JOSÉ ANTONIO QUESADA RICO

BIOESTADÍSTICA BÁSICA Y AVANZADA CON R

DIAZ DE SANTOS

Madrid • Buenos Aires • México • Bogotá

Ediciones Díaz de Santos
Internet: http//www.editdiazdesantos.com
E-mail: ediciones@editdiazdesantos.com

ISBN: 978-84-9052-068-0 (edición papel)
e- ISBN: 978-84-9052-093-2 (edición digital)
Depósito Legal: M-25062-2024

Diseño de cubierta y Fotocomposición: P55 Servicios Culturales

Printed in Spain - Impreso en España

Índice

PARTE II.
Bioestadística avanzada

Prefacio

Para quién es este libro

Este no es un libro para aprender Bioestadística ni es un libro para aprender *R*, este libro trata de cómo aplicar métodos bioestadísticos con el entorno *R* [1].

Para la Primera Parte, *Bioestadística básica,* se recomienda que el lector conozca previamente métodos estadísticos inferenciales básicos, como realizar e interpretar estimaciones, contrastes de hipótesis, conceptos de modelos multivariantes, etc. También el lector debe tener conocimientos básicos del entorno R, como operaciones con variables, dataframes, listas, matrices, utilización de bucles, sentencias if, etc. Este libro no utiliza métodos o conceptos de `tydeverse`, por lo que no es necesario tener estos conocimientos.

Para la Segunda Parte, *Bioestadística avanzada*, se recomienda que el lector tenga conocimientos básicos de modelos mixtos, regresión no-lineal, ajuste y validación, aunque se realizará una introducción básica en cada apartado.

Contexto de aplicación

Esta obra es el fruto de más de 20 años de dedicación profesional al análisis de datos en Ciencias de la Salud, mostrando soluciones sencillas mediante código R directo, a problemas reales de investigación traslacional e investigacion aplicada en Biomedicina.

La investigación traslacional es la investigación que traslada los conocimientos de la investigación básica a las personas. Por tanto, las unidades de análisis serán personas o pacientes, en contextos de estudios epidemiológicos o clínicos.

La bioestadística es la parte de análisis de datos dentro del proceso investigador, pero antes del análisis de datos, es necesario haber diseñado un

estudio correctamente, a partir de una pregunta de investigación, con una hipótesis, objetivos, diseño epidemiológico, variables, etc., y sobre todo, hay que tener muy claros los objetivos a resolver. Los objetivos son parte del diseño del estudio y deben ser claramente establecidos antes de analizar los datos.

He visto muchos casos de investigadores no-estadísticos que empiezan a darles vueltas a los datos "a ver qué sale" y, dependiendo de lo que vaya saliendo en el análisis, van planteando otros objetivos hasta que llegan a "algo" que les gusta. El llamado proceso de *salir a pescar*. Esto es un grave error metodológico, y las conclusiones pueden muchas veces ser erróneas. El principal motivo es el problema de multiplicidad estadística, donde al realizar muchos test, el error de tipo I aumenta y empiezan a encontrarse efectos significativos que son solos debidos al azar [2]. La multiplicidad es la gran desconocida en investigadores de Ciencias de la Salud con poca formación estadística. El segundo motivo de este error metodológico es la imposibilidad de replicación del estudio. Si vamos probando cosas hasta que el resultado nos gusta (sale significativo), y luego solo mostramos este último resultado, el experimento no es replicable por otros investigadores externos, y estea es una de las principales premisas del método científico.

Métodos

En bioestadística pueden existir distintos caminos para resolver un mismo objetivo, lo importante es que los métodos aplicados sean correctos en cada caso. En este libro se aplican métodos estándar ampliamente utilizados en la literatura estadística clásica, mostrando la referencia del autor original en cada caso.

Hay aspectos estadísticos que aparecen en las guías de presentación de resultados en investigaciones biomédicas, como SAMPL [3], que podrían ser algo discutibles:

Por ejemplo, comprobar previamente hipótesis como normalidad o homogeneidad de varianzas antes de realizar un test de comparación de medias, aumenta el error tipo I global [4].

Establecer previamente el "nivel de significatividad" en 0.05 para "decidir" si un test es significativo o no lo es, es ciertamente una falacia [5]. No hay una frontera a partir de la cual, una investigacion prorporciona evidencia de si un tratamiento es eficaz o no, y lo que es más importante, el investigador no debe "decidir" si es eficaz o no, solo mostrar grados de evidencia, en forma de incertidumbre, sobre la pregunta de investigación planteada.

Dicho esto, en este libro se hna comprobado previamente supuestos como normalidad y homogeneidad de varianzas en contrastes básicos, si bien el lector debe ser prudente y tener en cuenta esos aspectos mencionados.

Nomenclaturas

A lo largo de este estudio, a la *variable dependiente* se la llamará variable respuesta, variable resultado o evento de interés.

Las *variables independientes* se podrán llamar explicativas, predictores o factores. Según la naturaleza de las variables, llamaremos cuantitativas o continuas por un lado, o cualitativas, factores o categóricas por otro lado, tomándolos como términos sinónimos en la práctica.

Conjunto de datos

Todas las funciones de este libro están disponibles en el paquete Quesadilla, que se puede instalar mediante `devtools::install_github("jquesada2000/Quesadilla")`

En este libro se han utilizado distintas bases de datos médicos bien documentados como datos de ejemplo. Las bases de datos utilizadas son gratuitas y abiertas, y están disponibles en paquetes de R.

A continuacion se muetra cada base de datos, con el paquete de R, una descripcion y los nombres de las variables que incluye.

DATASET	PACKAGE
Pima.tr **Pima.te**	**MASS**
Diabetes in Pima Indian Women. A population of women who were at least 21 years old, of Pima Indian heritage and living near Phoenix, Arizona, was tested for diabetes according to World Health Organization criteria. The data were collected by the US National Institute of Diabetes and Digestive and Kidney Diseases. We used the 532 complete records after dropping the (mainly missing) data on serum insulin (Pima.tr contains a randomly selected set of 200 subjects, and Pima.te contains the remaining 332 subjects)	
Variables and values	
• glu: plasma glucose concentration in an oral glucose tolerance test. • bp:diastolic blood pressure (mm Hg). • skin:triceps skin fold thickness (mm). • bmi:body mass index (weight in kg/(height in m)\^2). • ped:diabetes pedigree function. • age:age in years. • type:Yes or No, for diabetic according to WHO criteria. • glu: plasma glucose concentration in an oral glucose tolerance test. • bp:diastolic blood pressure (mm Hg). • skin:triceps skin fold thickness (mm).	

DATASET	PACKAGE
pharmacoSmoking	**asaur**
Randomized trial of triple therapy vs. patch for smoking cessation. 125 patients and 14 variables.	
Variables and values	
• id: patient ID number • ttr: Time in days until relapse • relapse: Indicator of relapse (return to smoking) • grp: Randomly assigned treatment group with levels combination or patchOnly • age: Age in years at time of randomization • gender: Female or Male • race: black, hispanic, white, or other • employment: ft (full-time), pt (part-time), or other • yearsSmoking: Number of years the patient had been a smoker • levelSmoking: heavy or light	

DATASET	PACKAGE
prostate	**genridge**
Data to examine the correlation between the level of prostate-specific antigen (PSA) and a number of clinical measures in men who were about to receive a radical prostatectomy. A dataframe with 97 observations on the following 10 variables.	
Variables and values	
• lcavol: log cancer volume • lweight: log prostate weight • age: in years • lbph: log of the amount of benign prostatic hyperplasia • svi: seminal vesicle invasión (0=No; 1=Yes) • lcp: log of capsular penetration • gleason: Gleason Score: a numeric vector • pgg45: percent of Gleason score 4 or 5 • lpsa: response PSA • train: a logical vector	

DATASET	PACKAGE
covid_testing	**medicaldata**

This data set is from Amrom E. Obstfeld, who de-identified data on COVID-19 testing during the year 2020 at the Children's Hospital of Pennsylvania (aka CHOP). This data set contains data concerning testing for SARS-CoV2 via PCR as well as associated metadata. The data has been anonymized, time-shifted, and permuted.

The COVID-19 pandemic in 2020 required rapid development and deployment of a novel PCR test, and application to many outpatients with symptoms, and all inpatients upon admission to the emergency room or the hospital. These anonymized data (with fake subject_ids, names, ages, and genders) provide a snapshot of the first ~100 days of testing at a single pediatric hospital, with over 15,000 tests performed on patients in a variety of settings. The vast majority were a single type of test. These tests occurred in 88 named clinics (clinic_name) and in 5 demographic groups (demo_group). Roughly half of the tests were performed at drive-through sites (drive_thru_ind).

The test result (result) can be either negative, positive, or invalid. The ct_result provides the cycle number at which the positive threshold was reached during PCR. Lower integer values for ct_result occur with a higher viral load (higher number of viral particles) in the test sample (deep nasal swab).

The test order can have been placed as a one-off order, or within an order set. The payor_group for an ordered test can be one of 7 categories (includes insurance type and self-pay). The patients are categorized into 9 patient classes, including inpatient, emergency, outpatient, etc. The col_rec_tat variable records the time elapsed (in hours) between sample collection and receipt in the lab. The rec__ver_tat variable records the time elapsed (in hours) between sample receipt in the lab and result verification/posting.

Variables and values

variable	Variable Label	units	Codes
subject_id	Subject ID		
fake_first_name	Pseudo First Name		
fake_last_name	Pseudo Last Name		
gender	Gender (Anonymized)		
pan_day	Day after start of pandemic in which specimen was collected (Anonymized)	Day	
test_id	Test that was performed		
clinic_name	Clinic or ward where the specimen was collected		
result	Result of the test		

• demo_group	Type of subject		
• age	Age of subject at time of specimen collection (Anonymized)	Years	
• drive_thru_ind	Whether the specimen was collected via a drive-thru site		1: Collected at drive-thru site; 0: Not collected at drive-thru site
• ct_result	Cycle at which threshold reached during PCR		
• orderset	Whether and order set was used for test order		1: Collected via orderset; 0: Not collected via orderset
• payor_group	Payor associate with order		
• patient_class	Disposition of subject at time of collection		
• col_rec_tat	Time elapsed between collect time and receive time	Hours	
• rec_ver_tat	Time elapsed between receive time and verification time	Hours	

DATASET	PACKAGE
AlzheimerDisease	**AppliedPredictiveModeling**

Washington University conducted a clinical study to determine if biological measurements made from cerebrospinal fluid (CSF) can be used to diagnose or predict Alzheimer's disease (Craig-Schapiro et al. 2011). These data are a modified version of the values used for the publication. The R factor vector diagnosis contains the outcome data for 333 of the subjects. The demographic and laboratory results are collected in the dataframe predictors.

Variables and values

- diagnosis: labels for the patients, either "Impaired" or "Control".
- predictors: predictors for demographic data (eg. age, gender), genotype and assay results.

  Explanatory variables of interest in predictors
  - age
  - male: 0=woman, 1=men
  - tau, p_tau, Ab_42: 3 known proteins (numeric)

DATASET	PACKAGE
smartpill	**medicaldata**

Prospective Cohort Study of Intestinal Transit using a SmartPill to Compare Trauma Patients to Healthy Volunteers This study evaluated gastric emptying, small bowel transit time, and total intestinal transit time in 8 critically ill trauma patients. These data were compared with those obtained in 87 healthy volunteers from a separate trial. Data were obtained with a motility capsule that wirelessly transmitted pH, pressure, and temperature to a recorder attached to each subject's abdomen. Transit times were available for almost all patients, however, pH, pressure and temperature data is missing for all critically ill patients and sparsely missing for the healthy volunteers

Variables and values

- `Group` Study group, numeric, 0 = Critically Ill Trama Patient, 1 = Healthy Volunteer
- `Gender` Gender, numeric, range: 0 = Female, 1 = Male
- `Race` Race, numeric, 1 = White, 2 = Black, 3 = Asian/Pacific Islander, 4 = Hispanic, 5 = Other
- `Height` Height (centimeters), numeric, range: 132.1-193.0
- `Weight` Weight (kilograms), numeric, range: 44.9-127.0
- `Age` Age (years), numeric, range: 18.0-72.0
- `GE.Time` Gastric Emptying Time is time from ingestion to gastric emptying (hours), numeric, range: 1.7-74.3
- `SB.Time` Small Bowel Transit Time is time from gastric emptying to ileocecal junction (hours), numeric, range: 1.8-13.8
- `C.Time` Colonic Transit Time is time from ileocecal junction to body exit (hours), numeric, range: 0.7-118.9
- `WG.Time` Whole Gut Time is time from ingestion to body exit (hours), numeric, range: 6.0-816.0
- `S.Contractions` Stomach contractions are counted if the peak amplitude of the contraction is over 10 mmHg and under 300 mmHg, numeric, range: 47.0-1665.0
- `.Sum.of.Amplitudes` Stomach sum of amplitudes (mm Hg), numeric, range: 655.6-33800.3
- `S.Mean.Peak.Amplitude` Stomach mean peak amplitude is the sum of amplitudes divided by number of contractions (mm Hg), numeric, range: 4.6-43.4
- `S.Mean.pH` Stomach mean pH is the average pH over the whole recording time in the stomach with normal ~ 1.5-3.5, numeric, range: 1.5-5.9
- `SB.Contractions` Small Bowel contractions are counted if the peak amplitude of the contraction is over 10 mmHg and under 300 mmHg, numeric, range: 223.0-2375.0
- `SB.Sum.of.Amplitudes` Small Bowel sum of amplitudes (mm Hg), numeric, range:3899.4-41122.5

- `SB.Mean.Peak.Amplitude` Small Bowell mean peak amplitude is the sum of amplitudes divided by number of contractions (mm Hg), numeric, range: 15.0-27.9
- `SB.Mean.pH` Small Bowel mean pH is the average pH over the whole recording time in the small bowel, normal ~ 6-7.4, numeric, range: 4.7-8.6
- `Colon.Contractions` Colon contractions are counted if the peak amplitude of the contraction is over 10 mmHg and under 300 mmHg, numeric, range: 41.0-2672.0
- `Colon.Sum.of.Amplitudes` Colon sum of amplitudes (mm Hg), numeric, range:1872.6-117707.5
- `C.Mean.Peak.Amplitude` Colon mean peak amplitude is the sum of amplitudes divided by number of contractions (mm Hg), numeric, range: 32.8- 64.2
- `C.Mean.pH` Colon mean pH is the average pH over the whole recording time in the colon, normal ~ 5-7-6.7, numeric, range: 3.9-8.1

DATASET	PACKAGE
supraclavicular	**medicaldata**

This data set contains 103 patients who were scheduled to undergo an upper extremity procedure suitable for supraclavicular anesthesia. Patients were randomly assigned to either (1) combined group-ropivacaine and mepivacaine mixture; or (2) sequential group-mepivacaine followed by ropivacaine. A number of demographic and post-op pain medication variables (fentanyl, alfentanil, midazolam) were collected. The primary outcome is time to 4-nerve sensory block onset. The dataset is cleaned and relatively complete. There are no outliers or data problems.

Variables and values

- `subject` Subject ID, numeric, range: 1-103
- `group` Anesthetic group, numeric, 1 = Mixture; 2 = Sequential
- `gender` Gender, numeric, 1 = Male; 0= Female
- `bmi` Body mass index (kg/m^2), numeric, range:19-43.5
- `age` Age (years), numeric, range:18-74
- `fentanyl` Fentanyl pain medication (micrograms), numeric, range: 0-250.0
- `alfentanil` Alfentanil pain medication (milligrams), numeric, range: 0-4.3
- `midazolam` Midazolam hypnotic-sedative medication, numeric, range: 0-9.0
- `onset_sensory` Time to 4 nerve sensory block onset or, if onset_sensory block failed the observed worst outcome of minutes for any patient (50 minutes), numeric, range: 0-50.0
- `onset_first_sensory` Time to first sensory block in minutes, or if block failed, a value of 15 minutes, numeric, range: 6-15.0

- `onset_motor` Time to complete motor block or, if motor block failed, the observed worst outcome of minutes for any patient (50 minutes), numeric, range: 1-50.0
- `nerve_block_censor` block failed, numeric, 0 = nerve block succeeded, 1 = block failed (censored)
- `med_duration` Time from the onset of 4 nerve sensory block until the first request for an analgesic medication (hours), numeric, range: 0-48.0
- `med_censor` Patients who did not take an analgesic were censored at 48 hours, numeric, 0 = nerve succeeded, 1 = block failed (censored)
- `vps_rest` Maximum postop verbal pain score (at rest), on 11 point Likert scale (0-10), numeric, range: 0-10
- `vps_movement` Maximum postop verbal pain score (with movement), on 11 point Likert scale (0- 10), numeric, range: 0-10
- `opioid_total` Total opioid consumption in milligrams, numeric, range: 0-225.0

DATASET	PACKAGE
esoph_ca	**medicaldata**

Data from a case-control study of esophageal cancer in Ille-et-Vilaine, France, evaluating the effects of smoking and alcohol on the incidence of esophageal cancer. Smoking and alcohol are associated risk factors for squamous cell cancer of the esophagus, rather than adenocarcinoma of the esophagus, which is associated with obesity and esophageal reflux (more details available below the variable definitions). A dataframe with 88 rows and 5 variables, with 200 cases and 975 controls.

Variables and values

- `agegp` 6 levels of age: “25-34”, “35-44”, “45-54”, “55-64”, “65-74”, “75+”; type: ordinal factor
- `alcgp` 4 levels of alcohol consumption: “0-39g/day”, “40-79”, “80-119”, “120+”; type: ordinal factor
- `tobgp` 4 levels of tobacco consumption: “0-9g/day”, “10-19”, ’20-29”, “30+”; type: ordinal factor
- `ncases` Number of cases; type: integer
- `ncontrols` Number of controls; type: integer

DATASET	PACKAGE
cancer	**Survival**

Survival in patients with advanced lung cancer from the North Central Cancer Treatment Group. Performance scores rate how well the patient can perform usual daily activities.

Variables and values

- inst: Institution code
- time: Survival time in days
- status: censoring status 1=censored, 2=dead
- age: Age in years
- sex: Male=1 Female=2
- ph.ecog: ECOG performance score as rated by the physician. 0=asymptomatic, 1= symptomatic but completely ambulatory, 2= in bed <50% of the day, 3= in bed > 50% of the day but not bedbound, 4 = bedbound
- ph.karno: Karnofsky performance score (bad=0-good=100) rated by physician
- meal.cal: Calories consumed at meals
- wt.loss: Weight loss in last six months

PARTE I

Bioestadística básica

1

Análisis descriptivo

Suponemos que estamos en un contexto de realizar un análisis de datos sobre una muestra obtenida de una población de referencia, para resolver los objetivos de una investigación en Ciencias de la Salud, que podría ser un trabajo fin de grado, trabajo fin de máster, una tesis doctoral, un artículo científico, un proyecto de investigación, una comunicación a un congreso, etc.

Partimos de la base de que tenemos unos objetivos concretos y escritos para resolver dentro de un *problema supervisado*. Llamamos problema supervisado a aquel en el que disponemos de una (o varias) variables respuesta, y queremos explicar o predecir la respuesta mediante un conjunto de variables explicativas.

Si el diseño de investigación es transversal, hablaremos de asociaciones entre la respuesta y las explicativas. Si es longitudinal y la respuesta es incidente (casos nuevos aparecidos durante el seguimiento), se podría hablar de riesgo de ocurrencia de la respuesta, en función de las explicativas, estando en un contexto causal.

Lectura y preparación de la base de datos con R

Los datos los podemos tener en diferentes formatos, y debemos leerlos desde **R** y guardarlos en un dataframe.

Creamos un script en blanco llamado `descrip_analysis.r` que será nuestro código principal para realizar el análisis descriptivo. En él leeremos los datos, y llamaremos a las funciones que vamos creando.

Desde SPSS

```
library(foreign)

dataset0 <- read.spss("data.sav", use.value.labels=TRUE, max.val-
ue.labels=Inf, to.data.frame=TRUE, use.missings = FALSE)
```

La función `read.spss()` del paquete `foreign` lee el fichero `data.sav` del directorio en curso y lo guarda en un dataframe mediante el argumento `to.data.frame=TRUE`, guardando como factores las variables que tengan etiquetas puestas en las categorías en SPSS, y como numéricas las variables cuantitativas que no sean factores. La base de datos se guarda en el objeto `dataset0`.

Desde archivo tipo Excel

Se puede leer el fichero mediante la función `read.xlsx2` del paquete `xlsx`:

```
library(xlsx)

dataset0 <- read.xlsx2("TAEprov.xls", sheetName='Hoja1', as.data.
frame = TRUE)
```

Desde un archivo Access

```
library(RODBC)

base <- odbcConnectAccess2007("muy_file_name.accdb")

# "my_table" is the name of the table into Access

dataset0 <- sqlFetch(base, "my_table")
```

La función `odbcConnectAccess2007` del paquete `RODBC` lee el archivo `.accdb`, y la función `sqlFetch` lee la tabla `"my_table"` del archivo en formato Access y lo guarda en datos como un dataframe.

Depuración de la base

Nos interesa que la base de datos solo tenga variables cualitativas y cuantitativas para hacer un análisis con programación en **R**, por lo que variables tipo *fecha*, *texto*, etc., deberán no tenerse en cuenta.

Una forma sencilla de hacer esto es crear una sub-base eliminando las variables que no nos interesan.

```
names(dataset0)
```

La función `names()` nos proporciona los nombres y posición de todas las variables de la base `dataset0`. Si, por ejemplo, las tres primeras variables son fechas o texto, las podemos quitar.

```
dataset <- dataset0[, -c(1,2,3)]
```

Guardamos en `dataset` la base de datos original, quitando las variables con posición 1, 2 y 3. De esta forma, la nueva base `dataset` debe solo tener variables cuantitativas o factores.

∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞

Análisis descriptivo gráfico con R

En un análisis de datos, lo primero es echar un vistazo a los datos mediante graficas univariantes. Así nos podemos hacer una idea de la forma y estructura de los datos.

Mediante el siguiente sencillo código se calculan gráficos de barras para las variables cualitativas (factores) mediante `barplot()` e histogramas para las cuantitativas mediante `hist()`.

Pasamos un bucle `for` que recorre las variables y va poniendo los gráficos en ventanas nuevas mediante `win.graph()`.

Debemos poner el nombre de la base de datos en `my_dataset`

```
### gráficos uni ------------------------------------
library(nortest)

# put your dataset name-----
dataset <- my_dataset

vari <- names(dataset)

par(mfrow = c(3,3))

for(j in 1:length(vari)){

   if(is.factor(dataset[,j])) {

      barplot(table(dataset[, j]),ylab="Frencuency", xlab=vari[j],
      main="")

   } else {
       p <- round(lillie.test(dataset[, j])$p.value,3)
       hist(dataset[, j], freq=F, ylab="Frencuency", xlab=vari[j],
       main=paste("Normality p-value = ",p))

   }
}
```

∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞

Ejemplo 1.1. *Análisis descriptivo*

Se pretende describir el nivel de PSA (*prostate specific antigen*) según variables clínicas, en pacientes con prostatectomia radical. Los datos se encuentran en la base `prostate` del paquete `genridge`.

Leemos los datos y calculamos los gráficos con el código anterior.

```
library(genridge)
data("prostate")

> names(prostate)
 [1] "lcavol"  "lweight" "age"     "lbph"    "svi"     "lcp"
"gleason" "pgg45"   "lpsa"    "train"
```

Le echamos un vistazo a los datos. Debemos tener solo variables numéricas o factores. Debemos eliminar variables tipo fecha, texto, etc., o variables identificadoras de pacientes que no tenga sentido realizar un análisis de datos.

```
> head(prostate)
      lcavol  lweight age      lbph svi       lcp gleason pgg45
lpsa train
1 -0.5798185 2.769459  50 -1.386294   0 -1.386294       6     0
-0.4307829  TRUE
2 -0.9942523 3.319626  58 -1.386294   0 -1.386294       6     0
-0.1625189  TRUE
3 -0.5108256 2.691243  74 -1.386294   0 -1.386294       7    20
-0.1625189  TRUE
4 -1.2039728 3.282789  58 -1.386294   0 -1.386294       6     0
-0.1625189  TRUE
5  0.7514161 3.432373  62 -1.386294   0 -1.386294       6     0
0.3715636  TRUE
6 -1.0498221 3.228826  50 -1.386294   0 -1.386294       6     0
0.7654678  TRUE
```

Todas las variables son numéricas. La variable `train` situada en la columna 10 es un indicador de dos grupos para validar la base original, no nos interesa para el análisis y la eliminamos:

```
> dataset <- prostate[,-10]
> names(dataset)
[1] "lcavol"  "lweight" "age"     "lbph"    "svi"     "lcp"
"gleason" "pgg45"   "lpsa"
```

Ahora debemos asegurarnos de que las variables categóricas son factores y, si no lo son, convertirlas a factores. La única candidata (ver descripción de la base en el Capítulo 1) es `svi`.

```
> class(dataset$svi)
[1] "integer"
```

La variable `svi` es numérica entera, la convertimos a factor, y le ponemos las etiquetas a los valores (0=No; 1=Yes)

```
dataset$svi <- as.factor(dataset$svi)
levels(dataset$svi) <- c("No", "Yes")
```

```
> table(dataset$svi)
```

```
No Yes
76 21
```

Ya tenemos la base preparada, ahora pasamos el código de gráficos a la base `dataset`.

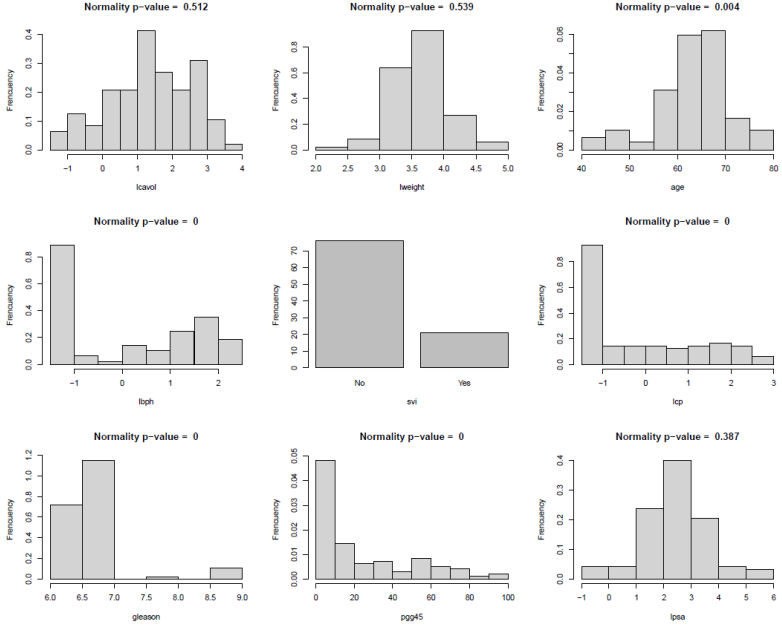

Figura 1.1. Gráficos descriptivos.

En la Figura 1.1 vemos que se ha calculado un histograma para todas las varibles, con el p-valor del test de normalidad puesto en el titulo, excepto para `svi` que era un factor, y ha calcualdo un diagrama de barras.

Las unicas varibles que no rechazamos normalidad son `Icavol`, `Iweight` y `lpsa` y se aprecia su forma acampanada.

∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞

Cálculo de estadísticos descriptivos con R

El primer análisis en una investigación de este tipo es un análisis descriptivo de todas las variables. Dependiendo de la naturaleza de las variables se calcula unos estadísticos u otros. De esta manera, para las variables cualitativas, se suele calcular el número y porcentaje. Para las variables cuantitativas se puede calcular el n valido (si hay missings en alguna variable), el valor mínimo, máximo, medio y desviación estándar para variables aproximadamente Gaussianas. Para distribuciones asimétricas se puede calcular la mediana y el rango intercuartílico (Tabla 1.1).

Tabla 1.1. Estadísticos descriptivos.

Variables cualitativas	
Número	n_a
Porcentaje	$(n_a*100/n)\%$
Variables cuantitativas	
n valido	n without missings
mínimo	$Min(x_i)$, i=1,...n
máximo	$Max(x_i)$, i=1,...n
Media	$\Sigma x_i/n$, i=1,...n
Desviación estándar	$\Sigma(x_i\text{-mean})^2/(n-1)$
Mediana	(n+1)/2
Rango intercuartilico	$x_{(n+1)75/n}$; $x_{(n+1)75/n}$

La idea es crear un código en **R** que lea toda nuestra base de datos, y calcule los estadísticos anteriores dependiendo como sea cada variable, y lo haga todo a la vez.

Para ello crearemos funciones en **R** para las variables cualitativas, y otra función para las cuantitativas, y luego llamaremos a las funciones desde un bucle que lea toda la base. Veamos:

Las funciones `table()` y `prop.table()` de la librería `base` estiman el número y porcentaje de variables cualitativas. Las variables cualitativas se leen en **R** como factores.

Podemos crear esta sencilla función:

```
## descriptive qualitative explanatory variables
descrip.cat <- function ( var ){
   t1 = table ( var )
   categ <- names(t1)
   p1 = prop.table( t1 )
   porc = round ( 100 * p1, dig = 1 )
   data.frame (cat = categ, n = as.vector(t1),p = as.vector(porc)
   )
}
```

La función lee la variable `var`, calcula las frecuencias con `table()` y las guarda en `t1`. Guarda los nombres de las categorías de la variable en `categ`. Calcula las proporciones con `prop.table()` y las guarda en `p1`. Convierte las proporciones a porcentajes, con 1 decimal, y las guarda en `porc`.

Por último, la salida de la función es un dataframe con el nombre de las categorías, la frecuencia y porcentaje de cada categoría.

Para las variables cuantitativas, el proceso es similar. Utilizamos la suma de los valores lógicos que no son *missings*, para calcular el número de casos válidos. Esto lo hacemos con `sum(!is.na(var))`. Los valores mínimo y máximo con `min()` y `max()` respectivamente. La media y la desviación estándar mediante `mean()` y `sd()`. La mediana y el rango intercuartilico mediante `median ()` y `IQR()`.

Con la función `round()` redondeamos a los decimales que queramos, aquí a 1 o a 2. El argumento `na.rm=TRUE` se añade para que el cálculo sea excluyendo a los valores *missings* si los hubiera.

Por último, la salida de la función es un dataframe, con el número válido de casos, mínimo, máximo, media, desviación estándar, mediana y rango intercuartílico.

Al principio de la función podemos comprobar que la variable no se trata de un factor, por lo que es cuantitativa.

```
## descriptive qualitative explanatory variables
descrip.con <- function ( var ) {
   if ( is.factor(var) == TRUE )
      { stop ( "Error: The variable is a factor" ) }

   nvalido = sum(!is.na(var))
```

```
   minimo = round(min(var, na.rm=TRUE),1)
   maximo = round(max(var, na.rm=TRUE),1)
   mean = round(mean (var, na.rm=TRUE),2)
   median = round(median(var, na.rm=TRUE),2)

   sd = round(sd (var, na.rm=TRUE),2)
   iqr = round(IQR (var, na.rm=TRUE),2)

   data.frame ( nvalid=nvalido, Min=minimo, Max=maxi-
   mo,Mean=mean,SD=sd, Median=median,IQR=iqr)
}
```

Las dos funciones `descrip.cat` y `descrip.con` las podemos guardar en un archivo con nombre `descrip_fun.r`, copiando y pegando el código tal cual, una debajo de la otra.

Ahora que ya hemos creado y guardado las funciones, volvemos al script principal del análisis, `descrip_analysis.r`, donde hemos leído los datos y hemos quitado las variables que no son cuantitativas o factores. Podemos leer las funciones creadas mediante la sentencia:

```
source("descrip_fun.r")
```

Ahora ya tenemos los datos y las funciones cargadas. Vamos a calcular los estadísticos descriptivos de la base de datos. Debemos poner el nombre de la base de datos en `my_dataset`.

```
library(plyr)

# put your dataset name-----
dataset <- my_dataset

vari <- names(dataset)
out <- out2 <- NULL
for( name.var in vari){

   if(is.factor(dataset [,name.var])) {
   out[[name.var]] <- descrip.cat(dataset[, name.var])
   } else {

      out2[[name.var]] <- descrip.con(dataset[, name.var])
   }
}
out.cat <- ldply (out, data.frame)
out.con <- ldply (out2, data.frame)
```

Este sencillo código hace lo siguiente:

- Guarda en vari los nombres de las variables de la base.

- Crea dos objetos en blanco tipo *list* llamados out y out2.
- Hace un bucle for para cada variable de la base.
- Dentro del bucle, si la variable *i-ésima* es un factor, le aplica la función descrip.cat, y guarda la salida en el objeto out. El objeto out tendrá tantos componentes como número de variables de la base.
- Si la variable no es un factor, aplica la función descrip.con a la variable, y análogamente lo guarda en out2.
- Cuando termina el bucle, convertimos los objetos out y out2 en dataframes mediante la función ldply, donde cada fila será una variable de la base, y las columnas serán los estadísticos descriptivos calculados con las funciones.
- Se crean dos dataframes, out.cat y out.con para las variables factores y variables cuantitativas, respectivamente.

Solo falta una cosa: el código anterior añade los nombres de la lista `out` al dataframe `out.cat`, pero los repite para cada variable. Para eliminar los nombres de las variables repetidos aplicamos el siguiente código:

```
# We remove repeated names from the variables and reorder the
data.frame
for(i in 2:length(out.cat$.id)){

   out.cat$var[1] <- out.cat$.id[1]

   if( out.cat$.id[i] == out.cat$.id[i-1]) {

      out.cat$var[i] <- ""

    } else {

      out.cat$var[i] <- out.cat$.id[i]

    }
}
```

Eliminamos la columna de valores repetidos y reordenamos:

```
out.cat2 <- out.cat[,-1]
out.cat2 <- out.cat2[,c(4,1,2,3)]
```

Esto no hay que hacerlo con `out.con`, ya que el dataframe de las variables continuas contiene solo una fila por cada variable, y nos nombres no se repiten.

Guardar los resultados en formato Word desde R

Ahora solo queda guardar estos dataframes en un documento Word. Hay varios paquetes y funciones que pueden realizar informes de resultados. Una solución sencilla es utilizar el paquete `officer`, que necesita los paquetes `flextable` y `magrittr`.

```
library(officer)
library(flextable)
library(magrittr)
```

La idea es convertir los dataframes en formato `flextable`, aplicándoles una serie de formatos a las tablas como bordes, tamaño y tipo de letra, márgenes, etc., mediante una función que la podemos tunear a nuestro gusto.

La función es esta, que es aconsejable guardarla en un fichero llamado, por ejemplo, `table_style.r`

```
my.table.style = function(x){
std_b = fp_border(color="black")
cuali.t <- regulartable(x)
cuali.t <- set_formatter_type(cuali.t, fmt_double = "%.01f")
cuali.t <- fontsize(cuali.t, size=10)
cuali.t <- font(cuali.t, fontname = "Calibri")
cuali.t <- align(cuali.t, align="left", j=1)
cuali.t <- align(cuali.t, align="left", j=2)
cuali.t <- border_remove(cuali.t)
cuali.t <- hline(cuali.t, border = std_b, part="header")
cuali.t <- hline_top(cuali.t, border = std_b, part="all")
cuali.t <- hline_bottom(cuali.t, border = std_b, part="all")
cuali.t <- width(cuali.t, width = 1.4)
}
```

También se podrían tener dos funciones, una para variables tipo factor y otra para variables cuantitativas, por si nos interesa, por ejemplo, poner más decimales a los descriptivos de las cuantitativas. Se puede consultar la ayuda de `flextable` para modificar estos parámetros.

Cargamos la función con

```
source("table_style.r")
```

Ahora aplicamos este formato a los dataframes de resultados que hemos calculado

```
out.cat.flex <- my.table.style (out.cat)
out.con.flex <- my.table.style (out.con)
```

El siguiente código crea un objeto en formato Word y guarda las dos tablas. Utiliza el paquete `plyr`, donde se van realizando acciones anidadas mediante %>%. Se puede consultar la ayuda de `officer` para más detalles, pero las funciones utilizadas aquí son:

`body_add_par ()`:	añade una línea de texto
`body_add_flextable ()`:	añade una tabla en formato flextable
`body_add_break ()`:	añade un salto de página

Finalmente se guarda en un fichero formato Word llamado `descriptives.docx`

```
library (plyr)

my_doc <- read_docx( )%>%
body_add_par("Factor variables") %>%
body_add_flextable(out.cat.flex)%>%
body_add_par("") %>%
body_add_break() %>%
body_add_par("Quantitative variables" ) %>%
body_add_flextable(out.con.flex)%>%
body_add_par("")
print(my_doc, target = "descriptives.docx")
```

∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞ ∞

Ejemplo 1.2. *Análisis descriptivo*

Siguiendo con la base de datos `prostate` analizada en el Ejemplo 1.1., aplicamos las funciones al objeto `dataset`, donde ya depuramos y convertimos a factor la variable `svi`.

El resultado se guardar en un Word llamado `descriptives.docx` con el resultado mostrado en la Tabla 1.2:

Tabla 1.2. Resultados del Ejemplo 1.2.

Factor variables

.id	cat	n	p
svi	No	76	78.4
	Yes	21	21.6

Quantitative variables

.id	nvalid	Min	Max	Mean	SD	Median	IQR
lcavol	97	-1.3	3.8	1.4	1.2	1.4	1.6
lweight	97	2.4	4.8	3.6	0.4	3.6	0.5
age	97	41.0	79.0	63.9	7.4	65.0	8.0
lbph	97	-1.4	2.3	0.1	1.4	0.3	2.9
lcp	97	-1.4	2.9	-0.2	1.4	-0.8	2.6
gleason	97	6.0	9.0	6.8	0.7	7.0	1.0
pgg45	97	0.0	100.0	24.4	28.2	15.0	40.0
lpsa	97	-0.4	5.6	2.5	1.1	2.6	1.3

Hay un total de 97 varones, con edad media de 65 años (rango 41 a 79), de los cuales al 21.6% se les aplicó invasión de vesícula (`svi`). El nivel medio de PSA fue de 2.5 con un rango entre -0.4 y 5.6.