  5          2     M      cond2   9.6
  6          2     M      cond3  10.5
  7          3     H      cond1   9.8
  8          3     H      cond2  12.2
  9          3     H      cond3  12.9
 10         4     H      cond1  10.1
 11         4     H      cond2  12.2
 12         4     H      cond3  13.3
```

Al pasar a formato corto y ancho, cada fila es la observación registrada de un individuo diferente y aparecen nuevas columnas. El ejemplo anterior quedaría como sigue, con la creación de tres nuevas variables: cond1, cond2 y cond3.

```
> mydata.ancho
  individuo sexo cond1 cond2 cond3
  1          1     M    8.1   11.2  12.3
  2          2     M    6.3    9.6  10.5
  3          3     H    9.8   12.2  12.9
  4          4     H   10.1   12.2  13.3
```

El software R tiene varias funciones para pasar de un formato largo y estrecho al formato corto y ancho, y viceversa. En el paquete `reshape2` la función `dcast()` se usa para la primera situación. El código R del ejemplo mostrado anteriormente es:

```
mydata.ancho <- dcast(mydata.largo, individuo + sexo ~ condicion, value.var="valor")
```

La función `melt()` sirve para convertir en la dirección opuesta. Para usuarios en R más avanzados y que conozcan el ecosistema `tidyverse`, se pueden usar las funciones `spread()` y `gather()` del paquete `tidyverse`.

## 2. Limpieza de los datos

Una vez recogida la información, se deben limpiar los posibles errores cometidos durante el proceso. Para detectar los errores hay que conocer las características y especificaciones de los datos leídos. La calidad final de los datos es primordial para realizar un buen estudio. A continuación, se describen posibles problemas que pueden aparecer en función del tipo de variable:

- **Variable categórica, tanto nominal como ordinal.** Errores sintácticos, como el hecho de que aparezcan más categorías que las esperadas por errores en la introducción de la información, como por ejemplo tres categorías en la variable sexo. Por tanto, hay que tener presente siempre el número de atributos de cada variable y sus valores para verificar los posibles errores. Otro tipo de error es el semántico, por ejemplo, contradicciones entre edad y fecha de nacimiento (que se pueden identificar al cruzar la información de variables), duplicaciones, como puede ocurrir con las direcciones postales o con claves que deben ser únicas.
- **Variable cuantitativa, tanto discreta como continua.** Hay que verificar que la unidad de medida sea igual en todos los casos; por ejemplo, en el caso de la variable altura, que siempre se use la misma magnitud, en metros con dos decimales. Un problema más habitual es la detección de valores atípicos (*outliers*), es decir, valores que están alejados de la variabilidad esperada. Otro problema común son los valores perdidos (*missing*). En algunas ocasiones son valores que corresponden al valor cero, y en otros son consecuencia de no haber podido recoger la información de la variable de interés.

En resumen, en esta fase pueden aparecer problemas como:

- Errores sintácticos y normalización en variables.
- Inconsistencias en variables cuantitativas.
- Valores atípicos (*outliers*).
- Valores perdidos (*missing*).

### 2.1. Errores sintácticos y normalización de variables

Es bastante común cometer pequeños errores sintácticos fáciles de corregir en variables cualitativas, aunque puede que se necesite información complementaria para decidir cómo corregir el error. En cambio, en variables cuantitativas se puede dar el caso de tener valores con diferentes unidades y, por tanto, hay que corregirlo fijando la misma unidad para todos los valores. Por ejemplo,

todos los valores de la variable altura en metros. Por último, puede ser necesario hacer transformaciones de las variables cuantitativas antes de ser usadas en el análisis estadístico.

Enumeramos algunos casos según el tipo de variable:

### 1) Variables cualitativas:

a) Estandarizar las variables a minúsculas o mayúsculas. En el lenguaje R se puede usar la función `tolower()` o `toupper()`, respectivamente. Por ejemplo,

```
> tolower("Coche")
[1] "coche"
> toupper("Coche")
[1] "COCHE"
```

b) Quitar espacios antes y después del texto o quitar caracteres especiales como el retorno de carro o tabulación. En el lenguaje R se puede utilizar la función `trimws()`. Por ejemplo:

```
> trimws(" El coche azul    \t")
[1] "El coche azul"
```

c) Eliminar acentos. Si las palabras tienen acento puede darse el caso de que, en algunos registros, estén acentuadas y en otros no. Para solucionar el problema se pueden acentuar todos los casos o quitar los acentos. En R se puede aplicar la función `iconv()`. Por ejemplo, quitar el acento de «María».

```
> iconv("Maria", to="ASCII//TRANSLIT")
[1] "Maria"
```

d) Revisión de categorías erróneas por errores sintácticos. Se puede detectar si se revisa la tabla de frecuencias de cada categoría. En R se puede obtener la tabla de frecuencias con la función `table()`. Un ejemplo sencillo es tener tres categorías en la variable sexo cuando se esperan dos.

```
> table(sexo)
sexo
H   M   N
20  14   1
```

e) Pueden aparecer errores más complicados que requieren analizar el contenido de todos los valores para verificar si cumple cierto patrón y, en caso contrario, poder ser rectificado. El software R tiene un gran número de funciones para el análisis de cadenas de caracteres («string»). En los casos más complicados se crea una expresión regular para construir el patrón y después aplicar

la función más adecuada para la situación a resolver. Por ejemplo, se puede verificar que el DNI está formado por ocho dígitos y una letra que puede estar en mayúsculas o minúsculas. El código R podría ser:

```
> expresion_regular <- "^([:digit:]]{8})+([[:alpha:]])$"
> grep(expresion_regular, c( "12345678a", "12345678Y", "123456789", "123456789B" ))
[1] 1 2
```

En el objeto R `expresión_regular` se tiene el patrón a verificar y con la función `grep()` se comprueba. En este caso, solo el primer y el segundo DNI cumplen el patrón.

Existen varios paquetes de R para manejar variables tipo string. En la tabla 4 se muestran varias funciones del paquete `stringr` que sirven para corregir los errores anteriormente indicados.

Tabla 4. Algunas funciones para manejo de strings del paquete `stringr`

Tarea	Función
Estandarizar a mayúsculas/minúsculas/título	<code>str_to_upper()</code> , <code>str_to_lower()</code> , <code>str_to_title()</code>
Eliminar acentos	<code>stri_trans_general()</code>
Recodificar	<code>stri_encode()</code>
Eliminar espacios en blanco antes o/y después de la palabra	<code>trimws()</code>
Detección de secuencia de caracteres («string»)	<code>str_detect()</code>
Extracción/reemplazamiento de string	<code>str_extract()</code>
Separación de string	<code>str_split()</code>
Rellenar string	<code>str_pad()</code>
Separación de palabras	<code>word()</code>

## 2) Variables cuantitativas:

### a) Confusiones con separadores decimales.

- Mezclar la coma decimal y el punto decimal en la misma variable. Por ejemplo, tener valores como 3.4; 3,2; 3,6; 4,7; 3.5 donde están los valores separados por punto y coma, pero los valores 3.4 y 3.5 deberían ser 3,4 y 3,5, respectivamente.
- Tener diferente separador decimal en las variables. Este caso es similar al anterior, pero ahora una variable puede tener como separador decimal la coma y otra variable el punto. Es un caso raro pero posible.
- Confundir la coma (,) como separador decimal cuando es separador de unidades de mil. Esto puede ocurrir con el formato inglés de número. Por ejemplo: 23,456.

**b)** Estandarizar las variables a las mismas unidades. Si los datos se han recogido con diferentes unidades de medida, se debe estandarizar a una unidad concreta. Por ejemplo, las ventas de productos han de estar recogidas en diferentes unidades de tiempo: por día, por semana o por mes. Se debe seleccionar una unidad de medida y estandarizar los datos a esa unidad.

**c)** Estandarizar las variables de fecha y tiempo. Las variables que contengan fecha o/y tiempo son variables cuantitativas con un formato especial; por ejemplo, la fecha puede tener el formato día/mes/año y el tiempo el formato hora:minuto:segundo. Se debe verificar que todos los valores tienen el mismo formato y, en caso contrario, corregirlo. En R existe el tipo `POSIXlt` y `POSIXct` para fechas o/y tiempo y `Date` solo para fechas sin el tiempo con muchas funciones asociadas. Un primer paso a tener en cuenta es cambiar de tipo `character` al tipo adecuado. La función `as.Date()` o `as.POSIXct()` convierte las fechas o el tiempo del tipo `character` al tipo `Date` o `POSIXct`, respectivamente. Un ejemplo de esta conversión es:

```
> ## fecha en formato 'dia/mes/año'
> dates <- c("03/11/17", "07/05/16", "14/02/97", "28/10/99", "02/08/95")
> class(dates)
[1] "character"
> ## Transformar el vector character con el formato 'dia/mes/año' a tipo Date
> dates_1 <- as.Date(dates, "%d/%m/%y")
> class(dates_1)
[1] "Date"
> dates_1
[1] "2017-11-03" "2016-05-07" "1997-02-14" "1999-10-28" "1995-08-02"
```

El objeto R `dates` contiene un vector de fechas de tipo `character`. Con la función `as.Date()` la convierte al tipo `Date` con el formato de fecha año-mes-día.

El paquete `lubridate()` tiene numerosas funciones para facilitar las operaciones básicas con variables de tipo fecha o/y tiempo. El código siguiente produce el mismo resultado anterior pero usando la función `dmy()`. La función espera que el orden de la fecha sea día, mes y año.

```
> library(lubridate)
> dates_3 <- dmy(dates)
> class(dates_3)
[1] "Date"
> dates_3
[1] "2017-11-03" "2016-05-07" "1997-02-14" "1999-10-28" "1995-08-02"
```

Este otro código R muestra la potencia de la función `dmy()` para extraer la fecha.

```
> same_date <- c("20/10/2017", "20 Oct 17", "La fecha es 20-10-2017", "20-10-2017")
> dmy(same_date)
[1] "2017-10-20" "2017-10-20" "2017-10-20" "2017-10-20"
```

En el objeto R `same_date` se presenta la misma fecha con el orden día, mes y año en diferentes situaciones. La función `dmy()` extrae la fecha de manera correcta. Además, existen más funciones para el resto de ordenaciones posibles de día, mes y año: `dmy()`, `mdy()`, `myd()`, `ymd()`, `ydm()`, `dym()`.

### 3) Transformación de las variables:

**a)** Normalización. Es cuando se modifican las variables cuantitativas para que sean comparables entre sí. La transformación más común es la normalización «z-score». La nueva variable,  $z$ , se calcula con la siguiente expresión:

$$z_i = \frac{x_i - \bar{x}}{S}$$

donde  $x$  es la variable original,  $\bar{x}$  es su media aritmética y  $S$  es su desviación típica. Tiene como propiedades que su valor medio es 0 y su desviación típica es 1. Otra transformación es la denominada normalización «min-max». La nueva variable,  $z$ , se obtiene aplicando la siguiente expresión:

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

donde  $\min(x)$  es el valor mínimo de  $x$  y  $\max(x)$  es el valor máximo de  $x$ . La nueva variable se mueve entre 0 y 1.

Existen muchas más funciones para transformar las variables.

**b)** Discretización (*binning*). En algunas ocasiones, la información es recogida con mucho «ruido». Esto quiere decir que el valor real de la variable cuantitativa puede ser bastante diferente del dato recogido; por tanto, interesa reducir este «ruido». Una posible solución es discretizar. Así, una serie de valores son los representantes de la variable cuantitativa. Un ejemplo de aplicación de la discretización en la representación gráfica es el histograma.

El proceso de discretización tiene dos fases:

- Ordenar los valores y hacer la partición. Algunas formas de crear la partición son:
  - Intervalos iguales (*equal-width bins*): Se divide el rango de valores en  $N$  intervalos iguales de tamaño  $(\max - \min)/N$ .

- Aproximadamente igual número de muestras (*equal-depth bins*). Se divide el rango de valores en N intervalos que contienen, aproximadamente, el mismo número de muestras.
- Suavizar cada partición. Para realizar el suavizado se puede aplicar alguno de estos criterios:
  - El valor medio. Se sustituye cada valor de la partición por su valor medio.
  - La mediana. Se sustituye cada valor de la partición por su mediana.
  - Por los límites. Se sustituye cada valor de la partición por el valor más próximo a sus valores extremos.

Para ilustrar cómo funciona el proceso de discretización se usan los siguientes datos ya ordenados:

12, 14, 16, 17, 18, 19, 22, 22, 23, 25, 25, 26, 27, 27, 33, 34, 36, 36

En primer lugar se crea una partición, por ejemplo de tres grupos (*bins*) que quedan como:

- Caso de intervalos iguales:
  - Grupo 1 ([12-20]): 12, 14, 16, 17, 18, 19
  - Grupo 2 ([20-28]): 22, 22, 23, 25, 25, 26, 27, 27
  - Grupo 3 ([28-36]): 33, 34, 36, 36
- Caso de igual número de muestras:
  - Grupo 1: 12, 14, 16, 17, 18, 19
  - Grupo 2: 22, 22, 23, 25, 25, 26
  - Grupo 3: 27, 27, 33, 34, 36, 36

A continuación, se aplican los criterios de suavización. Solo se presenta el caso de la partición con igual número de muestras:

- Criterio del valor medio:
  - Grupo 1: 16, 16, 16, 16, 16, 16
  - Grupo 2: 24, 24, 24, 24, 24, 24
  - Grupo 3: 32, 32, 32, 32, 32, 32
- Criterio de la mediana:
  - Grupo 1: 16, 16, 16, 16, 16, 16
  - Grupo 2: 23, 23, 23, 23, 23, 23
  - Grupo 3: 33, 33, 33, 33, 33, 33
- Criterio de los límites:
  - Grupo 1: 12, 12, 19, 19, 19

- Grupo 2: 22, 22, 22, 26, 26, 26
- Grupo 3: 27, 27, 36, 36, 36, 36

Se puede observar que el resultado final de la discretización puede cambiar notoriamente según el método aplicado.

#### 4) Duplicación de registros:

Suelen ser debido a errores en el proceso de adquisición de la información. Se pueden dar duplicaciones de registros por:

- Tener más de un registro con el mismo identificador. Por tanto, hay que fusionar o seleccionar los valores para conseguir tener un solo registro.
- Tener más de un registro con identificadores diferentes para el mismo individuo. Para detectar esta situación hay que comparar los valores de campos básicos para la identificación de los registros. Por ejemplo, si son clientes se pueden comparar los apellidos o su localización por si aparecen valores iguales o semejantes.

#### 2.2. Inconsistencias en variables cuantitativas

Son errores que pueden haberse ocasionado en el momento de la introducción de la información y son fácilmente reconocibles:

- Valor inválido. Los valores de la variable deben estar incluidos en un determinado intervalo de valores. Por ejemplo, en la variable edad no pueden aparecer valores negativos, o en una variable fecha no todos los valores son posibles.
- Verificar la consistencia del registro al cruzar información de varias variables. Por ejemplo:
  - Si se tiene la fecha de nacimiento y la edad, se puede comprobar que el valor de la edad coincide con el cálculo de la edad a partir de la fecha de nacimiento.
  - En caso de tener variables con resultados parciales y la variable total, se puede confirmar que el cómputo global de las variables parciales es igual a la variable total. Por ejemplo, si se tienen las variables de facturación de cada producto y una variable facturación total, la suma de la facturación de todos los productos debe ser igual a la variable facturación total.
  - Otro caso es tener variables con la dirección postal y su código postal. Se puede comprobar que la dirección postal corresponde al código postal registrado.

### 2.3. Valores atípicos (*outliers*)

Una definición general de valor atípico es aquella observación (o grupos de observaciones) que parecen ser inconsistentes con el conjunto de los datos (Barnett y Lewis, 1994).

Es una idea sencilla pero que es muy difícil de precisar para cada situación, puesto que se debería conocer la distribución teórica de las variables. Valores detectados como valores atípicos pueden ser correctos. Su eliminación en el análisis debe ser meditada, siendo esta una decisión estadística. Se pueden dar dos situaciones:

- Valor centinela
- Valor atípico propiamente

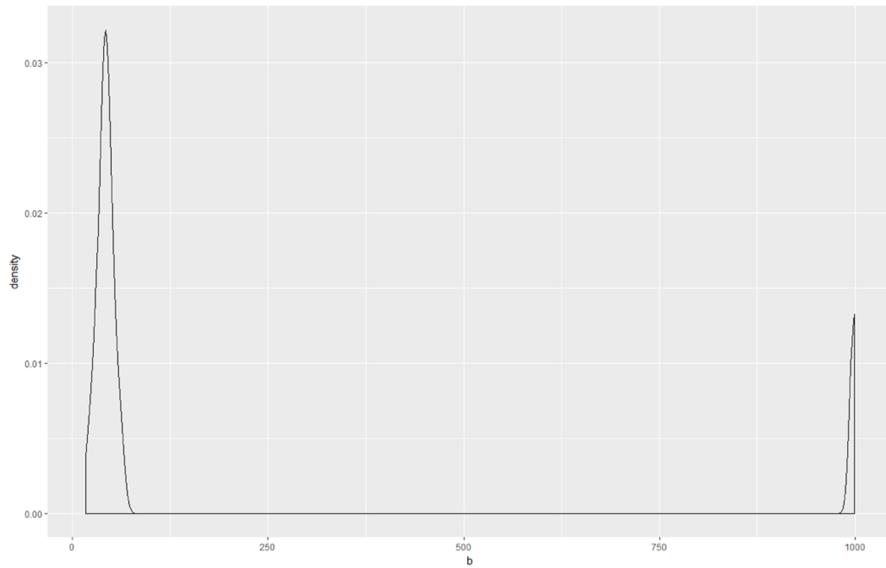
#### 2.3.1. Valor centinela

En algunos casos la detección de valor atípico corresponde a valores centinelas, es decir, a valores que son usados para representar alguna situación especial en una variable numérica. Por ejemplo, «valor desconocido» o «no aplicable». Suele ser un valor numérico con todas las cifras igual a 9. Por ejemplo, si fuera la variable edad podría ser un valor como 999 para indicar edad desconocida. Aunque también puede ser un valor que esté fuera del rango de los valores posibles. Por ejemplo, el valor -1 en la variable edad.

Una manera visual de detectar valores centinelas con valores extremos es representar la distribución de valores de la variable. Si hay valores centinelas se debe presentar un cierto pico en el extremo de la distribución de los valores. A continuación, se presenta un código en R que genera 50 valores bajo normalidad que pueden corresponder a edades. Posteriormente, se añaden 10 valores igual a 999 que sirven para indicar «edad desconocida». Una vez obtenida la serie de edades, se representa su distribución.

```
> library(ggplot2)
> # Generar valores
> b <- trunc(rnorm(50,40,10))
> # Añadir 10 valores igual a 999
> b <- c(b, rep(999,10))
> b
[1] 36 33 58 34 41 30 50 43 36 35 33 48 40 25 77 48 36 44 46 21
[21] 49 45 33 42 48 48 30 30 36 50 35 43 36 46 52 34 26 31 41 32
[41] 50 47 53 48 41 43 24 53 31 17 999 999 999 999 999 999 999 999 999
> # Representar la distribución de los datos
> ggplot(mapping= aes(x=b)) + geom_density()
```

Figura 1. Distribución de valores de una variable con valor centinela



Queda muy claro en el gráfico que hay dos picos, el cercano a 1000 corresponde al valor centinela.

También se puede realizar una estadística descriptiva y observar qué valor es el máximo.

```
> summary(b)
Min. 1st Qu. Median Mean 3rd Qu. Max.
14.00 35.75 43.00 200.30 52.50 999.00
```

El código R para representar «valor desconocido o no disponible» en una variable cuantitativa se escribe NA (*not available*). Un caso diferente es NaN (*not a number*), que aparece cuando se hace alguna operación indefinida como  $0/0$ ,  $\infty-\infty$  y  $\infty/\infty$ . Por tanto, hay que sustituir el valor centinela por NA.

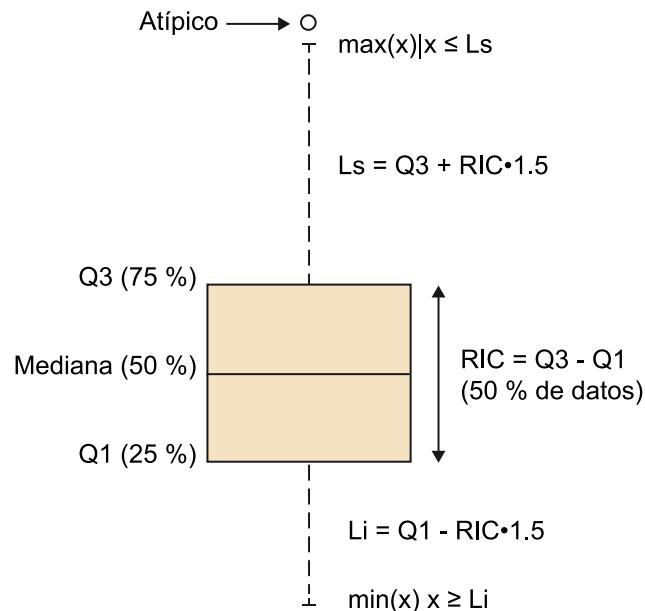
```
> val.centinela <- 999
> # Sustituir el valor centinela por NA
> b[b== val.centinela] <- NA
> # verificar resultado
> summary(b)
Min. 1st Qu. Median Mean 3rd Qu. Max. NA's
15.00 32.25 38.00 37.94 43.75 55.00 10
```

En resumen, en la mayoría de los casos los valores centinelas que son detectados como valor atípico corresponden a valores perdidos (*missing*).

### 2.3.2. Valor atípico propiamente

Una manera sencilla de ver si hay valores atípicos es realizar un diagrama de caja (*box plot*). Es un gráfico (figura siguiente) basado en los valores cuartiles donde aparece una caja central y dos segmentos llamados «bigotes».

Figura 2. Diagrama de caja o *box plot*



Fuente: Wikipedia

La caja central se extiende del cuartil 1 (Q1), que corresponde al percentil 25 %, hasta el cuartil 3 (Q3), que corresponde al percentil 75 %. La diferencia entre Q3 y Q1 se denomina rango intercuartílico (RIC).

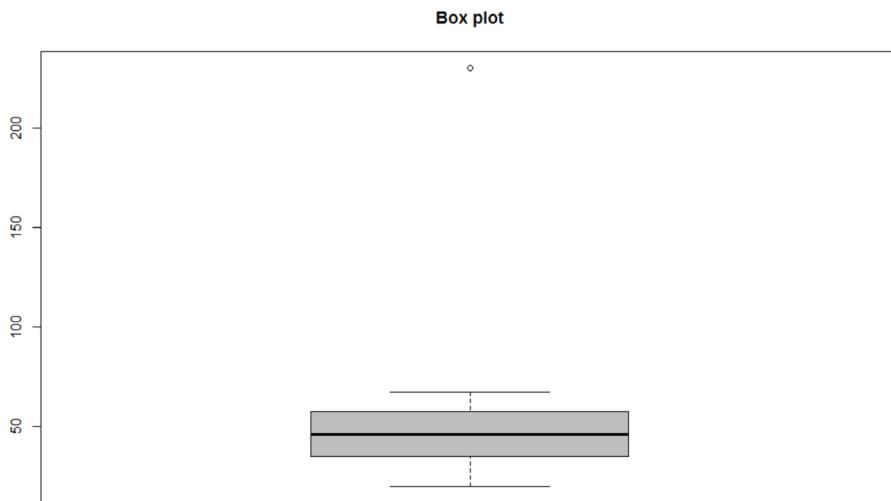
Los valores atípicos se representan como puntos extremos alejados de los «bigotes» o segmentos exteriores a la caja. Los «bigotes» se extienden hasta los valores mínimo y máximo de la serie de datos o hasta 1,5 veces el RIC. Por tanto, son valores inferiores a  $Q1 - 1.5 * RIC$  o superiores a  $Q3 + 1.5 * RIC$ . Por ejemplo, si tenemos los siguientes valores ordenados que corresponden a las edades de los empleados de una empresa:

20 21 25 32 33 37 40 41 44 45 47 49 50 53 55 60 61 62 67 230

el valor 230 es un claro candidato a valor atípico. El código R siguiente muestra cómo obtener el *box plot* y los valores atípicos.

```
> edad<- c(20, 21, 25, 32, 33, 37, 40, 41, 44, 45, 47, 49, 50, 53, 55, 60, 61, 62, 67, 230)
> boxplot(edad, main="Box plot", col="gray")
```

Figura 3. Box plot con un valor atípico



```
> boxplot.stats(edad)$out
[1] 230
```

La función `boxplot.stats()` muestra las características del *box plot* y, en particular, los valores atípicos.

Una vez detectado el valor atípico, se puede considerar producto de un error e intentar corregirlo. Por ejemplo, en el caso de 230, puede ser un error por concatenar el dígito 0 a la edad 23. En algunas ocasiones el valor es cambiado por «no disponible» (NA) y, en otras ocasiones, se queda tal como está ya que puede ser un valor real pero muy poco frecuente.

Existe un subcampo de la estadística denominado «estadística robusta» (Huber, 1981; Leroy, 1987; Maronna y otros, 2006), que desarrolla estimadores y modelos estadísticos alternativos a la estadística clásica pero con la condición de que los estimadores sean resistentes («robustos») al efecto de los valores atípicos y los modelos estadísticos puedan admitir ciertas desviaciones en las condiciones de aplicabilidad. A continuación, se presentan algunos estimadores robustos de tendencia central y dispersión.

El estimador de tendencia central clásico es la media aritmética. Tiene muy buenas propiedades; es un estimador insesgado, pero su estimación está muy influida por los valores atípicos. Si se calcula la media aritmética del ejemplo anterior de la serie de edades, se obtiene un valor de 53,6, que es muy superior a la media aritmética sin considerar el valor 230, que es de 44,32. En cambio, el valor de la mediana es de 46 y sin considerar el valor de 230, la mediana queda en 45, la diferencia es muy pequeña. La mediana es uno de los estimadores robustos más conocidos para medir la tendencia central de los datos. Otro estimador robusto para medir la tendencia central es la media recortada (*trimmed mean*). Se basa en eliminar el mismo porcentaje, k %, de los valores extremos, y calcular la media aritmética del resto de valores. Por ejemplo, la media recortada del 5 % de la serie de edades anterior corresponde a eliminar

el valor menor, 20 y el valor mayor, 230, y calcular la media aritmética del resto de valores. Una variante de la media recortada es la media winsorizada (*winsorized mean*). Los valores extremos eliminados son sustituidos por otros valores. El caso más sencillo es sustituir por el valor menor (o mayor) que ha quedado. Así la media winsorizada del 5 % sustituye el valor de 20 por el valor menor que queda en la serie, 21, y el valor 230 por el valor mayor que queda en la serie, 67, y posteriormente calcula la media aritmética. Otra opción es sustituir los valores eliminados por los cuantiles. En este caso, la media winsorizada del 5 % sustituye el valor de 20 por el cuartil del 5 %, 20,95, y el valor 230 por el cuartil del 95 %, 75,15, y se calcula la media aritmética. Se puede observar que la media recortada o winsorizada del 50 % es equivalente al cálculo de la mediana.

Para medir la dispersión de los datos el estimador más habitual es la desviación estándar, que no es un estimador robusto. La desviación estándar del ejemplo previo con datos de edades es 43,67, pero si se vuelve a calcular sin el valor 230 se obtiene 13,89, un valor muy inferior al obtenido con el valor atípico. El rango intercuartílico (RIC) y la desviación absoluta respecto de la mediana (DAM) son alternativas robustas a la desviación estándar. El RIC es la diferencia entre el cuartil 3 (percentil 75 %) y el cuartil 1 (percentil 25 %) de los datos. Es el tamaño de la caja del gráfico *box plot*. El valor de RIC de los datos de edades es 20,25, no muy diferente del valor de RIC sin el valor 230, que es 19. La DAM es un estimador análogo de la desviación estándar que usa la mediana en lugar de la media aritmética como valor central de los datos; se calcula el valor de la mediana de la diferencia de los valores absolutos entre los datos y su mediana.

$$DAM(y) = \text{mediana}(|y_i - \text{mediana}(y)|)$$

El valor de DAM con los datos de edades es 16,31, y sin el valor 230, es 14,83.

En la tabla 5 se presentan algunas funciones de R para lograr estimaciones robustas y no robustas de tendencia central y dispersión. Además, se muestra el resultado obtenido con el ejemplo de datos de edad para las diferentes funciones.

Tabla 5. Funciones de R para calcular estimaciones robustas y no robustas de tendencia central y dispersión

Estimador	Función	Ejemplo	Resultado
Media aritmética	mean()	mean(edad)	53,6
Mediana	median()	median(edad)	46
Media recortada	mean(x, trim=)	mean(edad, trim=0.05)	45,667
Media winsorizada	winsor.mean(x, trim=) (Paquete psych)	winsor.mean(edad, trim=0.05)	45,905
Desviación estándar	sd()	sd(edad)	43,667

Estimador	Función	Ejemplo	Resultado
Rango intercuartílico (RIC)	IQR ()	IQR (edad)	20,25
Desviación absoluta respecto de la mediana (DAM)	mad ()	mad (edad)	16,309

## 2.4. Valores perdidos (*missing*)

En muchas ocasiones la información recogida no es completa. Faltan algunos valores en ciertas observaciones. Este problema es uno de los más comunes y aparece muy a menudo cuando la información se recoge a partir de encuestas.

Una vez detectada la presencia de datos perdidos (no observados o *missing*) hay que decidir qué hacer. La opción más sencilla sería eliminar las observaciones que tuvieran datos perdidos, pero esto ocasiona un desaprovechamiento de la información recopilada con un aumento del error estándar de las estimaciones de los parámetros. Si son pocas observaciones las afectadas en relación con el total de observaciones, es una opción admisible, pero en caso contrario se debe estimar un valor plausible para llenar el hueco que deja un valor no observado. A este proceso se le denomina *imputación*.

### 2.4.1. Imputación de valores

La elección de la técnica de imputación depende de la relación entre los datos perdidos y los datos. El caso más sencillo es que los datos perdidos no estén relacionados con ninguna variable presente o no en los datos, es decir, completamente al azar. La imputación más sencilla es

$$\hat{y}_i = \bar{y}$$

donde la media de los datos observados ( $\bar{y}$ ) se usa para hacer la imputación del valor perdido ( $\hat{y}_i$ ). No provoca sesgo en la tendencia central de los datos, pero sí en la medida de dispersión. La implementación en R de este procedimiento es muy sencilla:

```
> y[is.na(y)] <- mean(y, na.rm = TRUE)
```

También se puede optar por sustituir la media aritmética por estimadores robustos de la tendencia central como la mediana, media recortada o media winsorizada.

Un segundo modelo de imputación es el denominado *ratio imputation*.

### Referencias bibliográficas

Existe una amplia bibliografía sobre imputación: Rubin (1976), Little y Rubin (1987), Allison (2001), De Waal, Pannekoek y Scholtus (2011).

$$\hat{y}_i = \hat{R}x_i$$

donde  $x_i$  es una covariable de  $y_i$  y  $\hat{R}$  es una estimación del cociente de la media entre  $x$  e  $y$ , que se puede obtener como suma de valores observados de  $y$  dividido por la suma de los correspondientes valores de  $x$ . Este modelo tiene la propiedad que si  $x = 0$  el valor imputado es  $\hat{y} = 0$ . Por tanto, puede ser un modelo adecuado si se tiene como restricciones,  $x \geq 0$  y/o  $y \geq 0$ . Su código en R es:

```
> I <- is.na(y)
> R <- sum(y[!I])/sum(x[!I])
> y[I] <- R * x[I]
```

Un tercer modelo más elaborado es imputar los valores mediante un modelo de regresión lineal (o generalizado)

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{1,i} + \cdots + \hat{\beta}_k x_{k,i}$$

donde  $\hat{\beta}_0, \hat{\beta}_1, \dots, \hat{\beta}_k$  son las estimaciones de los coeficientes de regresión de cada covariable  $x_1, x_2, \dots, x_k$ . Hay que tener en cuenta los problemas que pueden existir al usar el modelo de regresión, como por ejemplo colinealidad y valores influyentes para aplicar adecuadamente la imputación. Por ejemplo, en el código R siguiente la función `lm()` estima la regresión lineal usando solo los registros completos. Posteriormente, reconoce qué registros tienen valores ausentes y hace su predicción con la función `predict()`.

```
> lineal.model <- lm(y ~ x1 + x2 + x3 + x4, data= mydata)
> I <- is.na(y)
> y[I] <- predict(lineal.model, newdata = mydata[I, ])
```

Para finalizar se presenta un modelo más elaborado basado en distancias entre registros, la imputación basada en los  $k$  vecinos más próximos («kNN-imputation»). La idea es encontrar los  $k$  registros más cercanos al registro con uno o más valores perdidos. Entonces, un valor (o el resultado de una función) de los  $k$  registros más próximos sirve para hacer la imputación. Si definimos la distancia de Gower entre dos registros  $i$  y  $j$ ,  $d_g(i, j)$ , como:

$$d_g(i, j) = \frac{\sum_l w_{ijl} d_l(i, j)}{\sum_l w_{ijl}}$$

donde  $l$  indica la  $l$ -ésima variable y  $d_l(i, j)$  es la distancia entre el valor de la variable  $l$  en el registro  $i$  y el registro  $j$ . Para variables cualitativas, cuando el valor para la  $l$ -ésima variable es el mismo en el registro  $i$  que en  $j$  entonces  $d_l(i, j) = 0$ , en caso contrario es 1. En cambio, para variables cuantitativas, la distancia se define como  $d_l(i, j) = |x_{il} - x_{jl}| / (\max(x_{il}) - \min(x_{il}))$ . Cuando en la variable

$l$ -ésima del registro  $i$  o  $j$  es un valor perdido entonces el peso es  $w_{ijl} = 0$ , en caso contrario es 1. El paquete VIM de R tiene la función `kNN()` que usa la distancia de Gower. Un ejemplo sencillo es:

```
> library(VIM)
> mydata.completo <- kNN(mydata)
```

La función `kNN()` obtiene los  $k$  (por defecto son 5) registros más próximos al registro con valores perdidos. En variables cualitativas, el atributo más frecuente en los  $k$  registros más próximos se usa para hacer la imputación. Si la variable es cuantitativa, es el valor de la mediana de los  $k$  registros más próximos.

### 3. Reducción de los datos

En algunas ocasiones, se tiene tanta cantidad de información que es intratable su análisis estadístico.

El objetivo de la reducción de los datos es extraer una representación reducida de toda la información disponible que sea tratable y que contenga las características de la totalidad de la información.

El problema puede dividirse en:

- tener una gran cantidad de registros o instancias,
- tener muchas variables o características.

#### 3.1. Reducción del número de registros

Actualmente, con los medios tecnológicos que se dispone, registrar información de forma «cuasi» continua es relativamente fácil. Esto puede provocar tener un inmenso número de registros. Una de las soluciones habituales es extraer una muestra representativa de todos los registros disponibles. Las formas de extracción más sencillas son:

- **Muestreo aleatorio sin reemplazamiento.** Se basa en seleccionar los registros de forma que todos los registros del conjunto tienen la misma probabilidad de ser seleccionados. Un registro que se ha extraído no puede ser obtenido de nuevo.
- **Muestreo aleatorio con reemplazamiento.** Igual que en el muestreo anterior, los registros se escogen aleatoriamente con igual probabilidad, pero con la diferencia de que cuando se extrae el registro vuelve de nuevo al conjunto y, por tanto, puede volverse a extraer.

En lenguaje R está la función `sample()` para realizar estos tipos de muestreos. Con esta función se obtiene una muestra con o sin reemplazamiento del vector de datos a escoger. En el caso de seleccionar al azar registros de un conjunto de datos se aplica el muestreo a los identificadores de los registros que simplemente pueden ser su posición en la matriz de datos. En el ejemplo siguiente la variable `myregistros` contiene los identificadores de los registros; en este caso sencillo son los valores del 1 al `nrow(mtcars)`. El número de muestras que se quiere extraer es 10. El resultado del muestreo aleatorio sin reemplazamiento se guarda en la variable `mas` y el resultado del muestreo aleatorio con reemplazamiento se guarda en la variable `mas.R`.

```
> # selección de 10 registros del data frame mtcars
> myregistros <- 1: nrow(mtcars)
>
> # identificadores de los registros
> myregistros
[1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
[25] 25 26 27 28 29 30 31 32
>
> # muestreo aleatorio sin reemplazamiento de identificadores
> mas <- sample(myregistros,10)
> mas
[1] 28 10 31 17  7 13 22  4  6 30
>
> # registros completos seleccionados
>
> mtcars[mas,]
          mpg cyl disp hp drat wt qsec vs am gear carb
Lotus Europa 30.4   4  95.1 113 3.77 1.513 16.90 1  1    5    2
Merc 280     19.2   6 167.6 123 3.92 3.440 18.30 1  0    4    4
Maserati Bora 15.0   8 301.0 335 3.54 3.570 14.60 0  1    5    8
Chrysler Imperial 14.7   8 440.0 230 3.23 5.345 17.42 0  0    3    4
Duster 360    14.3   8 360.0 245 3.21 3.570 15.84 0  0    3    4
Merc 450SL    17.3   8 275.8 180 3.07 3.730 17.60 0  0    3    3
Dodge Challenger 15.5   8 318.0 150 2.76 3.520 16.87 0  0    3    2
Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44 1  0    3    1
Valiant       18.1   6 225.0 105 2.76 3.460 20.22 1  0    3    1
Ferrari Dino   19.7   6 145.0 175 3.62 2.770 15.50 0  1    5    6
>
> # muestreo aleatorio con reemplazamiento de identificadores
> mas.R <- sample(myregistros,10,replace = TRUE)
> mas.R
[1]  3  3 32 13 30 15 23  3 23 28
>
> # registros completos seleccionados
>
> mtcars[mas.R,]
          mpg cyl disp hp drat wt qsec vs am gear carb
Datsun 710    22.8   4 108.0  93 3.85 2.320 18.61 1  1    4    1
Datsun 710.1   22.8   4 108.0  93 3.85 2.320 18.61 1  1    4    1
Volvo 142E    21.4   4 121.0 109 4.11 2.780 18.60 1  1    4    2
Merc 450SL    17.3   8 275.8 180 3.07 3.730 17.60 0  0    3    3
Ferrari Dino   19.7   6 145.0 175 3.62 2.770 15.50 0  1    5    6
Cadillac Fleetwood 10.4   8 472.0 205 2.93 5.250 17.98 0  0    3    4
AMC Javelin    15.2   8 304.0 150 3.15 3.435 17.30 0  0    3    2
Datsun 710.2    22.8   4 108.0  93 3.85 2.320 18.61 1  1    4    1
AMC Javelin.1   15.2   8 304.0 150 3.15 3.435 17.30 0  0    3    2
```

Lotus Europa	30.4	4	95.1	113	3.77	1.513	16.90	1	1	5	2
--------------	------	---	------	-----	------	-------	-------	---	---	---	---

### 3.2. Reducción del número de variables

Haber recopilado muchas variables sobre el problema que se investiga no implica obligatoriamente que sea más fácil su análisis. Puede haber muchas variables que no sean informativas o redundantes, pero, además, la gestión es más complicada y algunas técnicas estadísticas no se pueden aplicar. Existen dos posibles soluciones:

1) Selección de variables o características (*feature selection*).

2) Extracción de características (*feature extraction*).

La primera se basa en escoger un grupo de variables originales (también llamadas *características* o *atributos* en el mundo del *machine learning*) que contenga la mayor parte de la información relevante para resolver el problema a tratar. Existen muchas metodologías y es un campo de investigación muy importante (Stanczyk y Jain, 2014). El caso más simple es estudiar la asociación entre las variables continuas mediante la correlación de Pearson. Un valor absoluto de correlación próximo a uno indica que esas variables se comportan de manera similar y, por tanto, con una variable es suficiente, el resto son redundantes.

La función `cor()` se usa en el lenguaje R para obtener la matriz de correlación de todas las variables a la vez.

Veamos un ejemplo de conjunto de datos formado por seis variables continuas:

```
> res <- cor(mydata)
> round(res, 2)
      v1     v2     v3     v4     v5     v6
v1  1.00 -0.85 -0.78  0.68 -0.92  0.42
v2 -0.85  1.00  0.79 -0.71  0.86 -0.43
v3 -0.78  0.79  1.00 -0.45  0.73 -0.71
v4  0.68 -0.71 -0.45  1.00 -0.65  0.09
v5 -0.92  0.86  0.73 -0.65  1.00 -0.32
v6  0.42 -0.43 -0.71  0.09 -0.32  1.00
```

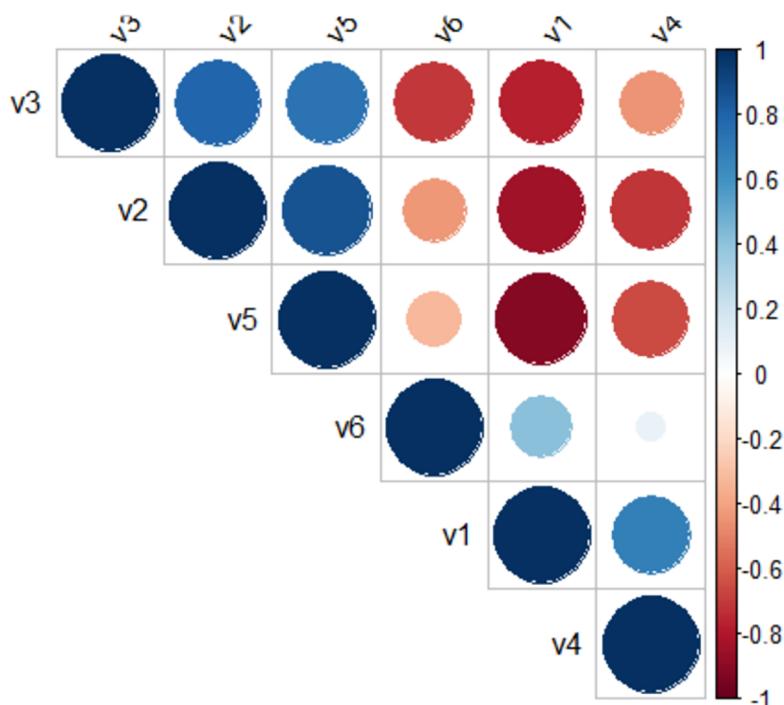
Se obtiene una matriz de tamaño seis por seis simétrica donde se muestra el valor de correlación de Pearson para cada pareja de variables. Se observa que hay tres valores de correlación muy cercanos a uno o menos uno: -0,92, 0,86 y -0,85. El primer valor obtenido entre v1 y v5, y el último obtenido entre v1 y v2 son valores negativos, así que las variables están linealmente relacionadas en sentido inverso (correlación negativa); esto quiere decir que cuando aumenta el valor de una variable el valor de la otra variable disminuye. En cambio, en el otro caso, v2 frente a v5, las variables presentan una dependen-

cia lineal en sentido directo (correlación positiva). En estos casos una variable es redundante respecto a la otra y se podría eliminar. Por tanto, con las variables v1, v3, v4 y v6 se podría recoger casi la misma información del conjunto de datos que con todas las variables. Como consecuencia, se mejora la interpretabilidad y se reduce el tamaño del conjunto de datos.

Un gráfico que puede ser útil para visualizar toda esta información es:

```
install.packages("corrplot")
library(corrplot)
corrplot(res, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45)
```

Figura 4. Gráfico de correlación entre variables



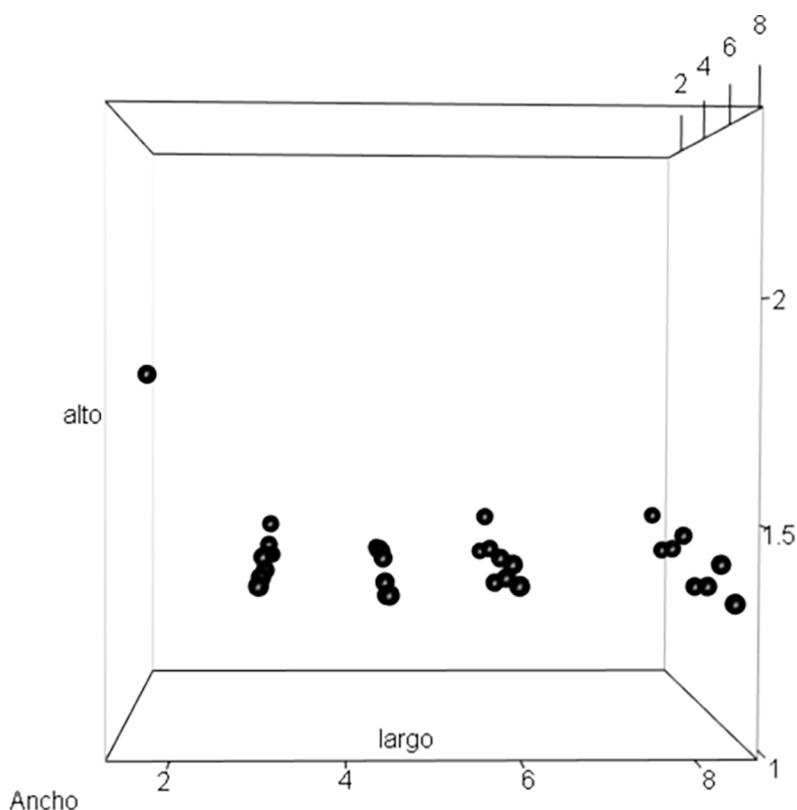
La segunda solución, extracción de características, transforma el espacio construido con las variables originales de alta dimensión (muchas variables) a un espacio de menor dimensión, es decir, aplica una reducción de la dimensión. El análisis de componentes principales (Everitt y Dunn, 2010) que se suele indicar como ACP en español o PCA en inglés, es una de las técnicas estadísticas multivariantes más usada para describir/explorar un conjunto de datos numéricos. Existen muchas otras técnicas de reducción de la dimensión como Multidimensional Scaling (Everitt y Hothorn, 2011) y el Análisis de Correlación Canónica (Härdle y Simar, 2015), entre otras.

La idea fundamental de las técnicas de reducción de dimensión es representar un conjunto de datos con un gran número de variables en solo dos o tres nuevas variables o componentes que deben cumplir algún criterio de optimización. Esto quiere decir que, si el conjunto de datos tiene  $p$  variables y, por tanto, cada observación se puede representar como un punto en un espacio de  $p$  dimensiones, se reduce el espacio de representación de las observaciones de  $p$  dimensiones a dos o tres dimensiones. En consecuencia, ahora se puede visualizar cada observación como un punto en el plano o en el espacio.

Para ilustrar el concepto de *reducción de dimensión* se muestra el siguiente ejemplo.

Nos fijamos en el conjunto de datos correspondiente a la posición de la cabeza de los individuos que están en una clase de 32 sillas, dispuestas en 4 filas y 8 columnas, además del profesor. Observamos que el alumnado estará sentado, por tanto, a una altura similar y el profesor de pie a una altura superior. Por tanto, nuestras observaciones corresponden a una matriz de datos con tres variables: largo, ancho, alto. Así que cada observación, en este caso cada individuo que está en el aula, tendrá tres valores que se pueden representar en tres dimensiones. En este gráfico se muestra cómo podría ser la representación:

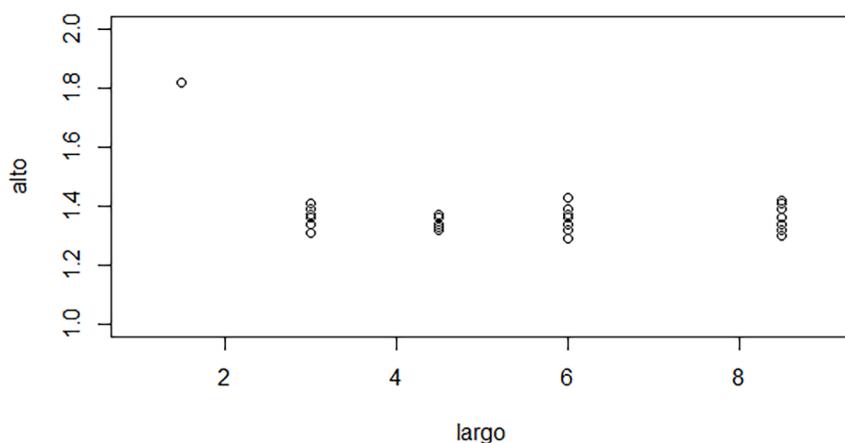
Figura 5. Representación tridimensional de la posición de la cabeza de los alumnos y del profesor



En la figura anterior se observa que el punto a la izquierda, con valor más alto, corresponde al profesor, ya que está de pie. El resto de los puntos se sitúan en cuatro filas y ocho columnas con alturas muy similares, ya que son alumnos sentados.

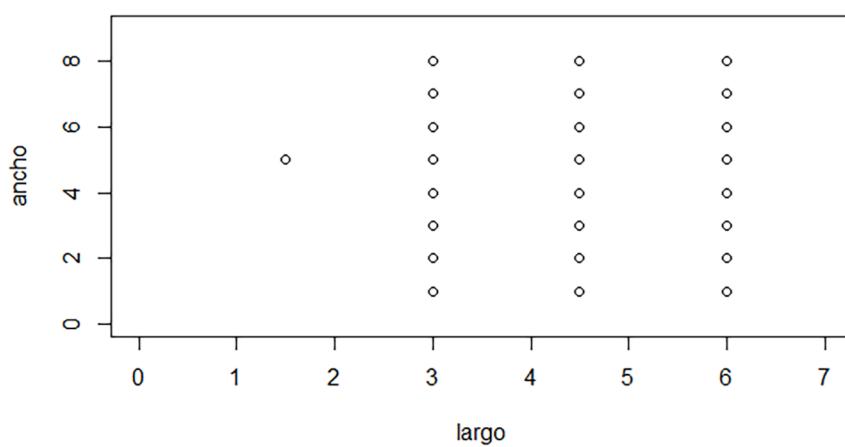
Una representación de las observaciones en el plano (dos dimensiones) sería una reducción de dimensión. Por ejemplo, se pueden proyectar los puntos sobre la pared y quedarían así:

Figura 6. Representación en dos dimensiones de la posición de la cabeza de los alumnos y del profesor de la figura 2, como proyección a la pared



Otra posible reducción de la dimensión en dos dimensiones puede ser proyectar los puntos sobre el suelo. La representación quedaría así:

Figura 7. Representación en dos dimensiones de la posición de la cabeza de los alumnos y del profesor de la figura 2, como proyección al suelo



Observemos que las observaciones se disponen de diferente manera según la proyección. Por ejemplo, en la figura 6 no queda clara la existencia de las ocho columnas de sillas equidistantes, mientras que la figura 7 no muestra que te-

nemos los alumnos a diferentes alturas. Por tanto, en una técnica de reducción de dimensión (o proyección) siempre hay una pérdida de información o, incluso, una deformación.

### 3.2.1. Análisis de componentes principales

El análisis de componentes principales busca una proyección que recoja la máxima variabilidad medida a partir de la varianza. ¿Cómo se hace? Se construyen sucesivas combinaciones lineales de las variables originales tal que entre ellas estén incorrelacionadas y tengan varianzas cada vez menores.

En el ámbito matemático, estas combinaciones lineales se obtienen de la solución de un problema de descomposición de valores singulares de la matriz de datos. La forma más habitual de resolver este problema es mediante diagonalización. Si  $X$  corresponde a la matriz de datos con  $n$  observaciones y  $p$  variables, el primer paso es calcular la matriz de covarianzas ( $C$ ) o correlaciones ( $R$ ) entre las  $p$  variables. A continuación, se diagonaliza la matriz  $M$  (puede ser  $C$  o  $R$ ) aplicando esta expresión:

$$(M - \lambda_k I) v_k = 0$$

Donde  $0$  es el vector con todos sus elementos igual a cero,  $I$  es la matriz identidad, es decir, es una matriz diagonal con valor 1 en toda la diagonal,  $\lambda_k$  corresponde al valor propio (*eigenvalue*, en inglés)  $k$ -ésimo, que cumplen que son valores positivos o cero con  $\lambda_1 > \lambda_2 > \dots > \lambda_p$  y  $v_k$  corresponde al vector propio (*eigenvector*, en inglés) asociado al valor propio  $k$ -ésimo:

Valor propio  $\lambda_1$  le corresponde el vector propio  $v_1 = (v_{11}, v_{12}, \dots, v_{1p})$

Valor propio  $\lambda_2$  le corresponde el vector propio  $v_2 = (v_{21}, v_{22}, \dots, v_{2p})$

...

Valor propio  $\lambda_p$  le corresponde el vector propio  $v_p = (v_{p1}, v_{p2}, \dots, v_{pp})$

Las nuevas variables, que se llamarán *componentes principales* (*cp*), se construyen como combinación lineal de las variables originales ( $x$ ) y el vector propio ( $v$ ). Hay tantas componentes principales como dimensiones.

$$cp_1 = v_{11} \cdot x_1 + v_{12} \cdot x_2 + \dots + v_{1p} \cdot x_p$$

$$cp_2 = v_{21} \cdot x_1 + v_{22} \cdot x_2 + \dots + v_{2p} \cdot x_p$$

...

$$c_p = v_{k1} \cdot x_1 + v_{k2} \cdot x_2 + \dots + v_{kp} \cdot x_p$$

Los valores propios  $\lambda_1, \dots, \lambda_p$  indican la cantidad de varianza explicada en cada componente principal. Si se usan todas las componentes principales como nuevas variables, en lugar de solo unas cuantas, la cantidad de varianza acumulada será igual que la obtenida con todas las variables originales; no habrá ninguna pérdida.

Para que la proyección obtenida sea factible se necesita que haya un grado de correlación alto entre las variables, es decir, que exista redundancia en la información recogida en la matriz de datos. Si esto no ocurre, es decir, que la matriz de correlación de las variables originales es casi diagonal, el ACP no tendrá efectividad.

Para ilustrar todo lo comentado hasta el momento se presenta un ejemplo clásico y sencillo basado en el conjunto de datos medidos sobre flores de la planta del género iris (Fisher, 1936). Estos datos contienen 150 ( $n$ ) valores para tres especies de plantas del género iris: *Iris versicolour*, *Iris setosa* e *Iris virginica*. Las variables ( $p$ ) recogidas son: longitud y anchura del sépalo, longitud y anchura del pétalo. Todas estas medidas están en centímetros. Los datos han sido descargados de UCI Machine Learning Repository.

En primer lugar se realiza la lectura, la presentación de los datos y el cálculo de la correlación en lenguaje R.

```
# Lectura y preparación del data frame
iris <- read.csv("iris.data", header = FALSE)
colnames(iris) <- c("sep.len", "sep.wid", "pet.len", "pet.wid", "species")
head(iris)
iris.m <- iris[,-5]

# Calculo de la correlación
cor(iris.m)
```

Los seis primeros registros son:

```
> head(iris)
  sep.len  sep.wid  pet.len  pet.wid    species
1     5.1      3.5     1.4      0.2 Iris-setosa
2     4.9      3.0     1.4      0.2 Iris-setosa
3     4.7      3.2     1.3      0.2 Iris-setosa
4     4.6      3.1     1.5      0.2 Iris-setosa
5     5.0      3.6     1.4      0.2 Iris-setosa
6     5.4      3.9     1.7      0.4 Iris-setosa
```

Una vez leídos los datos, se obtiene la matriz de correlaciones de las cuatro variables. Su resultado es:

```
> cor(iris.m)
      sep.len    sep.wid    pet.len    pet.wid
sep.len  1.0000000 -0.1093692  0.8717542  0.8179536
sep.wid -0.1093692  1.0000000 -0.4205161 -0.3565441
pet.len  0.8717542 -0.4205161  1.0000000  0.9627571
pet.wid  0.8179536 -0.3565441  0.9627571  1.0000000
```

Se observa que hay una fuerte correlación entre la longitud del sépalo y la longitud y anchura del pétalo. Además, la longitud y anchura del pétalo también están fuertemente correlacionadas. A partir de estos resultados se puede esperar que el ACP pueda realizar una reducción de dimensión efectiva.

Existen dos funciones en R para realizar el ACP: `prcomp()` y `princomp()` con similares características. Para realizar este ejemplo se usa la primera de ellas: `prcomp()`.

Una primera pregunta que hay que contestar es: ¿qué matriz debe usarse para realizar la diagonalización? Como regla general, si la matriz de datos está formada por variables con diferentes magnitudes y rangos de valores diversos, la mejor opción es la matriz de correlaciones, ya que corresponde a transformar las variables originales en estandarizadas con media cero y desviación típica uno, consiguiendo que todas las variables tengan un rango de variabilidad similar. En cambio, cuando la matriz de datos está formada por variables con magnitudes iguales se tiene también la posibilidad de solo centrar las variables y usar la matriz de covarianzas. En este caso, cada variable tendrá un rango de dispersión diferente produciendo estructuras factoriales mejor definidas (Tinsley y Tinsley, 1987). En este ejemplo estamos en la segunda situación. Para ver cómo puede afectar usar una matriz u otra vamos a desarrollar los dos escenarios.

**1) Matriz de covarianzas con los datos centrados.** El código R sería:

```
# Cálculo del ACP caso M. covarianzas
ACP.cov <- prcomp(x=iris.m, center = TRUE, scale.= FALSE)
summary(ACP.cov)

ACP.cov$sdev # raíz cuadrada de los valores propios
var.exp.cov <- (ACP.cov$sdev^2 / sum(ACP.cov$sdev^2))*100 # % de varianza explicada en cada CP
ACP.cov$rotation # vectores propios
ACP.cov$x # Componentes principales: Valores proyectados
```

Se escoge `center = TRUE` y `scale.=FALSE` para diagonalizar la matriz de covarianzas. Con `ACP.cov$sdev` se presenta la desviación estándar de cada componente principal que corresponde a la raíz cuadrada de los cuatro valores propios.

La función `summary()` muestra en la primera fila estos valores, en la segunda fila la proporción de la varianza para cada componente, y en la siguiente fila la proporción de la varianza acumulada hasta llegar a la última componente principal. Podemos observar que las variables originales que están en dimensión 4 se transforman en un nuevo espacio también de dimensión 4, pero usando menos dimensiones explicarán una gran parte de la variabilidad de los datos.

```
> summary(ACP.cov)
Importance of components:
PC1       PC2       PC3       PC4
Standard deviation   2.0554  0.49218  0.28022  0.15389
Proportion of Variance 0.9246  0.05302  0.01719  0.00518
Cumulative Proportion 0.9246  0.97763  0.99482  1.00000
```

La varianza de cada componente principal es el valor propio y se calcula elevando al cuadrado `ACP.cov$sdev`. Para obtener el porcentaje de la varianza explicada en cada componente principal hay que dividir cada valor propio entre la suma de todos los valores propios, y da un resultado de 92,46 %, 5,30 %, 1,72 %, 0,52 %. Así que la primera componente puede resumir un 92,46 % de varianza de las cuatro variables originales. Por tanto, se puede hacer una reducción de la dimensión a la primera de ellas, puesto que recoge la mayor parte de la varianza total.

```
> var.exp.cov <- (ACP.cov$sdev^2 / sum(ACP.cov$sdev^2))*100 # % de varianza explicada en cada CP
> var.exp.cov
[1] 92.4616207  5.3015568  1.7185140  0.5183085
```

Los vectores propios están en el objeto `ACP.cov$rotation`.

```
> ACP.cov$rotation
PC1       PC2       PC3       PC4
sep.len  0.36158968 -0.65653988  0.58099728  0.3172545
sep.wid -0.08226889 -0.72971237 -0.59641809 -0.3240944
pet.len  0.85657211  0.17576740 -0.07252408 -0.4797190
pet.wid  0.35884393  0.07470647 -0.54906091  0.7511206
```

El primer vector propio asociado al primer valor propio corresponde a la primera columna, el segundo vector propio asociado al segundo valor propio corresponde a la segunda columna, y así sucesivamente.

Las nuevas variables, denominadas *componentes principales*, son:

$$cp_1 = 0,36159 \cdot x_1 - 0,08227 \cdot x_2 + 0,85657 \cdot x_3 + 0,35884 \cdot x_4$$

$$cp_2 = -0,65654 \cdot x_1 - 0,72971 \cdot x_2 + 0,17577 \cdot x_3 + 0,07471 \cdot x_4$$

$$cp_3 = 0,58099 \cdot x_1 - 0,59642 \cdot x_2 - 0,07252 \cdot x_3 - 0,54906 \cdot x_4$$

$$cp_4 = 0,31725 \cdot x_1 - 0,32409 \cdot x_2 - 0,47972 \cdot x_3 + 0,75112 \cdot x_4$$

Ahora ya se pueden calcular las proyecciones de las nuevas variables en el nuevo espacio. Solo hay que sustituir  $x_1$ ,  $x_2$ ,  $x_3$ ,  $x_4$  por el valor de longitud del sépalo ( $x_1$ ), anchura del sépalo ( $x_2$ ), longitud del pétalo ( $x_3$ ) y anchura del pétalo ( $x_4$ ) correspondiente a cada flor una vez que han sido centrados.

Por ejemplo, la primera flor tiene unos valores de 5,1, 3,5, 1,4, 0,2 para cada variable. El valor medio de cada variable es 5,843, 3,054, 3,759, 1,199, respectivamente. Por tanto, los valores de la primera flor una vez centrados son: -0,743, 0,446, -2,359, -0,999. Los nuevos valores se obtienen aplicando esta combinación lineal:

$$\begin{aligned} cp_1 &= 0,36159 \cdot (-0,743) - 0,08227 \cdot 0,446 + 0,85657 \cdot (-2,359) + 0,35884 \cdot (-0,999) \\ &= -2,684 \end{aligned}$$

$$\begin{aligned} cp_2 &= -0,65654 \cdot (-0,743) - 0,72971 \cdot 0,446 + 0,17577 \cdot (-2,359) + 0,07471 \cdot (-0,999) \\ &= -0,327 \end{aligned}$$

$$\begin{aligned} cp_3 &= 0,58099 \cdot (-0,743) - 0,59642 \cdot 0,446 - 0,07252 \cdot (-2,359) - 0,54906 \cdot (-0,999) \\ &= 0,022 \end{aligned}$$

$$\begin{aligned} cp_4 &= 0,31725 \cdot (-0,743) - 0,32409 \cdot 0,446 - 0,47972 \cdot (-2,359) + 0,75112 \cdot (-0,999) \\ &= 0,001 \end{aligned}$$

El objeto `ACP.cov$x` contiene la proyección de todos los valores. Así que la primera fila corresponde al primer valor, que coincide (salvo pequeñas diferencias debidas al cálculo con más decimales) al valor obtenido anteriormente.

```
> ACP.cov$x[1,]
      PC1        PC2        PC3        PC4
-2.684207125 -0.326607315  0.021511837  0.001006157
```

**2) Matriz de correlaciones.** El código R sería:

```
# Cálculo del ACP caso M. correlaciones
ACP.cor <- prcomp(x=iris.m, center = TRUE, scale.= TRUE)
summary(ACP.cor)
```

```
ACP.cor$sdev # raíz cuadrada de los valores propios
var.exp.cor <- (ACP.cor$sdev^2 / sum(ACP.cor$sdev^2))*100 # % de varianza explicada en cada CP
ACP.cor$rotation # vectores propios
ACP.cor$x # Componentes principales: Valores proyectados
```

Nótese que la única diferencia con el código anterior es que se escoge TRUE en los argumentos center y scale. De esta manera se indica que la matriz a diagonalizar es la matriz de correlaciones.

En `summary(ACP.cor)` se obtienen estos resultados:

```
> summary(ACP.cor)
Importance of components:
PC1      PC2      PC3      PC4
Standard deviation   1.7061  0.9598  0.38387 0.14355
Proportion of Variance 0.7277  0.2303  0.03684 0.00515
Cumulative Proportion 0.7277  0.9580  0.99485 1.00000
```

Como era de esperar no se obtienen los mismos resultados que en el caso de diagonalizar la matriz de covarianzas. Observamos que la proporción de la varianza 0,7277, 0,2303, 0,03684, 0,00515 es diferente. La primera componente principal tiene una proporción de varianza menor que si diagonalizamos la matriz de covarianzas, pero observamos que la proporción de varianza acumulada entre las dos primeras componentes principales es muy similar a la diagonalización de la matriz de covarianzas.

En `ACP.cor$rotation` se encuentran los vectores propios.

```
> ACP.cor$rotation # vectores propios
PC1      PC2      PC3      PC4
sep.len  0.5223716 -0.37231836  0.7210168  0.2619956
sep.wid -0.2633549 -0.92555649 -0.2420329 -0.1241348
pet.len  0.5812540 -0.02109478 -0.1408923 -0.8011543
pet.wid  0.5656110 -0.06541577 -0.6338014  0.5235463
```

También se observa que los valores son diferentes a la diagonalización de la matriz de covarianzas. Las componentes principales son:

$$cp_1 = 0,52237 \cdot x_1 - 0,26335 \cdot x_2 + 0,58125 \cdot x_3 + 0,56561 \cdot x_4$$

$$cp_2 = -0,37232 \cdot x_1 - 0,92556 \cdot x_2 - 0,02109 \cdot x_3 - 0,06541 \cdot x_4$$

$$cp_3 = 0,72102 \cdot x_1 - 0,24203 \cdot x_2 - 0,14089 \cdot x_3 - 0,63380 \cdot x_4$$

$$cp_4 = 0,26199 \cdot x_1 - 0,12413 \cdot x_2 - 0,8012 \cdot x_3 + 0,52355 \cdot x_4$$

Ahora, para calcular las proyecciones de las nuevas variables en el nuevo espacio, solo hay que sustituir  $x_1, x_2, x_3, x_4$  por el valor de longitud del sépalo ( $x_1$ ), anchura del sépalo ( $x_2$ ), longitud del pétalo ( $x_3$ ) y anchura del pétalo ( $x_4$ ) correspondiente a cada flor una vez que han sido estandarizados.

En el caso de los datos de la primera flor sus valores estandarizados son: –0,8976739, 1,0286113, –1,336794, –1,308593, respectivamente. Así que, la combinación lineal para obtener las componentes principales de la primera flor es:

$$\begin{aligned} cp_1 &= 0,52237 \cdot (-0,898) - 0,26335 \cdot 1,029 + 0,58125 \cdot (-1,337) + 0,56561 \cdot (-1,309) \\ &= -2,257 \end{aligned}$$

$$\begin{aligned} cp_2 &= -0,37232 \cdot (-0,898) - 0,92556 \cdot 1,029 - 0,02109 \cdot (-1,337) - 0,06541 \cdot (-1,309) \\ &= -0,504 \end{aligned}$$

$$\begin{aligned} cp_3 &= 0,72102 \cdot (-0,898) - 0,24203 \cdot 1,029 - 0,14089 \cdot (-1,337) - 0,63380 \cdot (-1,309) = \\ &0,121 \end{aligned}$$

$$\begin{aligned} cp_4 &= 0,26199 \cdot (-0,898) - 0,12413 \cdot 1,029 - 0,8012 \cdot (-1,337) + 0,52355 \cdot (-1,309) = \\ &0,023 \end{aligned}$$

Para comprobar que el cálculo es correcto, se muestra la primera fila del objeto `ACP.cor$x` que contiene la proyección de todos los valores.

```
> ACP.cor$x[1,]
      PC1        PC2        PC3        PC4
-2.25698063 -0.50401540  0.12153619  0.02299628
```

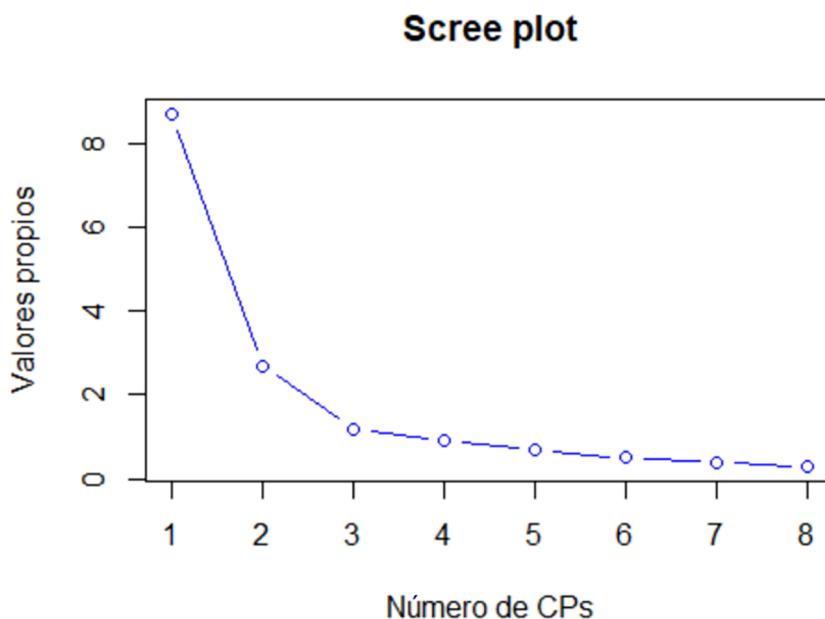
Se observa cómo coincide salvo diferencias en el tercer o cuarto decimal debido a redondeos.

### **¿Cómo decidir el número de componentes principales a usar?**

Existen dos criterios habituales para decidir el número de componentes principales a seleccionar: la regla de Kaiser-Guttman y el test de *scree*.

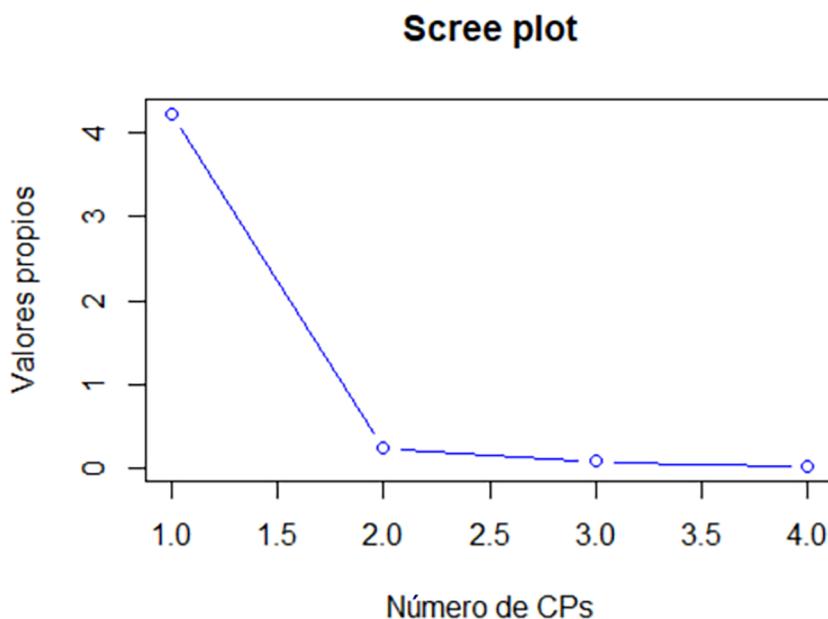
La regla de Kaiser-Gutman dice que se seleccionan aquellas componentes con un valor propio mayor que la media de valores propios. En contraste, el test de *scree* se basa en un criterio visual basado en el *scree plot*. En este gráfico los valores propios se representan como puntos de mayor a menor valor que se unen mediante una línea. La curva ideal debería tener una bajada muy pronunciada y doblarse en un «codo». Este sería el punto de corte visual, luego ya se aplana. En la figura 8 se presenta un *scree plot* donde el «codo» cogería las componentes principales 1, 2 y hasta la 3; después la pendiente ya es mucho más suave.

Figura 8. Scree plot



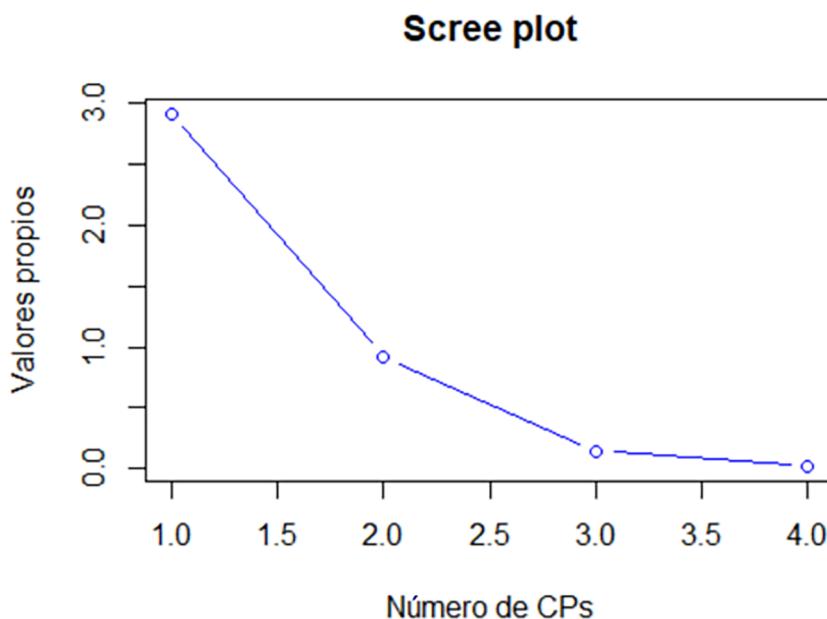
A continuación, se aplican los criterios presentados con el ejemplo de la flor de la planta del género iris. En el caso de realizar el ACP con la matriz de covarianzas, los valores propios son 4,22484077, 0,24224357, 0,07852391 y 0,02368303, y su valor medio es 1,1423. Aplicando el criterio de Kaiser-Gutman es suficiente con seleccionar la primera componente principal. Por otro lado, si se observa el gráfico de *scree* (figura 9), el «codo» se forma con el primer valor propio y después es plano. En este caso, los criterios coinciden en el número de componentes principales a escoger. Es suficiente con seleccionar una dimensión.

Figura 9. Scree plot de datos iris calculados con matriz de covarianzas



Si se realiza el ACP con la matriz de correlaciones, sus valores propios son 2,91081808, 0,92122093, 0,14735328 y 0,02060771, y 1 es el valor medio. Según el criterio de Kaiser-Gutman se seleccionaría la primera componente, aunque el valor propio de la segunda componente es cercano a 1. En cambio, en el gráfico de *scree* (figura 10) no se observa un «codo» claro. Podría estar sobre la tercera componente, así que se escogerían las dos primeras componentes. En este caso no hay coincidencia en el resultado. Se escoge la opción de las dos primeras componentes ya que el grado de variabilidad explicado es alto y es fácil realizar interpretaciones en el plano.

Figura 10. Scree plot de datos iris calculados con matriz de correlaciones

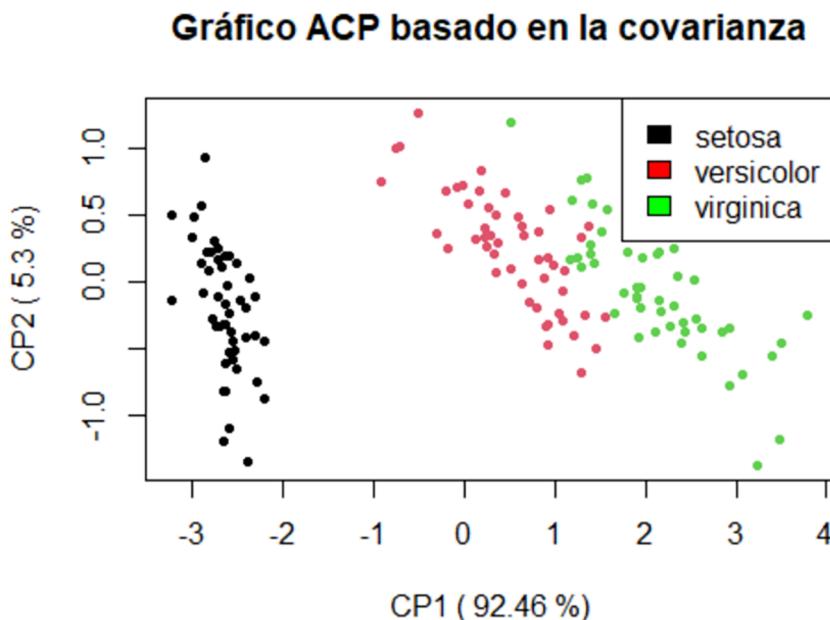


### Representación de los datos en dimensión reducida

Una vez escogido el número de componentes principales que vamos a usar para representar, solo queda realizar el gráfico e interpretar los resultados obtenidos.

Siguiendo con los datos de la planta del género iris, el ACP, tanto con la matriz de covarianzas como la matriz de correlaciones, se proyectará en el plano para poder comparar de manera similar ambas representaciones. Así que se pasa del espacio original de cuatro dimensiones a una proyección de solo dos dimensiones con un porcentaje de varianza explicada mayor del 95 %. En la figura 8 se presenta la proyección del ACP con la matriz de covarianzas.

Figura 11. Representación de las dos primeras componentes principales de datos iris calculados con matriz de covarianzas

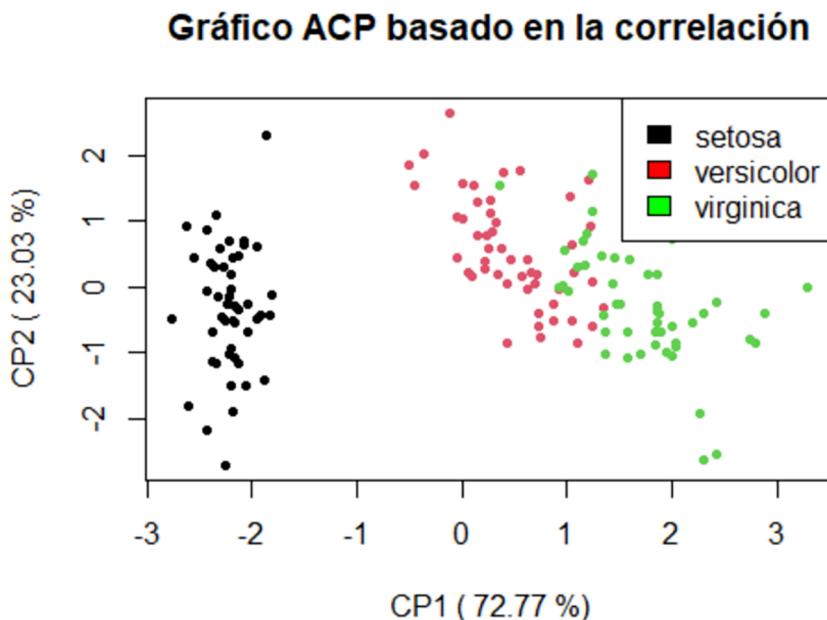


El código R para realizar la figura 11 es:

```
plot(ACP.cov$x[,1], ACP.cov$x[,2],
  xlab = paste("CP1 (", round(var.exp.cov[1],2), "%)"),
  ylab = paste("CP2 (", round(var.exp.cov[2],2), "%)"),
  main = "Gráfico ACP basado en la covarianza",
  col= factor(iris$species), pch=20)
legend("topright",c("setosa","versicolor","virginica"),
  fill=c("black","red","green"))
```

De manera similar se obtiene el ACP de las dos primeras componentes basado en la matriz de correlaciones (figura 12).

Figura 12. Representación de las dos primeras componentes principales de datos iris calculados con matriz de correlaciones



El código R que ha sido usado para obtener la figura 12 es el siguiente:

```
plot(ACP.cor$x[,1],ACP.cor$x[,2],
  xlab = paste("CP1 (", round(var.exp.cor[1],2), "%)"),
  ylab = paste("CP2 (", round(var.exp.cor[2],2), "%)"),
  main = "Gráfico ACP basado en la correlación",
  col= factor(iris$species), pch=20)
legend("topright",c("setosa","versicolor","virginica"),
  fill=c("black","red","green"))
```

Se puede observar que ambos ACP representan los puntos de una manera muy similar. La primera conclusión que se puede extraer es que los puntos se agrupan por especie a partir de las características de su flor. Esto quiere decir que la varianza entre especies es mayor que dentro de la especie. La segunda conclusión es que la primera componente separa las tres especies de iris. *Iris setosa* (puntos negros) está muy separado de las otras dos especies que tienen una zona de solapamiento. Como *Iris versicolor* (puntos rojos) e *Iris virginica* (puntos verdes) están muy próximos deben tener características comunes. Por último, la primera componente que explica más de un 70 % de la varianza total puede ser un buen identificador de la especie de iris.

Para interpretar las componentes principales en función de las variables originales hay que conocer qué variables influyen más en la construcción de la componente principal, es decir, aquellas que tienen un valor absoluto mayor. En el caso de la primera componente principal tenemos:

$cp_1 = 0,36159 \cdot x_1 - 0,08227 \cdot x_2 + 0,85657 \cdot x_3 + 0,35884 \cdot x_4$  para matriz de covarianzas

$cp_1 = 0,52237 \cdot x_1 - 0,26335 \cdot x_2 + 0,58125 \cdot x_3 + 0,56561 \cdot x_4$  para matriz de correlaciones

Nótese que el valor mayor corresponde a la longitud del pétalo ( $x_3$ ). En el caso del ACP basado en la matriz de covarianzas tiene un valor de 0,86, mucho más alto que el siguiente valor 0,36 que corresponde a la longitud del sépalo ( $x_1$ ) y a la anchura del pétalo ( $x_4$ ). Por tanto, la primera componente principal se basa principalmente en la longitud del pétalo y, en menor medida, en la longitud del sépalo y anchura del pétalo. Así, flores de iris con valores altos de longitud del pétalo, sépalo y anchura del pétalo tendrán un valor mayor de la primera coordenada principal. En el caso del ACP construido con la matriz de correlaciones, las variables más influyentes para la primera coordenada principal son las mismas que en el ACP de covarianzas pero con pesos diferentes: la longitud del pétalo es 0,58, la anchura del pétalo es 0,57 y la longitud del sépalo es 0,52. Son valores muy semejantes y la interpretación de la primera componente principal es similar al ACP basado en la matriz de covarianzas.

En el caso de la segunda componente principal, las variables con mayor valor absoluto son anchura y longitud del sépalo en ambos ACP (ved la combinación lineal de la  $cp_2$ ). Observamos que tienen valor negativo, esto indica que a mayor anchura y longitud del sépalo de una flor, el valor de la segunda componente principal será menor. En el caso del ACP construido con la matriz de covarianzas los pesos son de -0,73 para la anchura del sépalo y de -0,66 para la longitud del sépalo. Valores semejantes indican que ambas variables ponderan de manera semejante en la elaboración de la segunda componente principal. En cambio, en el ACP basado en la matriz de correlaciones, los pesos son -0,93 para la anchura del sépalo y -0,37 para la longitud del sépalo. La anchura del sépalo tiene una mayor ponderación que la longitud del sépalo para la obtención de la segunda componente principal.

En las figuras 13 y 14 se ilustran las interpretaciones indicadas anteriormente. Se observa cómo el *Iris setosa* tiene una flor más pequeña (longitud y anchura del pétalo) que las otras dos especies de iris, aunque la anchura del sépalo puede ser tan grande o más que las otras dos especies. El *Iris virginica* suele tener mayor longitud tanto de sépalo como de pétalo y, además, mayor anchura de pétalo.

Figura 13. ACP de las dos primeras componentes principales de datos iris calculados con matriz de covarianzas y reescalando los puntos en función de una variable original. Longitud del sépalo

(sep.len), longitud del pétalo (pet.len), anchura del sépalo (sep.wid) o anchura del pétalo (pet.wid).

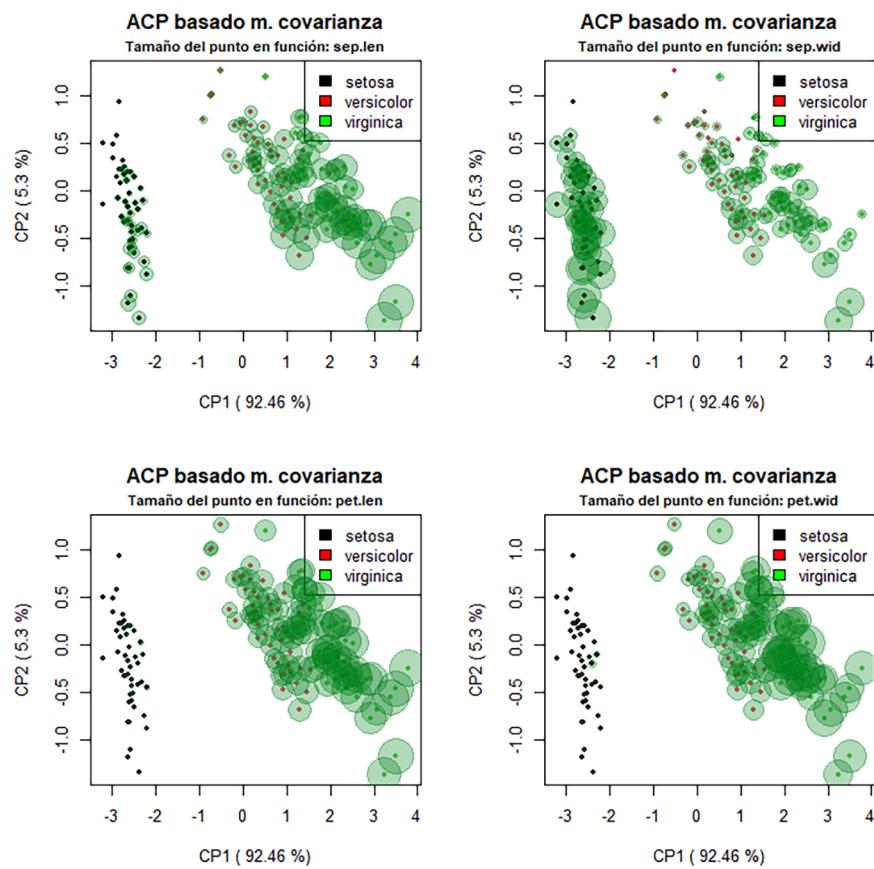
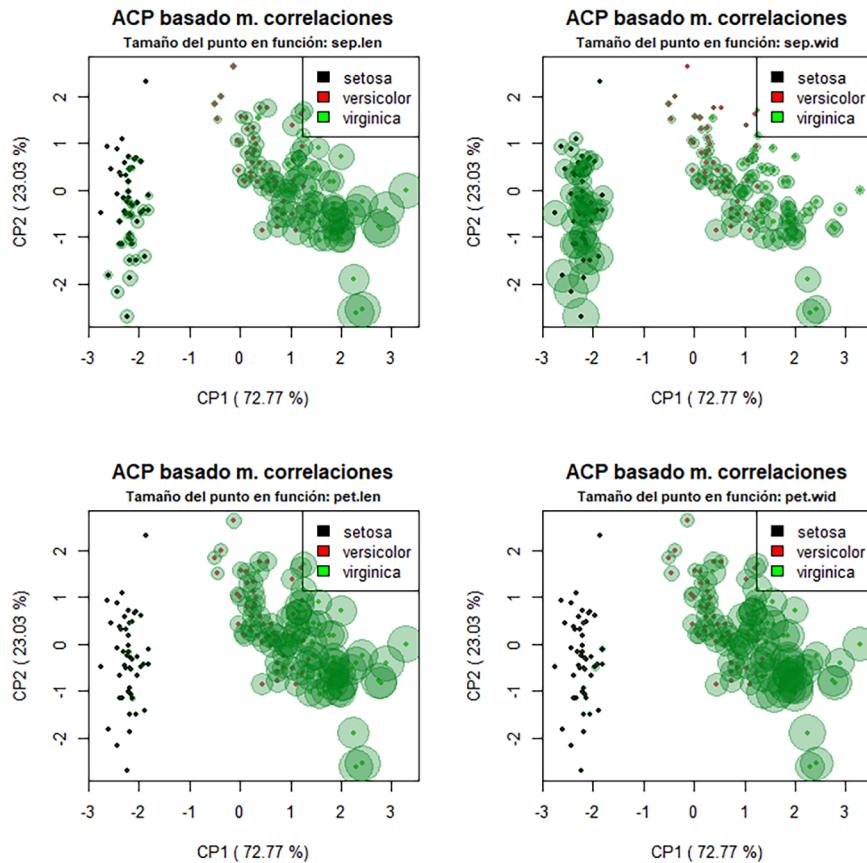


Figura 14. ACP de las dos primeras componentes principales de datos iris calculados con matriz de correlaciones y reescalando los puntos en función de una variable original. Longitud del

sépalo (*sep.len*), longitud del pétalo (*pet.len*), anchura del sépalo (*sep.wid*) o anchura del pétalo (*pet.wid*).



A continuación se muestra la función de R para realizar uno de estos ACP con los puntos reescalados. Observamos que la idea es reescalar cada punto en función del valor de la variable seleccionada entre 0,1 y *maxsize* +0,1.

```
ACP.cov.rescale.points <- function(var, maxsize) {
  # Rescale size points in function of variable value
  var <- var

  val.var<-iris.m[, var]
  maxsize <- maxsize
  rescale <- ((val.var - min(val.var))/(max(val.var)-min(val.var)))*maxsize + 0.1

  plot(ACP.cov$x[,1],ACP.cov$x[,2],
       xlab = paste("CP1 (", round(var.exp.cov[1],2), "%)"),
       ylab = paste("CP2 (", round(var.exp.cov[2],2), "%)"),
       main = "ACP basado m. covarianza",
       #sub = var ,
       col= factor(iris$species), pch=20)
  legend("topright",c("setosa","versicolor","virginica"),
         fill=c("black","red","green"))
  title(main = paste("Tama o del punto en función:", var),
        line = 0.5,
```

```
cex.main = 0.8)
points(ACP.cov$x[,1], ACP.cov$x[,2],
       pch = 19,
       cex = rescale,
       col = rgb(0, 0.5, 0.1, 0.3))

}

ACP.cov.rescale.points("sep.len", 5)
```

## Representación de datos y variables en dimensión reducida: ACP Biplot

El gráfico de ACP que representa además de los puntos muestrales las variables originales es el ACP Biplot. Con este gráfico se puede valorar e interpretar la relevancia de las variables originales en cada componente principal y la asociación entre ellas. Se suelen representar las variables originales como flechas que parten del origen. El sentido de la flecha indica la dirección del incremento de la variable. La longitud de la flecha indica su relevancia e influencia en la proyección representada. Flechas muy próximas indican que las variables originales tienen un alto grado de correlación entre ellas. Las coordenadas de las flechas corresponden a los valores de los vectores propios.

En las figuras 15 y 16 se muestra cómo la variable longitud y anchura del pétalo se incrementa en el mismo sentido, preferentemente siguiendo la primera componente principal. Anchura y longitud del sépalo se incrementan, en cambio, en sentidos diferentes, aunque ambos en sentido inverso al incremento de la segunda componente principal. La principal diferencia entre ambos ACP Biplot es la ponderación de los pesos de las variables. Resalta que la variable con mayor longitud, es decir con mayor peso, en el ACP construido con la matriz de covarianzas es la longitud del pétalo, mientras que en el caso del otro ACP es la anchura del sépalo. Todas estas apreciaciones coinciden con las realizadas con los valores propios de las dos primeras componentes en el apartado anterior.

Figura 15. ACP Biplot de las dos primeras componentes principales de datos iris calculados con matriz de covarianzas

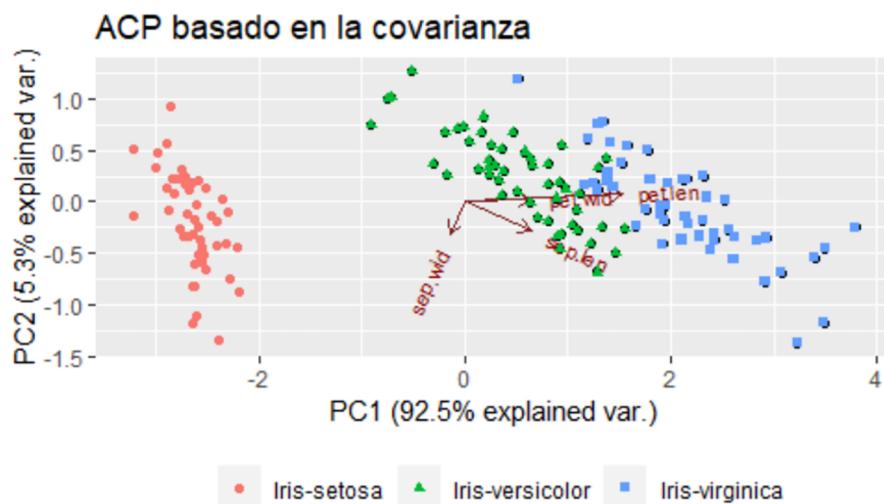
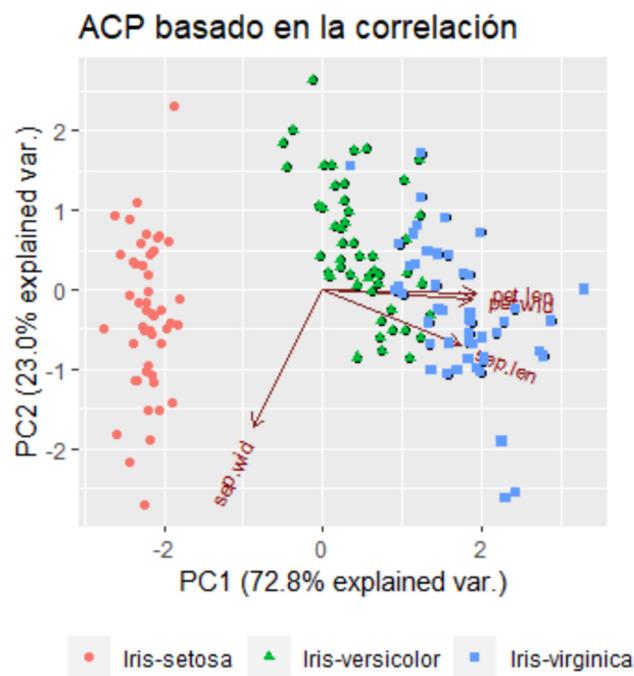


Figura 16. ACP Biplot de las dos primeras componentes principales de datos iris calculados con matriz de correlaciones



El código R para realizar el ACP Biplot usa la función `ggbiplots()` del paquete con el mismo nombre:

```
library(ggbiplots)
biplot.ACP.cov <- ggbiplots(ACP.cov, scale = 1, obs.scale = 1) +
  geom_point(aes(shape = factor(iris$species)),
```

```

            colour= factor(iris$species))) +
            ggtitle("ACP basado en la covarianza") +
            theme(legend.direction = 'horizontal',
                  legend.position = 'bottom',
                  legend.title = element_blank())
print(biplot.ACP.cov)

```

En resumen, el ACP Biplot es una buena manera de realizar interpretaciones de las componentes principales en función de las variables originales, además de explorar el espacio de representación y los puntos de este espacio.

## Detección de valores atípicos

Como se ha visto anteriormente, el ACP proyecta las observaciones en un espacio de dimensión reducida donde quedan reflejadas como puntos. La distribución de estos puntos nos puede dar algunas informaciones importantes sobre características del conjunto de datos. Se pueden observar agrupaciones de puntos (clústeres) o valores que están alejados de la nube de puntos general. Estos puntos (o punto) son candidatos a ser valores atípicos. Un ejemplo de esta situación se ilustra con el código R siguiente:

```

# Creación de la muestra
set.seed(998)

# Primeras 30 observaciones
mat0 <- matrix(rnorm(90), 30, 3)
var4<- 2*mat0[,3]+rnorm(30)
var5 <- -2 *mat0[,1]+rnorm(30)
mat0 <- cbind (mat0,var4,var5)
dimnames(mat0) <- list(paste("Observación", 1:30), paste0("Var", 1:5))

# Observacion 31
mat1 <- matrix(rnorm(5, mean = 3), 1, 5)
dimnames(mat1) <- list(paste("Observación", 31), paste0("Var", 1:5))

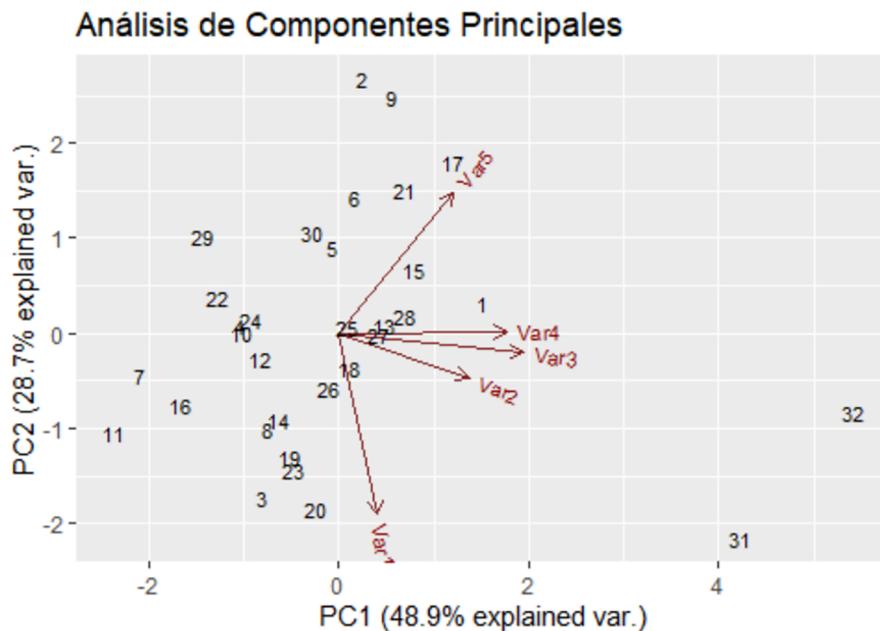
#Observacion 32
mat2 <- matrix(c(rnorm(2, mean = 3),rnorm(3, mean = 4)), 1, 5)
dimnames(mat2) <- list(paste("Observación", 32), paste0("Var", 1:5))
# Agrupación de las 32 Observaciones
mat <- rbind(mat0,mat1,mat2)

```

```
# Cálculo PCA  
myPCA <- prcomp(mat, center = T, scale. = T)  
  
  
library(ggbiplot)  
biplot.ACP <- ggbiplot(myPCA, scale = 1, obs.scale = 1,  
                        labels = 1:32) +  
  ggttitle("Análisis de Componentes Principales")  
  
print(biplot.ACP)
```

Se generan 32 observaciones con 5 variables. Las tres primeras variables en las 30 primeras observaciones siguen una distribución normal de media 0 y desviación típica 1, la cuarta y quinta variable están relacionadas linealmente (más un error aleatorio) con la tercera y primera variable, respectivamente. De esta manera se genera redundancia entre variables. En la observación 31 todas sus variables siguen una distribución normal de media 3 y desviación típica 1. Por último, en la observación 32 las tres primeras variables siguen una distribución normal de media 3 y desviación típica 1, y las dos últimas variables siguen una distribución normal de media 4 y desviación típica 1. Por tanto, la mayoría de las observaciones, 30, siguen la misma distribución, solo hay dos observaciones con un comportamiento distinto. La última observación tiene una distribución más diferente a la mayoría de las observaciones que la penúltima. Por tanto, en la visualización de las observaciones al hacer un ACP (figura 17) se observa una nube de los treinta primeros puntos próximos entre ellos y los dos últimos puntos más alejados de la nube de puntos. El más alejado es la observación 32, con una distribución de las variables más diferente de la mayoría de las observaciones. Además, el primer eje, PC1, es el que muestra mayor variabilidad, mientras que el segundo eje, PC2, mucho menos.

Figura 17. ACP Biplot de las dos primeras componentes principales mostrando las observaciones 31 y 32 alejadas de la nube de puntos.



### Regresión por componentes principales

Cada día es más fácil registrar un gran número de variables, lo que puede llevar a que muchas de ellas sean redundantes o no informativas. Otra utilidad del ACP consiste en sustituir las variables originales por un número de componentes principales suficiente como para recoger un porcentaje elevado de la varianza total. El número de nuevas variables puede ser muy reducido. Si se usan estas nuevas variables para hacer una regresión estaremos realizando una regresión por componentes principales. El problema de multicolinealidad en este tipo de regresión no existe, ya que las componentes principales están incorrelacionadas entre ellas por propia construcción. De esta manera se consigue un modelo de regresión más simple sin perder mucha de la información original.

## Resumen

En este módulo se muestra la importancia que tiene la preparación de los datos antes del análisis estadístico. Es una tarea poco glamorosa y divertida, aunque es básica para poder aprovechar toda la información disponible y no cometer errores en las conclusiones. La cantidad de tiempo necesario para la preparación de datos depende directamente de la salud de los datos. Y en la mayoría de los casos siempre hay algún tipo de problema, habitualmente asociado a errores o falta de información.

Se ha dividido la preparación de los datos en tres partes:

- 1) Lectura de datos.
- 2) Limpieza de los datos.
- 3) Reducción de los datos.

La lectura de los datos tiene como objetivo básico transmitir la información al software estadístico sin cometer errores y adecuarlo al software. Un punto importante a revisar es si se ha asignado a cada variable el tipo de variable estadística adecuada: cuantitativa discreta o continua o, cualitativa nominal u ordinal. La limpieza de los datos es el punto fundamental de la preparación de los datos. Pueden aparecer:

- errores sintácticos y de normalización en variables,
- inconsistencias entre variables,
- valores atípicos y
- valores perdidos o no observados.

Los valores atípicos se pueden detectar realizando *box plot* e intentando minimizar sus efectos con la aplicación de estimadores robustos. Respecto a los valores perdidos, hay gran cantidad de métodos para intentar imputar los valores faltantes y conseguir registros completos como por ejemplo, el k Nearest Neighbor. Por último, puede ser necesario realizar una reducción de los datos. En el caso de tener un gran número de registros, se extrae una muestra representativa del total de registros. En cambio, si el problema es el gran número de variables, se pueden aplicar técnicas de reducción de la dimensión, como el ACP, para extraer las características de los datos y sustituir a las variables originales en el análisis estadístico. El ACP también puede usarse para observar la disposición de la nube de puntos y detectar agrupaciones o datos atípicos. Además, el ACP es una de las maneras más usuales de explorar los datos desde el punto de vista multivariante.



## Bibliografía

- Allison, P.** (2001). *Missing values*. Thousand Oaks, CA: Sage Publications.
- Barnett, V.; Lewis, T.** (1994). *Outliers in Statistical Data* (3.<sup>a</sup> ed.). Nueva York: John Wiley & Sons.
- De Waal, T.; Pannekoek, J.; Scholtus, S.** (2011). *Handbook of statistical data editing and imputation. Wiley handbooks in survey methodology*. Nueva York: John Wiley & Sons.
- Everitt, B. S.; Dunn, G.** (2010). *Applied Multivariate Data Analysis*. Nueva York: John Wiley & Sons.
- Everitt, B.; Hothorn, T.** (2011). *An introduction to applied multivariate analysis with R*. Berlín: Springer Science & Business Media.
- Fisher, R. A.** (1936). «The use of multiple measurements in taxonomic problems». *Annual Eugenics* (vol. 7, parte II, págs. 179-188). También en *Contributions to Mathematical Statistics* (1950). Nueva York: John Wiley.
- Han, J.; Kamber, M.** (2001). *Data Mining: Concepts and Techniques*. San Francisco: Morgan Kaufmann Publishers, Inc.
- Härdle, W. K.; Simar, L.** (2015). «Canonical correlation analysis». En: *Applied multivariate statistical analysis* (págs. 443-454). Berlín: Springer.
- Huber, P.** (1981). *Robust Statistics*. Wiley.
- Little, R.; Rubin, D.** (1987). *Statistical Analysis with Missing Data*. Nueva York: John Wiley & Sons.
- Maronna, R. A.; Martin, R. D.; Yohai, V. J.** (2006). *Robust statistics: Theory and methods*. Wiley.
- Mayer-Schönberger, V.; Cukier, K.** (2013). *Big Data. A Revolution That Will Transform How We Live, Work and Think*. UK: John Murray.
- Pyle, D.** (1999). *Data Preparation for Data Mining*. San Francisco: Morgan Kaufmann Publishers, Inc.
- Rousseeuw, P. J.; Leroy, A. M.** (1987). *Robust regression and outlier detection*. Nueva York: John Wiley & Sons.
- Rubin, D.** (1976). *Inference and missing data* (vol. 3, núm. 63, págs. 581-592). *Biometrika*.
- Stanczyk, U.; Jain, L. C.** (2014). *Feature Selection for Data and Pattern Recognition*. Springer.
- Tinsley, H. E.; Tinsley, D. J.** (1987). «Uses of factor analysis in counseling psychology research». *Journal of counseling psychology* (vol. 34, n.<sup>o</sup> 4, pág. 414).

