Introducción al *machine learning* con R

PID_00293469

Marta Casals Fontanet
Alícia Vila Grifo

Tiempo mínimo de dedicación recomendado: 7 horas

Universitat Oberta de Catalunya

Marta Casals Fontanet

Estadística de formación y profesión. Licenciada en ITM. Máster en Educación y nuevas tecnologías. Profesora de Matemáticas en secundaria y universidad. Profesora colaboradora en el máster de Bioinformática y bioestadística de la UOC y la UB.

Alicia Vila Grifo

Licenciada en Matemáticas por la Universidad de Valencia y profesora consultora de Probabilidad, Estadística y Análisis de Datos en diferentes estudios de la UOC. Actualmente es profesora funcionaria de Informática en el Departamento de Educación de la Generalitat de Cataluña, en los ámbitos de programación y bases de datos. También participa en la coordinación de proyectos de aplicaciones web y de análisis de datos.

El encargo y la creación de este recurso de aprendizaje UOC han sido coordinados por los profesores: Álvaro Leitao Rodríguez y David Cantón Fabà

Cómo citar este recurso de aprendizaje con el estilo Harvard:

Casals, M. y Vila A. (2024). *Introducción al machine learning R.* [Recurso de aprendizaje textual]. 1.a ed. Fundació Universitat Oberta de Catalunya (FUOC).

Primera edición: febrero 2024

Av. Tibidabo, 39-43, 08035 Barcelona
Autoría: Marta Casals Fontanet, Alícia Vila Grifo
Producción: FUOC

Índice

1. Introducción al LAB5

El **LAB5** es un recurso didáctico complementario de la asignatura **Software para el análisis de datos (SAD)** del máster interuniversitario de Bioinformática y Bioestadística de la Universitat Oberta de Catalunya (UOC) y la Universidad de Barcelona (UB).

Este **LAB5** forma parte de un conjunto de laboratorios prácticos que conjugan contenidos teóricos con ejemplos, ejercicios y casos prácticos reales del ámbito de conocimiento del máster.

El **LAB5** explica los conceptos básicos del aprendizaje automatizado (*machine learning*) con el lenguaje **R** y su aplicación al ámbito biosanitario.

2. Instrucciones del LAB5

El **LAB5** contiene una serie de apartados con introducciones teóricas, ejemplos y casos prácticos, además de otros ejercicios que se propondrán para que sean resueltos por el estudiante. La temporización y las pautas para trabajar el **LAB5** serán indicadas en el aula de la asignatura. Del **LAB5** se presentará una propuesta de solución orientativa que servirá para que el estudiante pueda autoevaluar las soluciones realizadas por él mismo.

En ocasiones, para realizar los ejercicios, se utilizarán conjuntos de datos de paquetes propios de **RStudio**, como es el caso de *melanoma*, *ovarian* y *birthwt*, correspondientes a diversos paquetes propios de **R**.

Todos los ejercicios se realizarán en el entorno de desarrollo integrado para **R**, **RStudio** (https://cran.rstudio.com/), como lenguaje de programación para la informática estadística y los gráficos.

3. Contenidos del LAB5

En este **LAB5** trabajaremos los siguientes contenidos:

- Conceptos básicos del aprendizaje automatizado (*machine learning*).
- Aprendizaje supervisado con **R**.
- Aprendizaje no supervisado con **R**.
- Aprendizaje profundo (*deep learning*) con **R**.
- Introducción a ANOVA.
- Ejercicios y casos prácticos con **R**.
- Soluciones ejercicios y casos prácticos con **R**.

4. Conceptos básicos del aprendizaje automatizado (*machine learning*)

El trabajo de análisis de datos contiene procedimientos comunes que corresponden a procesar, limpiar, transformar y modelar datos con un determinado objetivo y con criterios que nos permitan tomar decisiones. Para ello, se utilizan diversas herramientas estadísticas y técnicas de visualización de datos que ayuden a definir patrones y modelos. Un nivel superior de gestión y análisis es el aprendizaje automático como disciplina de la inteligencia artificial centrada en el desarrollo de modelos que aprendan a partir de los datos y sean capaces de realizar predicciones o patrones, y automaticen tareas específicas.

En general, las cuatro etapas principales que pueden establecerse en un proceso de aprendizaje automático son:

4.1. Recopilación de datos

En esta etapa, se obtienen los datos que se utilizarán para entrenar y evaluar el modelo de *machine learning*. Los datos pueden provenir de diversas fuentes, como bases de datos o archivos de diferentes formatos. Es fundamental la elección de los datos, ya que solo la calidad de estos garantiza la rigurosidad y fiabilidad del estudio.

4.2. Preprocesamiento de datos

A partir de la colección de datos (*dataset*) elegida, es necesario realizar una serie de tareas de preprocesamiento para prepararlos para su uso en el modelo de *machine learning*. Para ello, es necesario lo que se conoce como **limpieza de datos**, que sirve para identificar errores, valores nulos o atípicos y valores no relevantes para el estudio. A continuación, es necesario **preparar las variables** para que, si es necesario, se realicen conversiones de tipo y también **organicen o dividan** los datos en lo que se llaman conjuntos de entrenamiento y prueba para evaluar los modelos que se definan.

4.3. Obtención del modelo de datos

En esta etapa, se selecciona y se entrena un modelo de *machine learning* utilizando el conjunto de datos seleccionados. La elección del modelo dependerá del tipo de problema que se esté trabajando (regresión, *clustering*, etc.) y debería ajustarse al conjunto de datos de entrenamiento con el objetivo de definir patrones. Los modelos de *machine learning* pueden variar desde algoritmos simples, como regresión lineal, hasta modelos más complejos, como redes neuronales.

4.4. Evaluación del modelo de datos

Una vez que el modelo está entrenado, es necesario evaluar su rendimiento utilizando otro conjunto de datos (conjunto de prueba). La evaluación se realiza empleando métricas adecuadas al objetivo del estudio.

El proceso general de *machine learning* es iterativo y es habitual volver a una etapa anterior para realizar los ajustes que sean necesarios. Por otra parte, el objetivo final del proceso de *machine learning* es desarrollar un modelo que pueda hacer predicciones o tomar decisiones

basadas en los datos. La elección de las técnicas y herramientas específicas en cada etapa dependerá de la naturaleza del problema y de los datos disponibles.

Ejemplo 1

Supongamos que queremos definir un modelo de aprendizaje automatizado para predecir si un paciente tiene diabetes o no a partir de un *dataset* de datos médicos. El proceso de *machine learning* para la predicción de diabetes podría contemplar las siguientes etapas:

En la etapa de la recopilación de datos, buscaríamos un conjunto de datos médicos de pacientes que incluyan características como edad, género, nivel de glucosa en sangre, presión arterial, índice de masa corporal (IMC), historial familiar de diabetes, diagnóstico actual de diabetes, etc. Es interesante también valorar la fecha de obtención del conjunto de datos y la valoración de estudios anteriores realizados. En la etapa del preprocesamiento de datos, se realiza una revisión y limpieza de datos para eliminar los que no sean relevantes para el estudio, valores atípicos, errores, etc. Es necesaria también una valoración de los tipos de variables por si, en algún momento, es necesario hacer un cambio de tipos; por ejemplo, si tenemos una variable con valores 'Sí' o 'No' en referencia al diagnóstico de diabetes del paciente, podemos determinar etiquetarla con 1 o 0. En la etapa del modelado de datos seleccionaríamos un algoritmo de *machine learning* y los dividiríamos en el *dataset* en un conjunto de datos de entrenamiento y en un conjunto de prueba para entrenar y evaluar el modelo. Habitualmente, los criterios de división del conjunto de datos inicial son aleatorios. Podríamos, por ejemplo, valorar también el equilibrio de las muestras que dividir para que en los dos conjuntos queden equilibradas las muestras de pacientes con o sin diabetes. Finalmente, la etapa de evaluación corresponde a las pruebas de rendimiento del modelo utilizando métricas de clasificación. Por ejemplo, supongamos que el conjunto de datos de prueba contiene características relevantes de niveles de glucosa y presión arterial además de etiquetas de diagnóstico (diabético o no diabético) y las métricas nos ofrecen porcentajes de predicciones correctas del diagnóstico.

Los tipos de aprendizaje automático más relevantes son el **aprendizaje supervisado**, el **no supervisado** y el **profundo**. De todos ellos veremos una breve introducción a continuación.

5. Aprendizaje supervisado con R

El aprendizaje supervisado se centra en definir un modelo que se entrena utilizando un conjunto de datos etiquetados con un determinado criterio. Este sería el conjunto de datos de entrenamiento, cuyo objetivo principal es rastrear las características de las variables de entrada y las variables resultado para que el modelo que se defina pueda hacer predicciones precisas sobre el conjunto de datos no etiquetados, que corresponderían al conjunto de datos de prueba. Es decir, el aprendizaje supervisado utiliza un conjunto de datos de entrenamiento para enseñar a los modelos a generar la salida deseada y, simultáneamente, dichos modelos van aprendiendo con el tiempo, de manera que el algoritmo correspondiente al modelo utilice métricas que midan la exactitud de ajuste y permita minimizar los errores.

Los conceptos clave del aprendizaje supervisado son:

- **Seleccionar un conjunto de datos etiquetado** que corresponden al conjunto de datos de entrenamiento, y que contienen datos de entrada asociados a datos de salida que representan el resultado esperado. Estas etiquetas sirven como supervisión para el modelo durante el proceso de entrenamiento.
- **Definir cuál es el objetivo de predicción**. El objetivo principal es entrenar un modelo para que pueda predecir de manera precisa las etiquetas de salida de nuevos datos de entrada, no utilizados en el conjunto de datos de entrenamiento. El modelo debe aprender a definir patrones y relaciones en los datos para realizar estas predicciones.

Principalmente, existen dos tipos de algoritmos de **aprendizaje supervisado,** que contienen:

- *Regresión*: En este tipo de situación, las etiquetas de salida son valores numéricos o continuos y tienen el objetivo de predecir un valor numérico específico, por ejemplo, el precio de un tratamiento médico.
- *Clasificación*: En este tipo de situación, las etiquetas de salida son categorías o clases discretas. El objetivo es asignar una categoría a cada valor de entrada, por ejemplo, cómo predecir si un paciente es diabético o no.

Así pues, los **modelos de aprendizaje supervisado** más utilizados son la regresión lineal, la regresión logística o lo que se llama máquinas de vectores de soporte (SVM), entre muchos otros. La elección del modelo depende del tipo de problema y de los datos disponibles.

Como hemos comentado, el **entrenamiento del modelo** se ajusta utilizando el conjunto de datos etiquetado. El modelo utiliza algoritmos y técnicas para aprender la relación entre las características de entrada y las etiquetas de salida. El objetivo es minimizar la diferencia entre las predicciones del modelo y las etiquetas reales en el conjunto de entrenamiento. La **evaluación del modelo** se realiza utilizando un conjunto de datos de prueba que no se utilizó durante el entrenamiento y, para ello, se utilizan métricas de evaluación apropiadas, como el error cuadrático medio (RMSE) en regresión o la aplicación de curva ROC (*Receiver Operating Characteristic*). A lo largo del **LAB** veremos algún ejemplo de ello. Una vez que el modelo está entrenado y es evaluado, se puede utilizar para hacer **predicciones** sobre nuevos datos. Esto es útil en una amplia variedad de aplicaciones, como diagnósticos médicos, recomendación de tratamientos, etc.

La clave del aprendizaje supervisado es que permite que los modelos aprendan patrones a partir de datos existentes y hagan predicciones útiles en situaciones del mundo real. Veamos a continuación algunos ejemplos utilizando técnicas de aprendizaje automatizado mediante **R**.

Ejemplo 2. Análisis de regresión

La idea central de realizar un análisis de regresión es estimar, con técnicas estadísticas, las relaciones existentes entre variables con el fin de realizar predicciones de comportamientos futuros. Para trabajar los conceptos de regresión lineal, podemos utilizar el conjunto de datos *birthwt* del paquete *MASS* de **R**, que hacen referencia a los factores asociados al bajo

peso en el nacimiento de bebés. A continuación, conozcamos la estructura del conjunto de datos:

```
##Visualizamos el conjunto de datos birthwt
library(MASS)
head(birthwt)
```

```
##    low age lwt race smoke ptl ht ui ftv  bwt
## 85   0  19 182    2     0   0  0  1   0 2523
## 86   0  33 155    3     0   0  0  0   3 2551
## 87   0  20 105    1     1   0  0  0   1 2557
## 88   0  21 108    1     1   0  0  1   2 2594
## 89   0  18 107    1     1   0  0  1   0 2600
## 91   0  21 124    3     0   0  0  0   0 2622
```

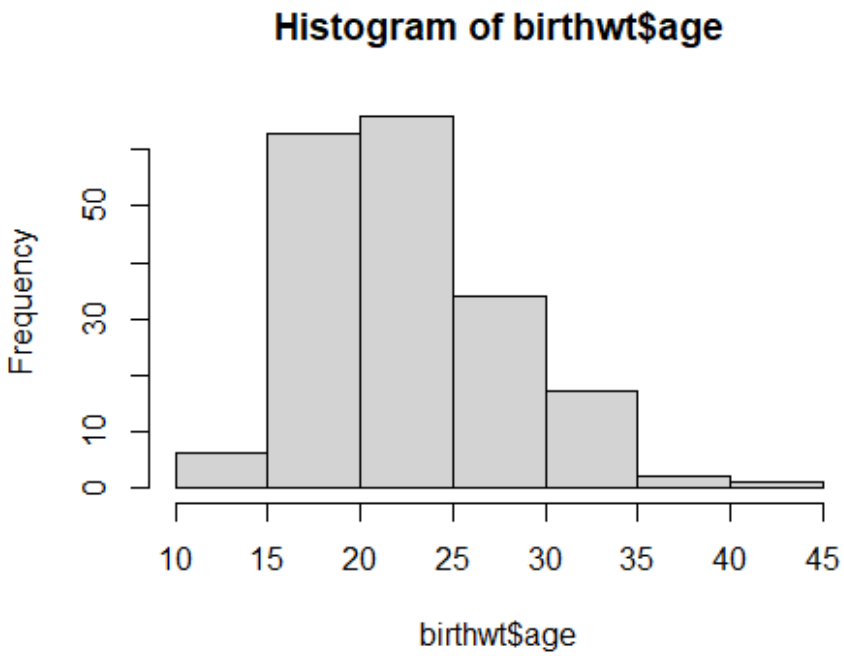
Veamos un resumen estadístico:

```
#Resumen de los parámetros estadísticos más destacados
summary(birthwt)
```

```
##       low              age              lwt             race      
##  Min.   :0.0000   Min.   :14.00   Min.   : 80.0   Min.   :1.000  
##  1st Qu.:0.0000   1st Qu.:19.00   1st Qu.:110.0   1st Qu.:1.000  
##  Median :0.0000   Median :23.00   Median :121.0   Median :1.000  
##  Mean   :0.3122   Mean   :23.24   Mean   :129.8   Mean   :1.847  
##  3rd Qu.:1.0000   3rd Qu.:26.00   3rd Qu.:140.0   3rd Qu.:3.000  
##  Max.   :1.0000   Max.   :45.00   Max.   :250.0   Max.   :3.000  
##      smoke            ptl               ht                ui        
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.00000   Min.   :0.0000  
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.0000  
##  Median :0.0000   Median :0.0000   Median :0.00000   Median :0.0000  
##  Mean   :0.3915   Mean   :0.1958   Mean   :0.06349   Mean   :0.1481  
##  3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.:0.0000  
##  Max.   :1.0000   Max.   :3.0000   Max.   :1.00000   Max.   :1.0000  
##       ftv              bwt        
##  Min.   :0.0000   Min.   : 709  
##  1st Qu.:0.0000   1st Qu.:2414  
##  Median :0.0000   Median :2977  
##  Mean   :0.7937   Mean   :2945  
##  3rd Qu.:1.0000   3rd Qu.:3487  
##  Max.   :6.0000   Max.   :4990
```

Veamos ahora una representación gráfica de, por ejemplo, las edades:

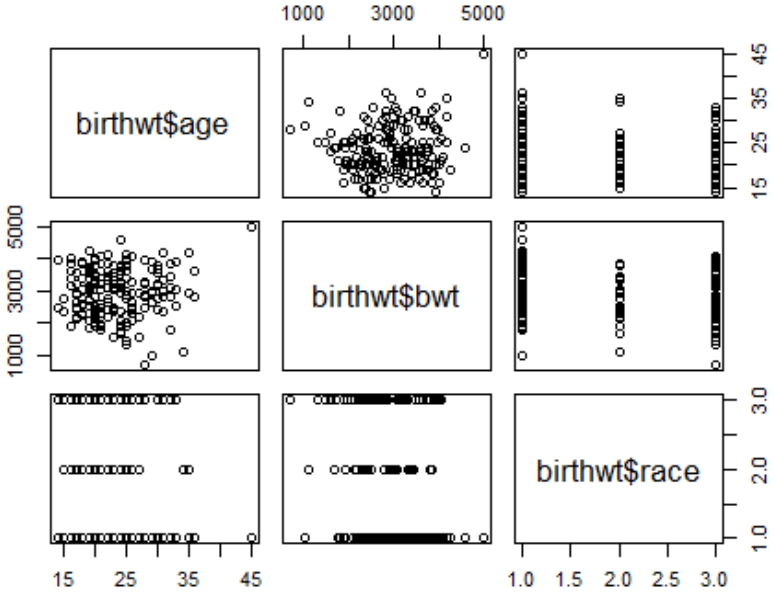
```
with(birthwt,
hist(birthwt$age),scale="frequency",breaks="Sturges",col="tomato")
```



A simple vista, podemos observar que el rango de edades más frecuente está entre 15 y 25 años. De forma análoga podríamos obtener un histograma del resto de variables del conjunto de datos.

Otro tipo de gráfico que podemos realizar es:

```
pairs(~birthwt$age+birthwt$bwt+birthwt$race)
```



Aunque no se visualiza toda la información correctamente, parece ser que sí podría haber una relación entre la edad y el peso de nacimiento y no parece, a simple vista, que la raza de la madre sea un factor importante.

Para visualizar el coeficiente de correlación de ambas variables, realizamos la *matriz de correlación*:

```
cor(birthwt[,c("age","bwt")], use="complete")
```

```
##            age        bwt
## age 1.00000000 0.09031781
## bwt 0.09031781 1.00000000
```

El coeficiente de correlación es 0.09, un valor muy bajo y alejado del 1, lo cual indica que, según los datos tratados, no existe una relación entre la edad de la madre y el peso de los recién nacidos.

Para generar el *modelo lineal de regresión* es necesario estimar los parámetros, lo que se denomina el *método de estimación por mínimos cuadrados*:

```
RegModel.2 <- lm(age~bwt, data=birthwt)
summary(RegModel.2)
```

```
##
## Call:
## lm(formula = age ~ bwt, data = birthwt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.8920 -3.9614 -0.7409  3.1685 20.4196
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.131e+01  1.605e+00   13.27   <2e-16 ***
## bwt         6.563e-04  5.292e-04    1.24    0.216
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.291 on 187 degrees of freedom
## Multiple R-squared:  0.008157,   Adjusted R-squared:  0.002853
## F-statistic: 1.538 on 1 and 187 DF,  p-value: 0.2165
```

```
lm(formula = age ~ bwt, data = birthwt)
```

```
##
## Call:
## lm(formula = age ~ bwt, data = birthwt)
##
## Coefficients:
## (Intercept)          bwt
##   2.131e+01    6.563e-04
```

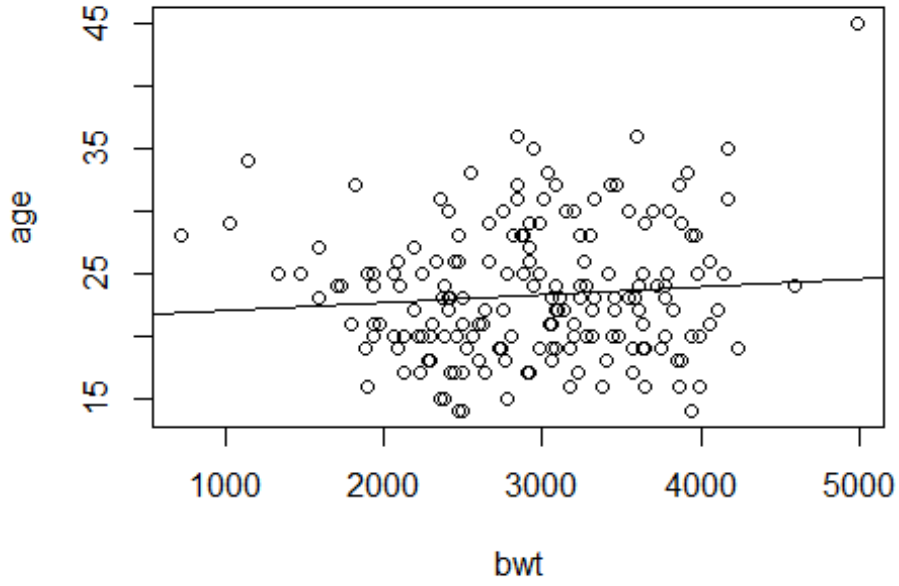
El comando básico en regresión es lm (modelos lineales). El primer argumento de este comando es una ecuación, Y = aX + b, donde Y es una variable dependiente y X es la variable independiente. El segundo argumento especifica lo que se llama archivo de datos, en el que se encuentran las variables y el resultado se almacena en un objeto llamado regresión, que es una lista que contiene toda la información relevante sobre el análisis.

La ecuación de la recta de mínimos cuadrados es: Y = 0.0007X + 21.31.

El *coeficiente de determinación*, es decir, el coeficiente de correlación al cuadrado, mide la bondad de ajuste de la línea a los datos. En nuestro caso, *R-Squared* es 0.003.

Los siguientes comandos representan la nube de puntos y contienen la *recta de mínimos cuadrados*:

```
plot(birthwt$bwt,birthwt$age, xlab="bwt", ylab="age")
abline(RegModel.2)
```



Con todos los datos analizados, no se detecta una correlación entre la edad de la madre y el peso del recién nacido. Una vez estimados los parámetros y generado el modelo, se pueden realizar inferencias y predicciones. Uno de los objetivos de la regresión es predecir el valor de la respuesta Y (variable dependiente) en un individuo de la población de estudio a partir del cual se conoce X (variable independiente). En el ejemplo anteriormente trabajado, a partir de una variable X, que sería la edad, se pretendía comprobar si esta variable tenía una influencia en el valor de la variable Y, que sería el peso. Con los datos trabajados no se tienen suficientes elementos para demostrar que es así.

Ejemplo 3. Regresión lineal múltiple

La idea de la *regresión lineal múltiple* se basa en definir un modelo lineal, de manera que la variable Y resultante (variable dependiente) se determina a partir de diversas variables independientes que se llamarán predictores.

Por ejemplo, siguiendo con el caso anterior del conjunto de datos *birthwt*, queremos comprobar si dicho peso puede venir determinado por diversos factores. Para ello, inicialmente calculamos el coeficiente de correlación de Pearson para las distintas combinaciones de variables, y realizamos la *matriz de correlación*:

```
cor(x=birthwt,method="pearson")
```

```
##                low         age         lwt         race       smoke
ptl
## low    1.00000000 -0.11893928 -0.16962694  0.137792751  0.16140431
0.196087267
## age   -0.11893928  1.00000000  0.18007315 -0.172817953 -0.04434618
0.071606386
## lwt   -0.16962694  0.18007315  1.00000000 -0.165048544 -0.04417908
-0.140029003
## race   0.13779275 -0.17281795 -0.16504854  1.000000000 -0.33903074
0.007951293
## smoke  0.16140431 -0.04434618 -0.04417908 -0.339030745  1.00000000
0.187557063
## ptl    0.19608727  0.07160639 -0.14002900  0.007951293  0.18755706
1.000000000
## ht     0.15237025 -0.01583700  0.23636040  0.019929917  0.01340704
-0.015399579
## ui     0.16904283 -0.07515558 -0.15276317  0.053602088  0.06215900
0.227585340
## ftv   -0.06296026  0.21539394  0.14052746 -0.098336254 -0.02801314
-0.044429660
## bwt   -0.78480516  0.09031781  0.18573328 -0.194713487 -0.19044806
-0.154653390
##                 ht          ui         ftv         bwt
## low    0.15237025  0.16904283 -0.06296026 -0.78480516
## age   -0.01583700 -0.07515558  0.21539394  0.09031781
## lwt    0.23636040 -0.15276317  0.14052746  0.18573328
## race   0.01992992  0.05360209 -0.09833625 -0.19471349
## smoke  0.01340704  0.06215900 -0.02801314 -0.19044806
## ptl   -0.01539958  0.22758534 -0.04442966 -0.15465339
## ht     1.00000000 -0.10858506 -0.07237255 -0.14598189
## ui    -0.10858506  1.00000000 -0.05952341 -0.28392741
## ftv   -0.07237255 -0.05952341  1.00000000  0.05831777
## bwt   -0.14598189 -0.28392741  0.05831777  1.00000000
```

De los datos anteriores extraídos, podemos razonar que la variable que parece presentar mayor correlación lineal con *bwt* es *low*, que corresponde al indicador de peso inferior a 2.5 kg. A continuación, generamos el *modelo de regresión múltiple*:

```
mrm_birthwt<-
lm(birthwt$bwt~birthwt$low+birthwt$age+birthwt$lwt+birthwt$race+birthwt$smoke
+birthwt$ptl+birthwt$ht+birthwt$ui+birthwt$ftv,data=birthwt)
summary(mrm_birthwt)
```

```
##
## Call:
## lm(formula = birthwt$bwt ~ birthwt$low + birthwt$age + birthwt$lwt +
##     birthwt$race + birthwt$smoke + birthwt$ptl + birthwt$ht +
##     birthwt$ui + birthwt$ftv, data = birthwt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -991.22 -300.96   -5.39  277.74 1637.80
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3612.508    229.457  15.744  < 2e-16 ***
## birthwt$low   -1131.217     73.957 -15.296  < 2e-16 ***
## birthwt$age      -6.245      6.347  -0.984 0.326416
## birthwt$lwt       1.051      1.133   0.927 0.355085
## birthwt$race   -100.905     38.544  -2.618 0.009605 **
## birthwt$smoke  -174.116     72.000  -2.418 0.016597 *
## birthwt$ptl      81.340     68.552   1.187 0.236980
## birthwt$ht     -181.955    137.661  -1.322 0.187934
## birthwt$ui     -336.776     93.314  -3.609 0.000399 ***
## birthwt$ftv      -7.578     30.992  -0.245 0.807118
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 433.7 on 179 degrees of freedom
## Multiple R-squared:  0.6632, Adjusted R-squared:  0.6462
## F-statistic: 39.16 on 9 and 179 DF, p-value: < 2.2e-16
```

El modelo anterior generado, una vez que hemos supuesto todas las variables como predictores, resulta un coeficiente de determinación *Multiple R-squared* bastante alto (0.6632) y el p-value también es significativo. Ahora, de entre todos los predictores posibles, deberíamos seleccionar los mejores. Para ello, hacemos lo siguiente:

```
step(object=mrm_birthwt,direction ="both", trace=1)
```

```
## Start:  AIC=2305.09
## birthwt$bwt ~ birthwt$low + birthwt$age + birthwt$lwt + birthwt$race +
##     birthwt$smoke + birthwt$ptl + birthwt$ht + birthwt$ui + birthwt$ftv
##
##                 Df Sum of Sq      RSS    AIC
## - birthwt$ftv    1     11246 33682899 2303.2
## - birthwt$lwt    1    161709 33833361 2304.0
## - birthwt$age    1    182159 33853812 2304.1
## - birthwt$ptl    1    264837 33936490 2304.6
## - birthwt$ht     1    328638 34000291 2304.9
## <none>                       33671653 2305.1
## - birthwt$smoke  1   1100074 34771727 2309.2
## - birthwt$race   1   1289179 34960832 2310.2
## - birthwt$ui     1   2450185 36121838 2316.4
```

```
## - birthwt$low    1  44009294 77680946 2461.1
##
## Step:  AIC=2303.15
## birthwt$bwt ~ birthwt$low + birthwt$age + birthwt$lwt + birthwt$race +
##     birthwt$smoke + birthwt$ptl + birthwt$ht + birthwt$ui
##
##                 Df Sum of Sq      RSS    AIC
## - birthwt$lwt    1    154400 33837299 2302.0
## - birthwt$age    1    205543 33888442 2302.3
## - birthwt$ptl    1    268799 33951698 2302.7
## - birthwt$ht     1    319845 34002744 2302.9
## <none>                       33682899 2303.2
## + birthwt$ftv    1     11246 33671653 2305.1
## - birthwt$smoke  1   1095243 34778142 2307.2
## - birthwt$race   1   1280252 34963151 2308.2
## - birthwt$ui     1   2442561 36125460 2314.4
## - birthwt$low    1  44045331 77728230 2459.2
##
## Step:  AIC=2302.02
## birthwt$bwt ~ birthwt$low + birthwt$age + birthwt$race + birthwt$smoke +
##     birthwt$ptl + birthwt$ht + birthwt$ui
##
##                 Df Sum of Sq      RSS    AIC
## - birthwt$age    1    158960 33996259 2300.9
## - birthwt$ht     1    229750 34067049 2301.3
## - birthwt$ptl    1    232303 34069602 2301.3
## <none>                       33837299 2302.0
## + birthwt$lwt    1    154400 33682899 2303.2
## + birthwt$ftv    1      3937 33833361 2304.0
## - birthwt$smoke  1   1132887 34970186 2306.2
## - birthwt$race   1   1427486 35264785 2307.8
## - birthwt$ui     1   2527757 36365056 2313.6
## - birthwt$low    1  45611646 79448944 2461.3
##
## Step:  AIC=2300.9
## birthwt$bwt ~ birthwt$low + birthwt$race + birthwt$smoke + birthwt$ptl +
##     birthwt$ht + birthwt$ui
##
##                 Df Sum of Sq      RSS    AIC
## - birthwt$ptl    1    189792 34186051 2299.9
## - birthwt$ht     1    228505 34224763 2300.2
## <none>                       33996259 2300.9
## + birthwt$age    1    158960 33837299 2302.0
## + birthwt$lwt    1    107817 33888442 2302.3
## + birthwt$ftv    1     19622 33976637 2302.8
## - birthwt$smoke  1   1054854 35051113 2304.7
## - birthwt$race   1   1301334 35297593 2306.0
## - birthwt$ui     1   2450537 36446796 2312.1
## - birthwt$low    1  45476560 79472818 2459.4
##
```

```
## Step:  AIC=2299.95
## birthwt$bwt ~ birthwt$low + birthwt$race + birthwt$smoke + birthwt$ht +
##     birthwt$ui
##
##                 Df Sum of Sq      RSS    AIC
## - birthwt$ht     1    236483 34422534 2299.3
## <none>                       34186051 2299.9
## + birthwt$ptl    1    189792 33996259 2300.9
## + birthwt$age    1    116450 34069602 2301.3
## + birthwt$lwt    1     85589 34100462 2301.5
## + birthwt$ftv    1     21990 34164061 2301.8
## - birthwt$smoke  1    941319 35127371 2303.1
## - birthwt$race   1   1268492 35454543 2304.8
## - birthwt$ui     1   2279585 36465636 2310.2
## - birthwt$low    1  45501243 79687294 2457.9
##
## Step:  AIC=2299.26
## birthwt$bwt ~ birthwt$low + birthwt$race + birthwt$smoke + birthwt$ui
##
##                 Df Sum of Sq      RSS    AIC
## <none>                       34422534 2299.3
## + birthwt$ht     1    236483 34186051 2299.9
## + birthwt$ptl    1    197771 34224763 2300.2
## + birthwt$age    1    114599 34307935 2300.6
## + birthwt$lwt    1     24448 34398086 2301.1
## + birthwt$ftv    1     12780 34409754 2301.2
## - birthwt$smoke  1    935878 35358412 2302.3
## - birthwt$race   1   1269413 35691947 2304.1
## - birthwt$ui     1   2122280 36544815 2308.6
## - birthwt$low    1  48004723 82427257 2462.3

##
## Call:
## lm(formula = birthwt$bwt ~ birthwt$low + birthwt$race + birthwt$smoke +
##     birthwt$ui, data = birthwt)
##
## Coefficients:
##   (Intercept)    birthwt$low   birthwt$race  birthwt$smoke     birthwt$ui
##       3586.50       -1139.20         -97.34        -157.42        -303.19
```

El mejor modelo con los predictores más significativos es:

Call: lm(formula = birthwt*bwt birthwt*low + birthwt*race + *birthwt*smoke + birthwt$ui, data = birthwt) Coefficients: (Intercept) birthwt*lowbirthwt*race birthwts*smokebirthwt*ui 3586.50 -1139.20 -97.34 -157.42 -303.19

Por ello, si ejecutamos nuevamente este modelo:

```
mrm_birthwt_predictores<-(lm(formula = birthwt$bwt ~birthwt$low+birthwt$race
+birthwt$smoke +birthwt$ui, data = birthwt))
summary(mrm_birthwt_predictores)

##
## Call:
## lm(formula = birthwt$bwt ~ birthwt$low + birthwt$race + birthwt$smoke +
##     birthwt$ui, data = birthwt)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1025.8  -351.0    30.8   285.8  1500.8
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3586.50      86.68  41.379  < 2e-16 ***
## birthwt$low   -1139.20      71.12 -16.019  < 2e-16 ***
## birthwt$race    -97.34      37.37  -2.605 0.009942 **
## birthwt$smoke  -157.42      70.38  -2.237 0.026510 *
## birthwt$ui     -303.19      90.02  -3.368 0.000922 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 432.5 on 184 degrees of freedom
## Multiple R-squared:  0.6557, Adjusted R-squared:  0.6482
## F-statistic: 87.59 on 4 and 184 DF,  p-value: < 2.2e-16
```

Los conceptos de regresión lineal y, especialmente, de regresión múltiple son mucho más extensos y complicados. Este apartado solo pretende hacer una pequeña introducción a la realización de modelos para variables de tipo numérico.

Ejemplo 4. Modelo de clasificación

Para ejemplificar un modelo de clasificación, utilizamos el anteriormente mencionado SVM y vemos cómo definimos un conjunto de entrenamiento y uno de prueba, y el modo en que realizamos el proceso completo de modelo supervisado.

A continuación, utilizamos el conjunto de datos Iris, ampliamente utilizado en diferentes estudios, y realizamos una tarea de clasificación de tres clases (setosa, versicolor y virginica) utilizando el algoritmo de máquinas de vectores de soporte (SVM).

```
# Cargamos el conjunto de datos Iris
data(iris)

# Vemos una muestra de los datos
head(iris)

##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
```

```
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa

# Dividimos los datos en conjuntos de entrenamiento y prueba
set.seed(123)
library(caTools) #paquete que contiene algunas funciones útiles
split <- sample.split(iris$Species, SplitRatio = 0.7)
data_train <- iris[split, ] #conjunto de datos de entrenamiento
data_test <- iris[!split, ] #conjunto de datos de prueba

# Entrenamos un modelo de clasificación (SVM)
library(e1071) #libreria que se utiliza para modelos SVM. Otra libreria útil
seria LiblineaR o kernlab
model <- svm(Species ~ ., data = data_train)

# Realizamos predicciones en el conjunto de prueba
predictions <- predict(model, newdata = data_test)

# Calculamos la matriz de confusión y métricas de evaluación
library(caret) #es la libreria más utilizada en métodos predictivos

## Loading required package: ggplot2

## Loading required package: lattice

confusion_matrix <- confusionMatrix(predictions, data_test$Species)
confusion_matrix

## Confusion Matrix and Statistics
##
##             Reference
## Prediction   setosa versicolor virginica
##   setosa         15          0         0
##   versicolor      0         12         1
##   virginica       0          3        14
##
## Overall Statistics
##
##                Accuracy : 0.9111
##                  95% CI : (0.7878, 0.9752)
##     No Information Rate : 0.3333
##     P-Value [Acc > NIR] : 8.467e-16
##
##                   Kappa : 0.8667
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
```

```
##                      Class: setosa Class: versicolor Class: virginica
## Sensitivity                 1.0000            0.8000           0.9333
## Specificity                 1.0000            0.9667           0.9000
## Pos Pred Value              1.0000            0.9231           0.8235
## Neg Pred Value              1.0000            0.9062           0.9643
## Prevalence                  0.3333            0.3333           0.3333
## Detection Rate              0.3333            0.2667           0.3111
## Detection Prevalence        0.3333            0.2889           0.3778
## Balanced Accuracy           1.0000            0.8833           0.9167
```

La precisión del modelo utilizado se centra en la matriz de confusión definida. La precisión se mide como la proporción de predicciones correctas realizadas por el modelo en el conjunto de datos de prueba. Dicha matriz de confusión muestra la cantidad de predicciones correctas (verdaderos positivos y verdaderos negativos) y predicciones incorrectas (falsos positivos y falsos negativos) realizadas por el modelo.

6. Aprendizaje no supervisado con R

El aprendizaje no supervisado es uno de los ámbitos del aprendizaje automático (*machine learning*) en el que partimos de un desconocimiento de los resultados para describir el modelo o patrón que sigue el conjunto de datos. Así, el objetivo de este tipo de aprendizaje es buscar patrones que pueden estar ocultos en los datos sin necesidad de haber definido previamente etiquetas para catalogarlos. Un ejemplo sencillo de aplicación sería el descubrimiento de patrones ocultos en datos de pacientes, lo que puede ser útil para la segmentación de pacientes, la personalización de tratamientos y la identificación de grupos de riesgo. Este tipo de aprendizaje se aplica, básicamente, a grandes y complejos conjuntos de datos en los que, *a priori*, no es posible etiquetar y organizar la estructura de los datos. **R** proporciona una amplia gama de herramientas y bibliotecas para llevar a cabo tareas de aprendizaje no supervisado.

En el aprendizaje no supervisado, hay dos enfoques principales, el ***clustering*** y la **reducción de dimensionalidad**. El ***clustering***, básicamente, consiste en agrupar datos similares en un mismo grupo. Para ello, se ha definido una serie de algoritmos de *clustering* que buscan formar grupos a partir de la semejanza entre los puntos de datos. Una herramienta común es la aplicación del ***clustering k-means*** que implementaremos en **R**. La **reducción de dimensionalidad** se aplica cuando trabajamos con un gran número de datos y queremos simplificarlos, para ello haremos un **análisis de componentes principales (PCA)** con **R** para reducir la dimensionalidad de un conjunto de datos según unos criterios definidos.

Ejemplo 5. Aplicación de aprendizaje no supervisado (*clustering*)

A partir de un conjunto de datos que contiene información de pacientes (género, edad, índice de masa corporal, presión arterial...), *cargamos el conjunto* de datos en el entorno de trabajo correspondiente (en este caso en el entorno de R) y realizamos un análisis en cuanto a valores nulos, atípicos, etc. Una vez *preparados estos datos*, aplicaremos el algoritmo *k-means* para agrupar a los pacientes en diferentes segmentos (clústeres) según sus perfiles de salud.

A partir de aquí determinaremos un número óptimo determinado de clústeres utilizando diferentes métodos existentes (por ejemplo, *elbow method*). Una vez obtenidos los resultados a partir del *algoritmo k-means*, examinaremos los grupos resultantes y analizamos las características de los pacientes en cada clúster, con lo que esperamos obtener patrones como grupos de pacientes con perfiles de salud similares. Así, una vez identificados los clústeres y realizado el seguimiento y evaluación del modelo, podemos personalizar los tratamientos y las recomendaciones para grupos de pacientes con necesidades similares.

Ejemplo 6. Aplicación aprendizaje no supervisado (PCA)

Supongamos que tenemos el conjunto de datos *diabetic* del paquete *Survival* de Rstudio que trabaja datos de pacientes con riesgo de diabetes. Lo que pretendemos es reducir la dimensionalidad del conjunto de datos para tratar de centrar la información relevante. En este ejemplo, para centrar la idea del modelo, nos centraremos solo en las variables de tipo numérico y estandarizaremos los datos:

```
#instalamos el paquete survival
#install.packages("survival")
#cargamos el paquete y el conjunto de datos
library(survival)

##
## Attaching package: 'survival'

## The following object is masked from 'package:caret':
##
##     cluster

data("diabetic")
#mostramos los primeros registros del dataset
head(diabetic)

##   id laser age   eye trt risk  time status
## 1  5 argon  28  left   0    9 46.23      0
## 2  5 argon  28 right   1    9 46.23      0
## 3 14 xenon  12  left   1    8 42.50      0
## 4 14 xenon  12 right   0    6 31.30      1
## 5 16 xenon   9  left   1   11 42.27      0
## 6 16 xenon   9 right   0   11 42.27      0

# Consideramos solo las columnas numéricas y eliminamos alguna información no
relevante
estandarizar_datos_diabetic <- scale(diabetic[,sapply(diabetic,is.numeric)])
head(estandarizar_datos_diabetic)

##             id        age        trt       risk       time     status
## [1,] -1.752093  0.4873237 -0.9987302 -0.4731892  0.4987246 -0.8042944
## [2,] -1.752093  0.4873237  0.9987302 -0.4731892  0.4987246 -0.8042944
## [3,] -1.733930 -0.5928762  0.9987302 -1.1511404  0.3240656 -0.8042944
## [4,] -1.733930 -0.5928762 -0.9987302 -2.5070427 -0.2003798  1.2401701
```

```
## [5,] -1.729894 -0.7954137  0.9987302  0.8827130  0.3132957 -0.8042944
## [6,] -1.729894 -0.7954137 -0.9987302  0.8827130  0.3132957 -0.8042944
```

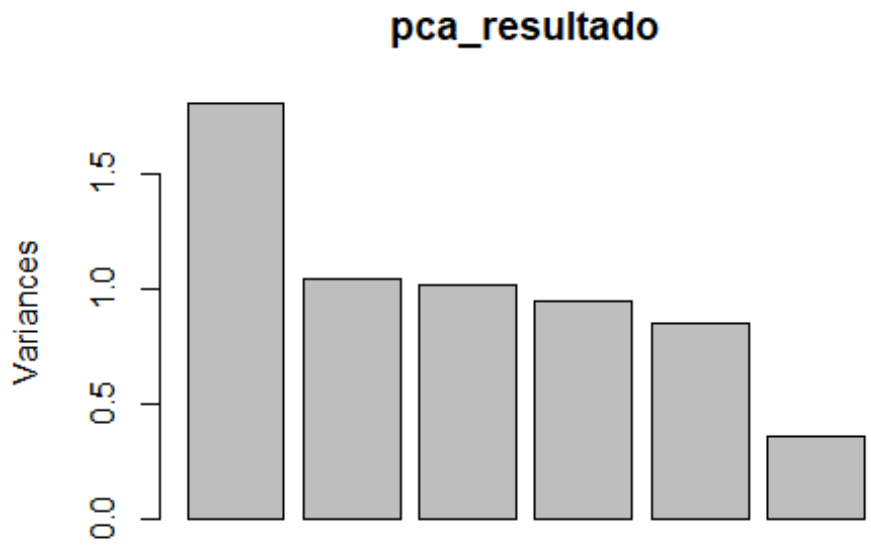
Ahora aplicamos la PCA

```
# Aplicamos PCA
pca_resultado <- prcomp(estandarizar_datos_diabetic, center = TRUE, scale =
TRUE)

# Resumen de la varianza explicada por cada variable
summary(pca_resultado)
```

```
## Importance of components:
##                           PC1    PC2    PC3    PC4    PC5     PC6
## Standard deviation     1.3429 1.0197 1.0053 0.9701 0.9224 0.59531
## Proportion of Variance 0.3005 0.1733 0.1684 0.1569 0.1418 0.05907
## Cumulative Proportion  0.3005 0.4738 0.6423 0.7991 0.9409 1.00000
```

Visualización de los resultados:

```
# Representamos la varianza explicada por cada componente
plot(pca_resultado)
```



Las siguientes instrucciones nos proporcionarán una matriz que muestra los cargos de las variables en cada componente principal, lo que permitirá interpretar la relación entre las variables y los componentes principales. La interpretación de los datos siguientes se hace en función de los valores de los cargos de cada variable en cada PCx. Un valor alto y positivo significa que una variable contribuye positivamente al componente principal, mientras que un valor alto y negativo significa una contribución negativa.

```
cargos <- pca_resultado$rotation
cargos
```

```
##                 PC1         PC2         PC3         PC4        PC5
PC6
## id      0.16678088  0.36431250 -0.77694756 -0.11715576  0.4711662
0.009483838
## age    -0.01816076 -0.67309994 -0.52045065  0.47904368 -0.2114305 -
0.039396480
## trt     0.32876896 -0.04397126  0.34051215  0.60666511  0.6277774
0.109004205
## risk   -0.19618846  0.63781220 -0.08181248  0.61997758 -0.4040944 -
0.018652343
## time    0.63079092  0.04371833 -0.04386911 -0.06480527 -0.3592668
0.681905172
## status -0.65374169 -0.05969680 -0.03028714  0.01331184  0.2164026
0.721895757
```

De los datos anteriores, podríamos decir que, por ejemplo, en PC1 *time* y *status* son las variables más influyentes, en positivo y negativo, respectivamente. En PC2, las variables *age* y *risk* son las variables más influyentes. Un aumento en *age* se asocia con un aumento positivo en PC2, mientras que un aumento en *risk* se asocia con un aumento negativo en PC2. Para el resto de los componentes podríamos realizar un razonamiento parecido. Por lo tanto, los cargos indican cómo cada variable original contribuye a la dirección de cada componente principal. Puedes utilizar esta información para comprender qué características o aspectos de tus datos se relacionan más estrechamente con cada componente principal. En este hipotético caso, PC1 podría estar relacionado con la duración de seguimiento y la identificación del paciente, mientras que PC2 se asocia más con la edad y el riesgo, y así sucesivamente.

6.1. Técnicas de *clustering* con R

Las técnicas de *clustering* consisten en formar grupos, lo más homogéneos posible, a partir de un conjunto de observaciones que tienen características similares con la finalidad de crear patrones que aporten una información específica. Los métodos más utilizados pueden ser de tipo jerárquico o no jerárquico.

Los **métodos jerárquicos** crean grupos buscando el número de clústeres óptimo en cada momento y las técnicas para conseguirlo pueden ser aglomerativas (*bottom up*) o divisivas (*bottom down*), en función de si el punto de partida son elementos individuales, que acaban generando un número determinado de clústeres, o de si el origen es un conjunto de datos. Es decir, en el caso de las técnicas aglomerativas, cada punto de datos se inicia como un clúster individual y, a medida que se avanza en el proceso, los clústeres se combinan para formar clústeres más grandes, es decir, se parte de clústeres muy pequeños y se fusionan gradualmente en clústeres más grandes. En el caso de las técnicas divisivas, todos los datos se consideran un solo clúster en el nivel superior de la jerarquía. Luego, se dividen en clústeres más pequeños a medida que descienden en la jerarquía. Comienza con un clúster grande y se divide en subclústeres más pequeños. Existen diversos paquetes en **R** para trabajar clústeres jerárquicos, por ejemplo, *hclust* o *agnes*, en el caso de técnicas aglomerativas, y *diana*, en el caso de técnicas divisivas. Los gráficos generados se denominan

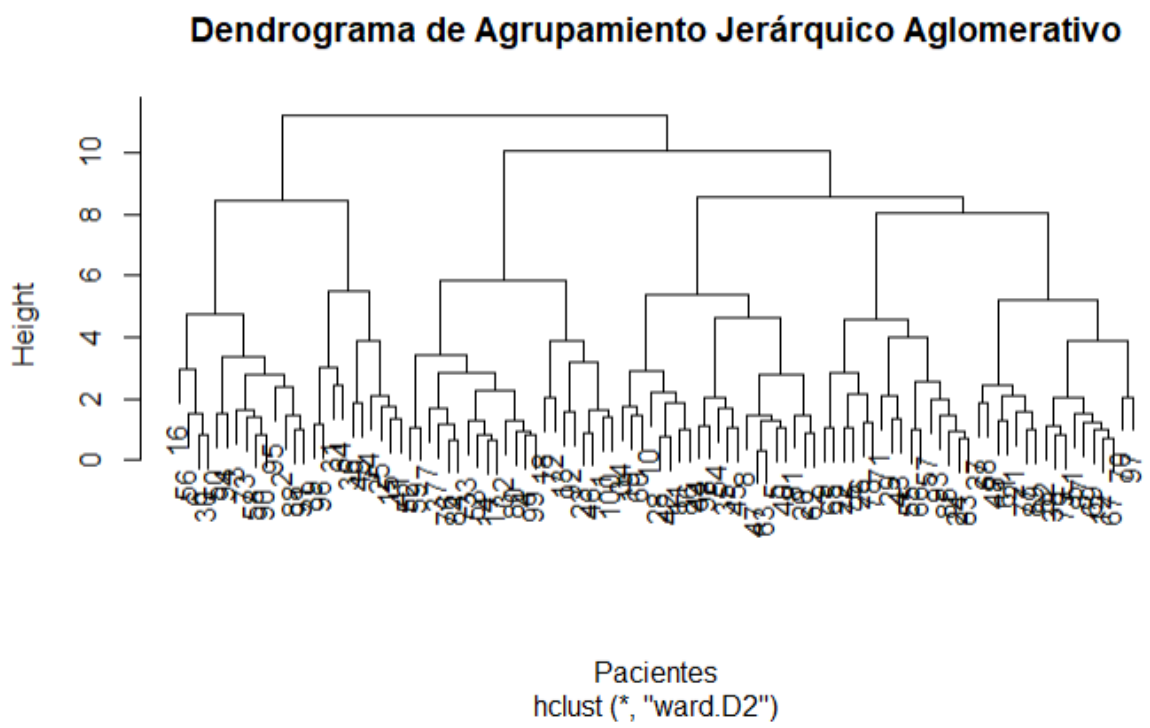
dendrogramas, que son representaciones en forma de árbol en las que cada grupo está vinculado a otros grupos descendientes. Es decir, es un diagrama de árbol que muestra la estructura jerárquica de los clústeres. En un dendrograma, los clústeres individuales se encuentran en la parte inferior y se combinan gradualmente a medida que se sube hacia la parte superior del árbol. No es objetivo de este **LAB5** profundizar en estos conceptos, sino dar a conocer algunas técnicas iniciales por medio de ejemplos.

Las ventajas del agrupamiento jerárquico son que proporcionan una representación visual de la estructura de agrupación en varios niveles de detalle, lo que permite explorar la similitud y las relaciones entre los clústeres en diferentes niveles sin que sea necesario especificar el número de clústeres de antemano. Por el contrario, algunas de las desventajas del agrupamiento jerárquico es que pueden ser computacionalmente costosos para conjuntos de datos grandes debido a la construcción de la jerarquía y no son tan eficientes como algunos métodos no jerárquicos en cuanto a tiempo de ejecución. La interpretación de la jerarquía puede tener un grado de subjetividad y depende de cómo se decida cortar el dendrograma para seleccionar un número específico de clústeres. El agrupamiento jerárquico se utiliza en diversas aplicaciones, como biología o análisis de redes sociales, donde se requiere una comprensión detallada de la estructura de los clústeres en diferentes niveles de resolución.

Ejemplo 7. Agrupamiento jerárquico aglomerativo

Supongamos que tenemos un conjunto de datos de pacientes con características médicas y que aplicaremos un agrupamiento jerárquico aglomerativo para segmentar a los pacientes en clústeres:

```
# Generamos algunos datos simulados suponiendo que son datos de pacientes
set.seed(123)
n <- 100  # Número de pacientes
variables <- 5  # Número de variables médicas
datos <- matrix(rnorm(n * variables), ncol = variables) #definimos una matriz
para guardar los datos

# Realizamos agrupamiento jerárquico aglomerativo
dist_matrix <- dist(datos)  # Calcular la matriz de distancias
hc_aglomerativo <- hclust(dist_matrix, method = "ward.D2")  # Método de
agrupamiento aglomerativo

# Reprsentamos el dendrograma
plot(hc_aglomerativo, main = "Dendrograma de Agrupamiento Jerárquico
Aglomerativo", xlab = "Pacientes")
```




```r
# Elegimos un nivel de corte del dendograma
nivel_corte <- 4  # Por ejemplo, dividir en 4 clústeres
cluster_asignado <- cutree(hc_aglomerativo, k = nivel_corte)

# Mostramos los resultados
resultado <- data.frame(Paciente = 1:n, Cluster = cluster_asignado)
print(resultado)
```

```
##     Paciente Cluster
## 1          1       1
## 2          2       2
## 3          3       2
## 4          4       1
## 5          5       3
## 6          6       2
## 7          7       1
## 8          8       3
## 9          9       4
## 10        10       3
## 11        11       3
## 12        12       4
## 13        13       1
## 14        14       1
## 15        15       2
## 16        16       2
## 17        17       1
## 18        18       1
```

```
## 19        19        3
## 20        20        3
## 21        21        4
## 22        22        4
## 23        23        1
## 24        24        4
## 25        25        2
## 26        26        4
## 27        27        4
## 28        28        3
## 29        29        4
## 30        30        2
## 31        31        2
## 32        32        1
## 33        33        1
## 34        34        3
## 35        35        3
## 36        36        2
## 37        37        1
## 38        38        4
## 39        39        2
## 40        40        3
## 41        41        2
## 42        42        3
## 43        43        4
## 44        44        3
## 45        45        3
## 46        46        1
## 47        47        3
## 48        48        4
## 49        49        2
## 50        50        2
## 51        51        4
## 52        52        1
## 53        53        1
## 54        54        3
## 55        55        4
## 56        56        2
## 57        57        4
## 58        58        2
## 59        59        1
## 60        60        3
## 61        61        4
## 62        62        3
## 63        63        3
## 64        64        2
## 65        65        4
## 66        66        4
## 67        67        4
## 68        68        4
```

```
## 69         69       4
## 70         70       4
## 71         71       4
## 72         72       1
## 73         73       2
## 74         74       2
## 75         75       4
## 76         76       1
## 77         77       4
## 78         78       4
## 79         79       4
## 80         80       1
## 81         81       3
## 82         82       4
## 83         83       4
## 84         84       1
## 85         85       4
## 86         86       3
## 87         87       4
## 88         88       2
## 89         89       4
## 90         90       2
## 91         91       2
## 92         92       1
## 93         93       4
## 94         94       2
## 95         95       2
## 96         96       2
## 97         97       4
## 98         98       3
## 99         99       1
## 100       100       1
```

El resultado final es un marco de datos que muestra a qué clúster pertenece cada paciente en función del nivel de corte elegido.

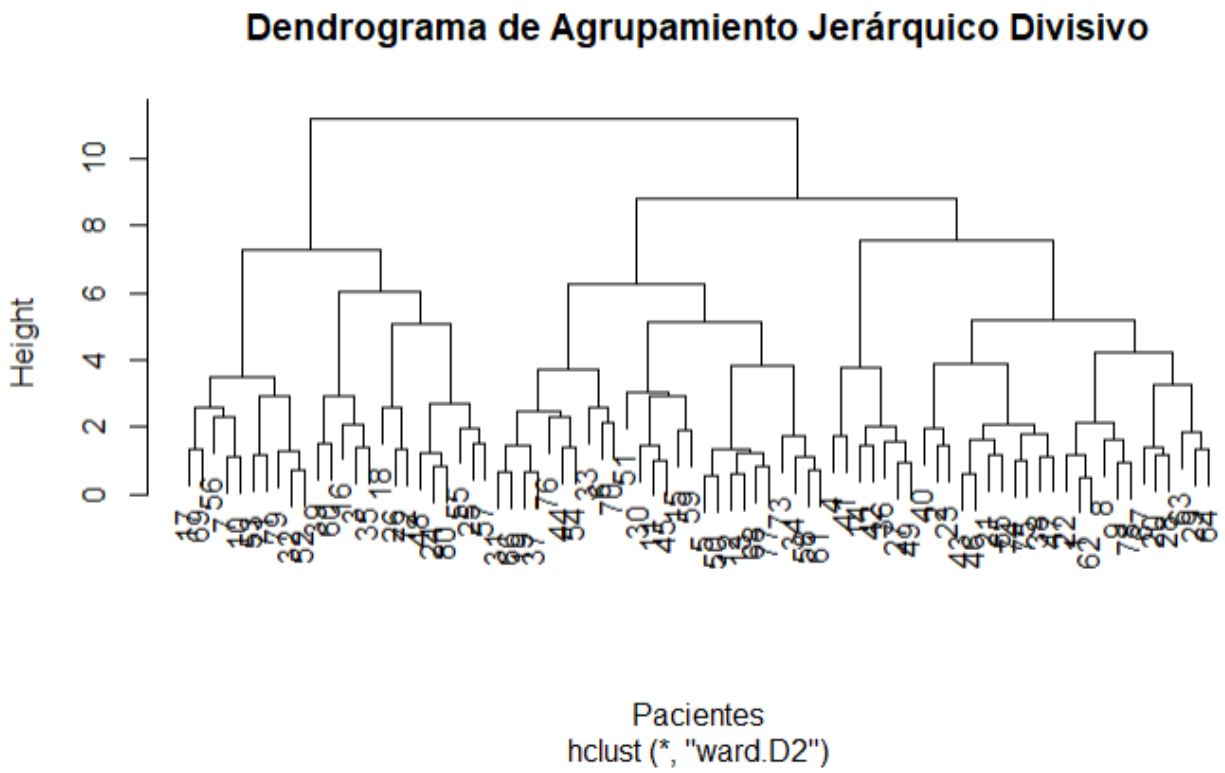
Ejemplo 8. Agrupamiento jerárquico divisivo

Supongamos que tenemos datos de pacientes con información médica y que realizamos un agrupamiento jerárquico para segmentar a los pacientes en diferentes niveles de detalle.

Primero, generamos un conjunto de datos simulados y luego aplicamos el agrupamiento jerárquico, en este caso divisivo:

```
# Generamos datos simulados
set.seed(123)
n <- 80  # Número de pacientes
variables <- 5  # Número de variables médicas
datos <- matrix(rnorm(n * variables), ncol = variables)
```

```r
# Realizamos agrupamiento jerárquico divisivo
dist_matrix <- dist(datos)  # Calcular la matriz de distancias
hc_divisivo <- hclust(dist_matrix, method = "ward.D2")  # Método de
agrupamiento divisivo

# Representación del dendrograma
plot(hc_divisivo, main = "Dendrograma de Agrupamiento Jerárquico Divisivo",
xlab = "Pacientes")
```




```r
# Elegimos un nivel de corte del dendrograma
nivel_corte <- 4  # Por ejemplo, dividir en 4 clusters
cluster_asignado <- cutree(hc_divisivo, k = nivel_corte)

# Mostramos los resultados
resultado <- data.frame(Paciente = 1:n, Cluster = cluster_asignado)
print(resultado)
```

```
##    Paciente Cluster
## 1         1       1
## 2         2       1
## 3         3       2
## 4         4       3
## 5         5       4
## 6         6       4
## 7         7       2
## 8         8       1
```

```
## 9          9        1
## 10        10        2
## 11        11        4
## 12        12        4
## 13        13        4
## 14        14        3
## 15        15        4
## 16        16        2
## 17        17        2
## 18        18        2
## 19        19        4
## 20        20        1
## 21        21        1
## 22        22        1
## 23        23        1
## 24        24        2
## 25        25        2
## 26        26        2
## 27        27        3
## 28        28        1
## 29        29        1
## 30        30        4
## 31        31        4
## 32        32        2
## 33        33        4
## 34        34        4
## 35        35        2
## 36        36        3
## 37        37        4
## 38        38        1
## 39        39        2
## 40        40        1
## 41        41        3
## 42        42        3
## 43        43        1
## 44        44        4
## 45        45        4
## 46        46        1
## 47        47        1
## 48        48        2
## 49        49        3
## 50        50        4
## 51        51        4
## 52        52        2
## 53        53        2
## 54        54        4
## 55        55        2
## 56        56        2
## 57        57        2
## 58        58        4
```

```
## 59       59       4
## 60       60       2
## 61       61       4
## 62       62       1
## 63       63       1
## 64       64       1
## 65       65       1
## 66       66       4
## 67       67       1
## 68       68       4
## 69       69       2
## 70       70       4
## 71       71       2
## 72       72       2
## 73       73       4
## 74       74       1
## 75       75       1
## 76       76       4
## 77       77       4
## 78       78       1
## 79       79       2
## 80       80       2
```

En el anterior dendrograma vemos a qué clúster pertenece cada paciente.

La principal diferencia entre los dos métodos (aglomerativo y divisivo) es la dirección en la que se forma la jerarquía de clústeres. En el aglomerativo, empezamos con clústeres individuales y los fusionamos para formar clústeres más grandes. En el divisivo, comenzamos con un solo clúster grande y lo dividimos en clústeres más pequeños a medida que se desciende en el dendrograma.

La elección del número de clústeres en el agrupamiento jerárquico divisivo puede ser subjetiva y depende de los objetivos de análisis y del contexto de los datos. Igualmente, existen algunos criterios que se pueden tomar en consideración, como la visualización del dendograma, el lugar donde se forman brazos o ramas, que sería un indicio para realizar los cortes. Por otra parte, hay otros métodos formales, como el «método del codo» (Elbow Method), que consiste en la observación de la suma de las varianzas dentro del clúster a medida que aumentan el número de clústeres. El punto en el que la reducción en la varianza disminuye significativamente se considera un codo en el gráfico y se podría utilizar para elegir el número de clústeres.

La elección entre el agrupamiento jerárquico aglomerativo y el divisivo depende de varios factores. Por una parte, el aglomerativo es útil cuando se requiere la exploración de estructuras de datos de diferentes niveles y no hay una idea clara del número de clústeres, ya que en este caso no hay que definir *a priori* un número determinado. Por otra parte, el divisivo es útil cuando hay un cierto control del número de clústeres que obtener y se requiere tener un control sobre ese número. Asimismo, si el conjunto de datos es muy

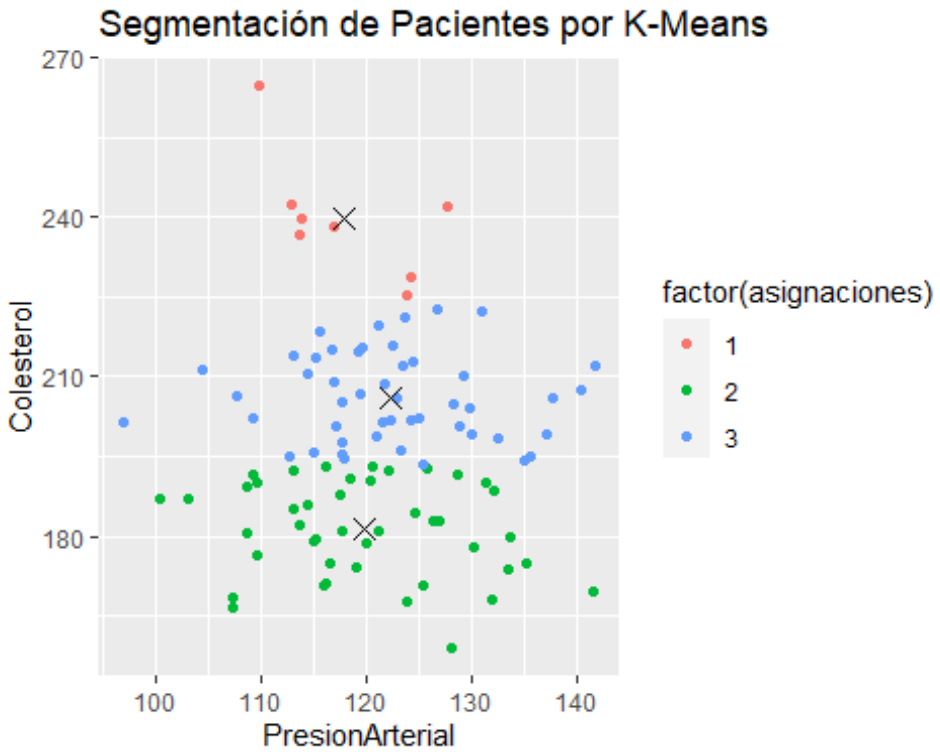
grande, este tipo de agrupamiento es más eficiente computacionalmente. Por ejemplo, se suele aplicar a la genómica y la organización de bases de datos.

Los **métodos no jerárquicos** organizan los elementos en grupos según un número de clústeres definido de antemano, creando grupos homogéneos y teniendo en cuenta que ningún elemento puede pertenecer a más de un grupo. Estos además serán heterogéneos entre ellos. Una de las técnicas más utilizadas de este tipo de método es *k-means*, cuya idea es agrupar las observaciones en clústeres distintos determinados de antemano.

Ejemplo 9. Método no jerárquico (*k-means*)

Supongamos que tenemos datos de pacientes con características médicas y queremos segmentar a los pacientes en grupos utilizando *k-means*. El objetivo principal del análisis *k-means* es agrupar a los pacientes en clústeres que compartan características médicas similares. Cada clúster representará un grupo de pacientes que son más similares entre sí.

```
# Generamos datos simulados
set.seed(123)
n <- 100  # Número de pacientes
variables <- 3  # Número de variables médicas
datos <- data.frame(
  PresionArterial = rnorm(n, mean = 120, sd = 10),
  Colesterol = rnorm(n, mean = 200, sd = 20),
  Edad = rnorm(n, mean = 50, sd = 10)
)

# Aplicar k-means con k=3 (3 clústeres)
k <- 3
set.seed(123)  # Fijamos una semilla para reproducibilidad
kmeans_resultado <- kmeans(datos, centers = k)

# Visualizamos los resultados
library(ggplot2)
asignaciones <- kmeans_resultado$cluster
centros <- kmeans_resultado$centers

# Convertir los centros calculados en un data frame
centros_df <- as.data.frame(centros)

# Representamos un gráfico de dispersión de los datos con colores para los
clústeres
ggplot(datos, aes(x = PresionArterial, y = Colesterol, color =
factor(asignaciones))) +
  geom_point() +
  geom_point(data = centros_df, aes(x = PresionArterial, y = Colesterol),
color = "black", size = 3, shape = 4) +
  labs(title = "Segmentación de Pacientes por K-Means")
```



El resultado del anterior gráfico es una segmentación de pacientes basada en sus características médicas (presión arterial, edad y colesterol). Cabe destacar que la interpretación de los clústeres se basa en las características médicas consideradas. Por ejemplo, si vemos que los pacientes en el clúster 1 tienen valores más bajos de presión arterial y colesterol en comparación con los demás clústeres, podríamos interpretar que estos pacientes tienden a tener niveles más bajos de presión arterial y colesterol.

7. Aprendizaje profundo (*deep learning*) con R

El aprendizaje profundo (*deep learning*) es una disciplina del aprendizaje automático que se centra en la construcción y el entrenamiento de algoritmos y modelos basados en redes neuronales artificiales. Estas redes neuronales se inspiran en la estructura y el funcionamiento del cerebro humano, y están diseñadas para aprender y representar patrones a partir de datos. A diferencia de las redes neuronales tradicionales, las redes de aprendizaje profundo tienen múltiples capas (de ahí el término *profundo*) que les permiten aprender representaciones más complejas y abordar tareas altamente complejas. Algunas de las aplicaciones en el ámbito de la medicina son la detección de enfermedades según imágenes y datos médicos y la predicción de resultados clínicos.

Las arquitecturas más comunes del aprendizaje profundo son las redes neuronales convolucionales (CNN) y las redes neuronales recurrentes (RNN).

Las **redes neuronales convolucionales (*Convolutional Neural Networks* o CNN)** son un tipo de red neuronal artificial especialmente diseñada para procesar y analizar datos con

estructura de cuadrícula, como imágenes y, en ocasiones, secuencias temporales. Las CNN se han vuelto fundamentales en tareas de visión por computadora y procesamiento de imágenes, y han tenido un impacto significativo en una amplia gama de aplicaciones, incluidas la detección de objetos, el reconocimiento de patrones, la clasificación de imágenes y más.

Por ejemplo, pueden ser útiles para realizar una clasificación de imágenes de radiografías de tórax como «normal» o «anormal» en R; podemos utilizar bibliotecas especializadas como keras y tensorflow para construir y entrenar una red neuronal convolucional (CNN).

Las **redes neuronales recurrentes (RNN)** son un tipo de red neuronal especializada en el procesamiento de datos secuenciales y temporales, lo que la hace adecuada para una amplia gama de aplicaciones en campos como el procesamiento de lenguaje natural, la medicina o la visión por computadora. Está diseñada para procesar datos secuenciales o temporales. A diferencia de las redes neuronales convolucionales (CNN), que se utilizan principalmente para tareas de visión por computadora, las RNN están diseñadas para trabajar con secuencias de datos, como series de tiempo, texto, audio y otros tipos de información donde la relación entre los elementos de la secuencia es importante.

El desarrollo del contenido del aprendizaje profundo queda fuera del abasto de esta asignatura. Igualmente, podéis consultar algún ejemplo de aplicación de redes neuronales en: https://cienciadedatos.net/documentos/68-redes-neuronales-r#Ejemplo_clasificaci%C3%B3n.

8. Análisis de varianza (ANOVA)

El análisis de varianza (ANOVA) es una técnica estadística que se utiliza para comparar las diferencias entre las medias de dos o más grupos. La ANOVA no es una disciplina del aprendizaje automático, pero puede ser una herramienta relevante en el proceso de desarrollo de modelos de *machine learning*, ya que permite analizar la importancia de las características de grupos, comparar modelos y realizar pruebas de hipótesis relacionadas con la variabilidad en los datos y el rendimiento de los modelos. Por ello, veamos algunos ejemplos a continuación.

Ejemplo 10

Supongamos que queremos determinar si existe una diferencia significativa en el tiempo de recuperación de pacientes después de someterse a tres diferentes tratamientos médicos. Para esto, recopilamos datos de tres grupos de pacientes, grupos que recibieron un tratamiento diferente, y medimos el tiempo de recuperación en días.

```
# Crear un dataframe con datos de pacientes y tiempos de recuperación
datos_salud <- data.frame(
  Tratamiento = factor(c("Tratamiento A", "Tratamiento B", "Tratamiento C",
"Tratamiento A", "Tratamiento B")),
  Tiempo_Recuperacion = c(10, 12, 15, 9, 11)
)
```

```
# Realizar un ANOVA
resultado_anova <- aov(Tiempo_Recuperacion ~ Tratamiento, data = datos_salud)

# Resumen del ANOVA
summary(resultado_anova)

##             Df Sum Sq Mean Sq F value Pr(>F)
## Tratamiento  2   20.2    10.1    20.2 0.0472 *
## Residuals    2    1.0     0.5
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# Realizar un test de significancia (si es necesario)
 TukeyHSD(resultado_anova)

##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = Tiempo_Recuperacion ~ Tratamiento, data = datos_salud)
##
## $Tratamiento
##                              diff        lwr       upr     p adj
## Tratamiento B-Tratamiento A  2.0 -2.1653913  6.165391 0.1857215
## Tratamiento C-Tratamiento A  5.5  0.3984584 10.601542 0.0433221
## Tratamiento C-Tratamiento B  3.5 -1.6015416  8.601542 0.1005410
```

La función *aov()* para realizar el análisis de varianza nos permite evaluar si existen diferencias significativas en los tiempos de recuperación entre los tres tratamientos. ANOVA proporciona información sobre si existe una diferencia significativa entre los tratamientos. Si el valor p es menor que un nivel de significación elegido (por ejemplo, 0.05), entonces podríamos concluir que hay diferencias significativas en los tiempos de recuperación entre al menos dos de los tratamientos. Si el ANOVA muestra diferencias significativas, se pueden realizar pruebas posteriores, como el procedimiento de Tukey (TukeyHSD(resultado_anova)) para identificar qué tratamientos difieren entre sí. De los resultados anteriores, podríamos concluir que hay una diferencia significativa en los tiempos de recuperación entre el «Tratamiento C» y el «Tratamiento A», ya que el valor de p es menor que 0.05. No hay diferencias significativas entre el «Tratamiento B» y el «Tratamiento A», ni entre el «Tratamiento C» y el «Tratamiento B», ya que los valores de p son mayores que 0.05 en ambos casos.

El objetivo del análisis de varianza (ANOVA) es obtener información acerca del efecto de un determinado factor sobre la media y la varianza de una variable continua. De forma predeterminada, en una aplicación de ANOVA, la hipótesis nula es que la media es la misma para los diferentes grupos de análisis y la hipótesis alternativa es que, al menos dos medias de los diferentes grupos, difieren. Así pues, para realizar un estudio ANOVA, inicialmente seleccionaremos los grupos a partir de los cuales podremos calcular las medias y, posteriormente, comparar las varianzas de estas medias con la varianza promedio de cada

grupo. En función de los datos que debamos estudiar, si estos están o no pareados y en función de si tenemos en cuenta un único factor o diversos, tendremos ANOVA de una vía o de dos para datos independientes o dependientes.

Veamos otro ejemplo que profundiza más en el desarrollo de una ANOVA.

Ejemplo 11

A continuación, realizamos un análisis de varianza (ANOVA) sobre el conjunto de datos *birthwt* del paquete *MASS* considerando las variables *smoke* y *bwt*.

```
#Cargamos el paquete MASS
library(MASS)
#Referenciamos el conjunto de datos birthwt
data("birthwt")
head(birthwt)
```

```
##    low age lwt race smoke ptl ht ui ftv  bwt
## 85   0  19 182    2     0   0  0  1   0 2523
## 86   0  33 155    3     0   0  0  0   3 2551
## 87   0  20 105    1     1   0  0  0   1 2557
## 88   0  21 108    1     1   0  0  1   2 2594
## 89   0  18 107    1     1   0  0  1   0 2600
## 91   0  21 124    3     0   0  0  0   0 2622
```

```
fuma<-c(birthwt$smoke)
peso<-c(birthwt$bwt)
df_birthwt_fuma<-data.frame(fuma,peso) #definimos el conjunto de datos
head(df_birthwt_fuma,4) #mostramos la información del data frame
```

```
##   fuma peso
## 1    0 2523
## 2    0 2551
## 3    1 2557
## 4    1 2594
```

```
table(df_birthwt_fuma$fuma) #número de observaciones por cada grupo
```

```
##
##   0   1
## 115  74
```

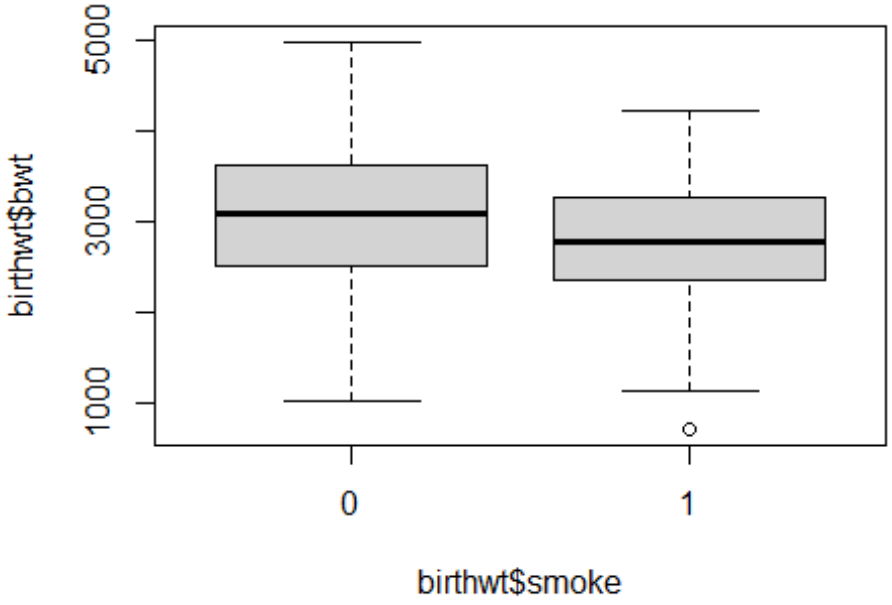
Calculamos la media y varianza:

```
##Media de pesos por raza
aggregate(peso~fuma,data=df_birthwt_fuma,FUN=mean)
```

```
##   fuma     peso
## 1    0 3055.696
## 2    1 2771.919
```

Vemos en una representación gráfica si pudieran existir asimetrías, datos atípicos u otras diferencias significativas:

```
boxplot(birthwt$bwt~birthwt$smoke, data=birthwt, id.method="y")
```



No se perciben grandes diferencias entre ambos grupos. Comprobamos las condiciones para aplicar un ANOVA. Por una parte, disponemos de una muestra de estudio donde las observaciones de cada grupo son independientes entre sí; por otra parte, vemos mediante un estudio de normalidad si la variable cuantitativa, en este caso peso, se distribuye de forma normal en cada grupo.

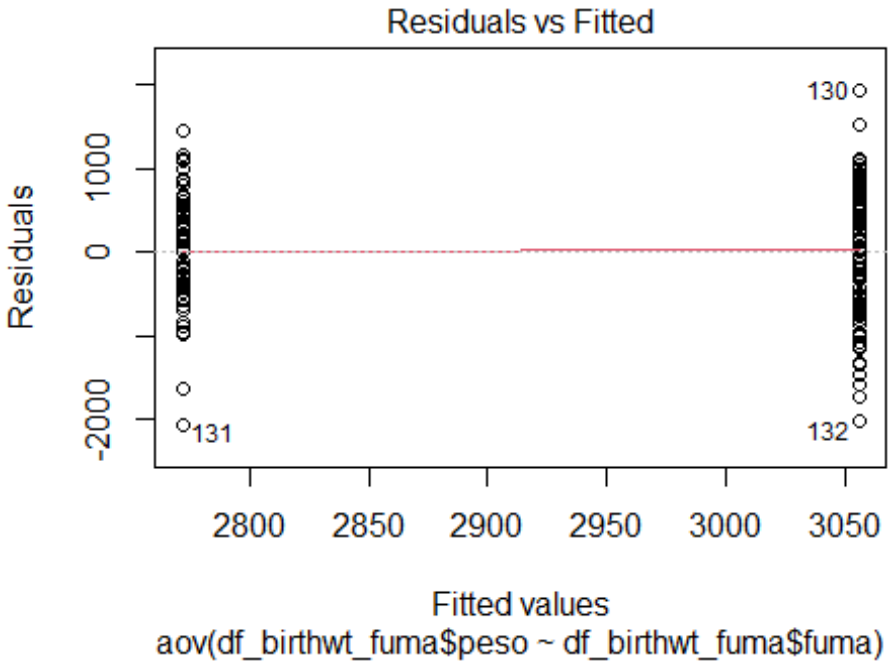
A continuación, aplicamos el test de Kolmogorov-Smirnov, que acabará de determinar la normalidad de la distribución.

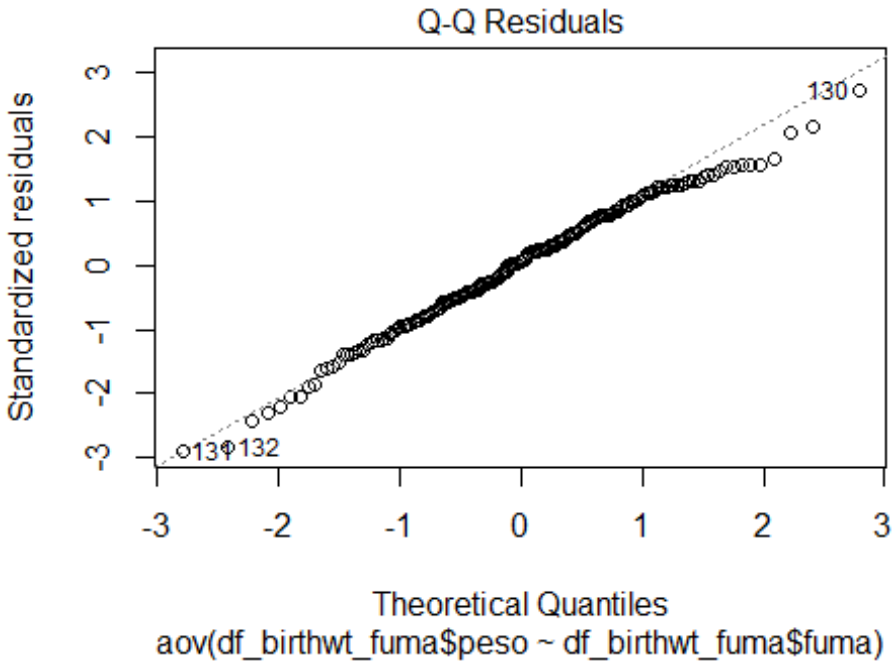
```
##Test Kolmogorov-Smirnov
##Instalaremos el paquete nortest que contiene las funciones a aplicar
require(nortest)

## Loading required package: nortest

## Loading required package: nortest
by(data=df_birthwt_fuma,INDICES =
df_birthwt_fuma$fuma,FUN=function(x){lillie.test(x$peso)})

## df_birthwt_fuma$fuma: 0
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  x$peso
## D = 0.059863, p-value = 0.3967
##
## ------------------------------------------------------------
## df_birthwt_fuma$fuma: 1
```

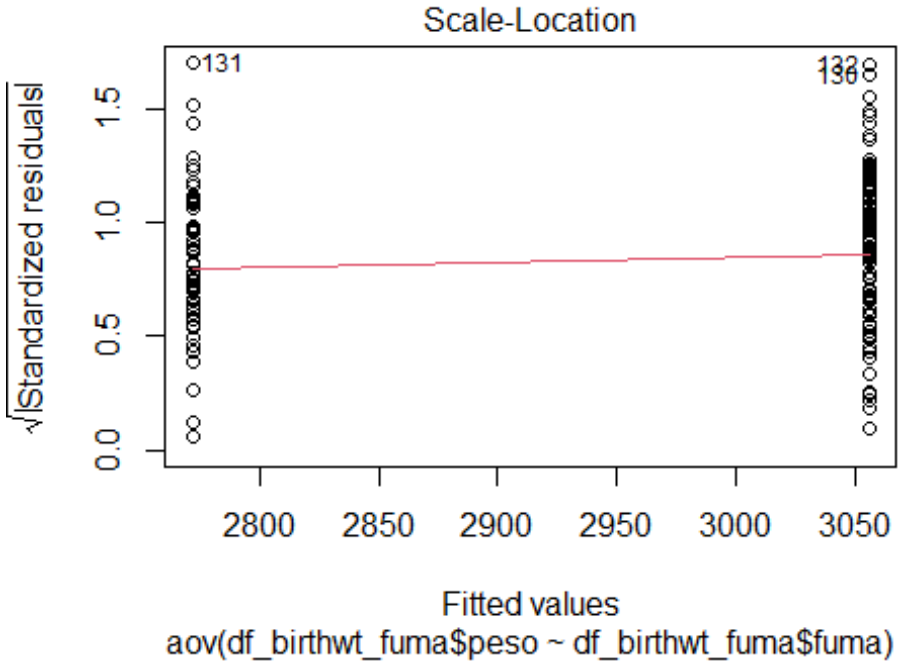
```
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  x$peso
## D = 0.068092, p-value = 0.5396
```

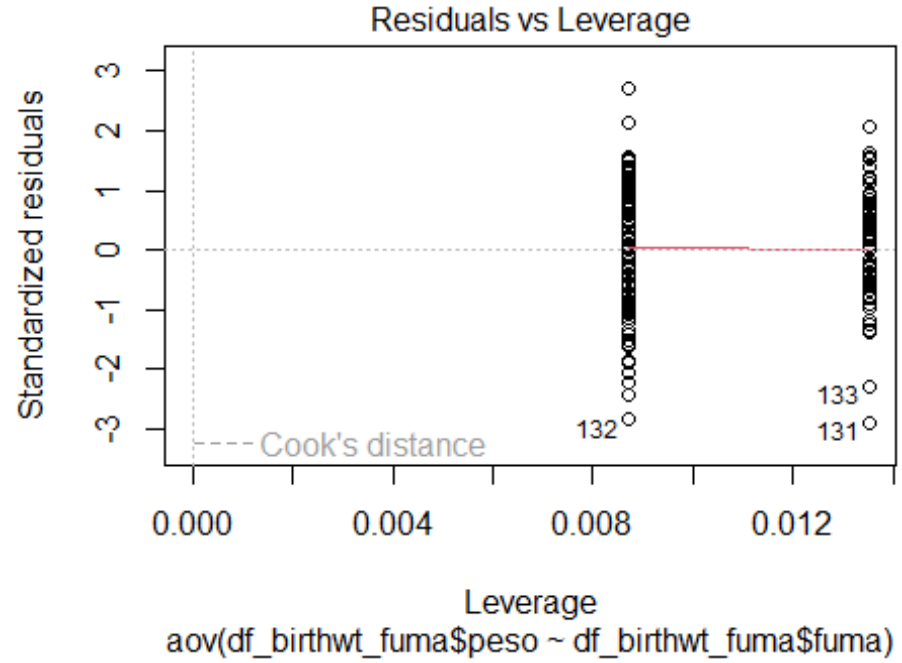
Los anteriores test de normalidad muestran normalidad, ya que los p-value muestran valores por encima de 0.05. A continuación, debemos evaluar la varianza constante entre los grupos, para ello aplicamos el test de Fligner-Killeen:

```
##Test Fligner-Killen diagnosis homocedasticidad
fligner.test(df_birthwt_fuma$peso~df_birthwt_fuma$fuma,df_birthwt_fuma)

##
##  Fligner-Killeen test of homogeneity of variances
##
## data:  df_birthwt_fuma$peso by df_birthwt_fuma$fuma
## Fligner-Killeen:med chi-squared = 1.5788, df = 1, p-value = 0.2089

##Estudio anova
anova_birthwt_f<-
aov(df_birthwt_fuma$peso~df_birthwt_fuma$fuma,data=df_birthwt_fuma)
summary(anova_birthwt_f)

##                       Df   Sum Sq Mean Sq F value  Pr(>F)
## df_birthwt_fuma$fuma   1  3625946 3625946   7.038 0.00867 **
## Residuals            187 96343710  515207
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

plot(anova_birthwt_f)
```


Residuals vs Fitted
Residuals
1000
0
-2000
130
131
132
2800
2850
2900
2950
3000
3050
Fitted values
aov(df_birthwt_fuma$peso ~ df_birthwt_fuma$fuma)


Q-Q Residuals
Standardized residuals
3
2
1
0
-1
-2
-3
130
131
132
-3
-2
-1
0
1
2
3
Theoretical Quantiles
aov(df_birthwt_fuma$peso ~ df_birthwt_fuma$fuma)





De los resultados se desprende que no existen diferencias significativas entre los grupos.

9. jercicios y casos prácticos en R

Ejercicio 1. Modelos supervisados. Análisis de regresión múltiple

A partir del conjunto de datos *swiss* del paquete *datasets*, calculad la matriz de correlación y, a partir de esta, realizad un análisis de regresión múltiple. Razonad los resultados.

Ejercicio 2. Modelos supervisados. Modelo de clasificación SVM

A partir del conjunto de datos *Pima.tr* del paquete *MASS* que hace referencia a datos de diabetes de pacientes indias, realizad un modelo de clasificación SVM y realizad una clasificación de pacientes para predecir si están en riesgo de padecer diabetes. Para realizar este ejercicio utilizad el paquete *kernlab* para entrenar el modelo SVM.

Ejercicio 3. Modelos no supervisados. PCA

A partir del conjunto de datos Iris, se pide realizar un análisis de componentes centrado, por ejemplo, en las medidas de longitud y ancho del sépalo y el pétalo.

Ejercicio 4. Agrupamiento jerárquico. Agrupamiento jerárquico aglomerativo

Realizad un agrupamiento jerárquico aglomerativo con el paquete de datos *birthwt* del paquete *MASS*.

Ejercicio 5. Agrupamiento jerárquico. Agrupamiento jerárquico divisivo

Realizad un agrupamiento jerárquico divisivo con el paquete de datos *birthwt* del paquete *MASS*.

Ejercicio 6. Agrupamiento jerárquico. Agrupamiento jerárquico aglomerativo

Realizad un agrupamiento jerárquico de tipo aglomerativo para el conjunto de datos *melanoma* del paquete *MASS*.

Ejercicio 7. Agrupamiento jerárquico. Agrupamiento jerárquico divisivo

Realizad un agrupamiento jerárquico de tipo aglomerativo para el conjunto de datos *melanoma* del paquete *MASS*.

Ejercicio 8. Agrupamiento no jerárquico

Realizad un agrupamiento no jerárquico utilizando el conjunto de datos *birthwt* del paquete *MASS*.

Ejercicio 9. Agrupamiento no jerárquico

Realizad un agrupamiento no jerárquico utilizando la técnica *k-means()* para el conjunto de datos *melanoma* del paquete *MASS*. Valorad los resultados para distintos números de clústeres y definiendo una determinada variable objetivo. Podéis transformar alguna variable entera de modo categórico, por ejemplo, la referente al estatus.

Ejercicio 10. ANOVA

A partir del conjunto de datos Iris, aplicad la función ANOVA y razonad los resultados obtenidos.

Caso práctico 1. Modelos supervisados. Regresión lineal y múltiple

A partir del conjunto de datos *Ovarian* del paquete *survival*, haced un estudio de regresión lineal sobre un par de variables que, *a priori*, penséis que pueden presentar correlación, y un estudio de regresión múltiple seleccionando los mejores predictores.

Caso práctico 2. Modelos aprendizaje automático y ANOVA

A partir del conjunto de datos *lung* del paquete *survival*, se pide realizar un breve estudio aplicando los conceptos estudiados en este **LAB**. Para ello, se pide realizar un estudio de regresión lineal, regresión múltiple, test ANOVA y test *clustering* sobre dicho conjunto de datos. Las variables elegidas en cada uno de los casos posibles que tratar se dan a elegir al estudiante.

Caso práctico 3. ANOVA

A partir del conjunto de datos *birthwt* del paquete *MASS*, vamos a intentar obtener conclusiones acerca de la relación existente entre la raza de la madre, teniendo en cuenta si esta es o no fumadora, y el bajo peso del recién nacido. Este estudio podría contextualizarse en una prueba ANOVA de una vía para datos independientes.

Solución a los ejercicios propuestos y casos prácticos en R

Solución del ejercicio 1:

Cargamos el conjunto de datos:

```
# Cargar el conjunto de datos swiss
data(swiss)
```

Realizamos la matriz de correlación:

```
cor(x=swiss,method="pearson")
```

```
##                   Fertility Agriculture Examination   Education   Catholic
## Fertility         1.0000000  0.35307918  -0.6458827 -0.66378886  0.4636847
## Agriculture       0.3530792  1.00000000  -0.6865422 -0.63952252  0.4010951
## Examination      -0.6458827 -0.68654221   1.0000000  0.69841530 -0.5727418
## Education        -0.6637889 -0.63952252   0.6984153  1.00000000 -0.1538589
## Catholic          0.4636847  0.40109505  -0.5727418 -0.15385892  1.0000000
## Infant.Mortality  0.4165560 -0.06085861  -0.1140216 -0.09932185  0.1754959
##                  Infant.Mortality
## Fertility              0.41655603
## Agriculture           -0.06085861
## Examination           -0.11402160
## Education             -0.09932185
## Catholic               0.17549591
## Infant.Mortality       1.00000000
```

Podemos observar que no hay una alta correlación, *a priori*, entre ninguna de las variables.

Procedemos a realizar el análisis de regresión múltiple:

```
# Realizamos un análisis de regresión múltiple
model <- lm(Fertility ~ Agriculture + Examination + Education + Catholic +
Infant.Mortality, data = swiss)

# Mostramos un resumen del modelo
summary(model)
```

```
##
## Call:
## lm(formula = Fertility ~ Agriculture + Examination + Education +
##     Catholic + Infant.Mortality, data = swiss)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.2743  -5.2617   0.5032   4.1198  15.3213
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      66.91518   10.70604   6.250 1.91e-07 ***
```

```
## Agriculture      -0.17211    0.07030  -2.448  0.01873 *
## Examination      -0.25801    0.25388  -1.016  0.31546
## Education        -0.87094    0.18303  -4.758 2.43e-05 ***
## Catholic          0.10412    0.03526   2.953  0.00519 **
## Infant.Mortality  1.07705    0.38172   2.822  0.00734 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.165 on 41 degrees of freedom
## Multiple R-squared:  0.7067, Adjusted R-squared:  0.671
## F-statistic: 19.76 on 5 and 41 DF,  p-value: 5.594e-10
```

El resultado anterior indica que el modelo tiene ajuste moderado y el valor de *R-Squared*, que no está demasiado cercano al 1, indica que existen variables irrelevantes en el modelo. Por otra parte, el p-value tan bajo puede indicar que el modelo en general es significativo pero mejorable.

Solución del ejercicio 2:

```
# Cargamos el paquete MASS
library(MASS)

# Cargamos el conjunto de datos de 'Diabetesin Pima Indian Women'
data(Pima.tr)

#Mostramos el datase
head(Pima.tr)
```

```
##   npreg glu bp skin  bmi   ped age type
## 1     5  86 68   28 30.2 0.364  24   No
## 2     7 195 70   33 25.1 0.163  55  Yes
## 3     5  77 82   41 35.8 0.156  35   No
## 4     0 165 76   43 47.9 0.259  26   No
## 5     0 107 60   25 26.4 0.133  23   No
## 6     5  97 76   27 35.6 0.378  52  Yes
```

```
# Dividimos el conjunto de datos en entrenamiento y prueba
set.seed(123)
indices_entrenamiento <- sample(1:nrow(Pima.tr), 0.5 * nrow(Pima.tr)) #pueden
realizarse pruebas escogiendo otro criterio
conjunto_entrenamiento <- Pima.tr[indices_entrenamiento, ]
conjunto_prueba <- Pima.tr[-indices_entrenamiento, ]

# Cargamos el paquete "kernlab" para SVM
library(kernlab)

##
## Attaching package: 'kernlab'
```

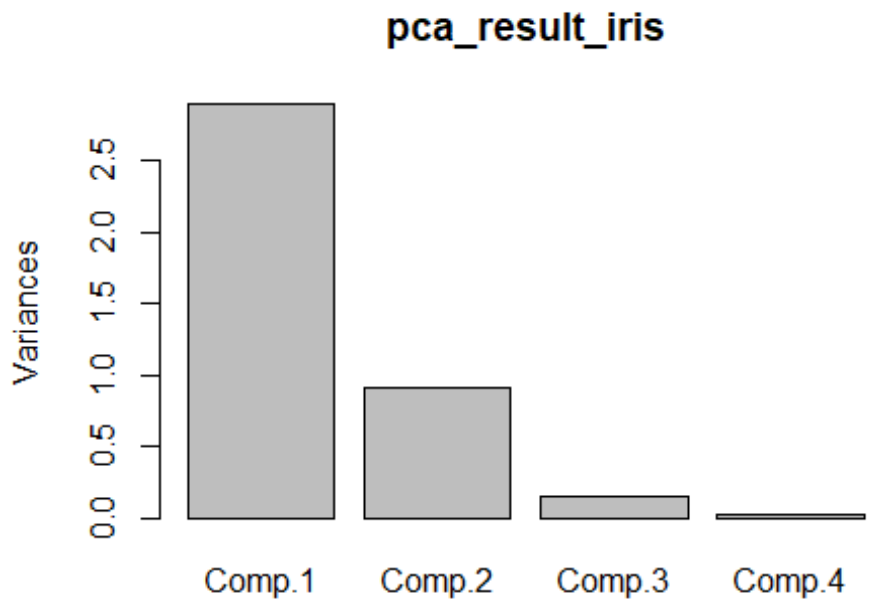
```
## The following object is masked from 'package:ggplot2':
##
##     alpha

# Entrenamos un modelo SVM
modelo_svm <- ksvm(type ~ ., data = conjunto_entrenamiento)

# Realizar predicciones en el conjunto de prueba
predicciones <- predict(modelo_svm, newdata = conjunto_prueba)

# Calcular la matriz de confusión
confusion_matriz <- table(Real = conjunto_prueba$type, Predicción =
predicciones)

# Calcular la precisión
precision <- sum(diag(confusion_matriz)) / sum(confusion_matriz)
cat("Precisión del modelo SVM:", precision, "\n")

## Precisión del modelo SVM: 0.73

# Mostrar la matriz de confusión
confusion_matriz

##      Predicción
## Real  No Yes
##   No  57   5
##   Yes 22  16
```

Los ajustes de predicción del modelo son del 65 %.

Solución del ejercicio 3:

```
# Cargamos el conjunto de datos Iris
data(iris)

# Visualizamos las primeras filas del conjunto de datos
head(iris)

##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa

# Seleccionamos las variables relevantes para el análisis de PCA (por
ejemplo, las medidas de longitud y ancho del sépalo y pétalo)
selection_iris <- iris[, c("Sepal.Length", "Sepal.Width", "Petal.Length",
"Petal.Width")]
```

```r
# Estandarizamos los datos (es recomendable para PCA)
scaled_data <- scale(selection_iris)

# Realizamos el PCA
pca_result_iris <- princomp(scaled_data)

# Resumen de los resultados
summary(pca_result_iris)

## Importance of components:
##                           Comp.1    Comp.2     Comp.3      Comp.4
## Standard deviation     1.7026571 0.9528572 0.38180950 0.143445939
## Proportion of Variance 0.7296245 0.2285076 0.03668922 0.005178709
## Cumulative Proportion  0.7296245 0.9581321 0.99482129 1.000000000

# Realizamos una representación gráfica de la pca_result_iris
plot(pca_result_iris)
```

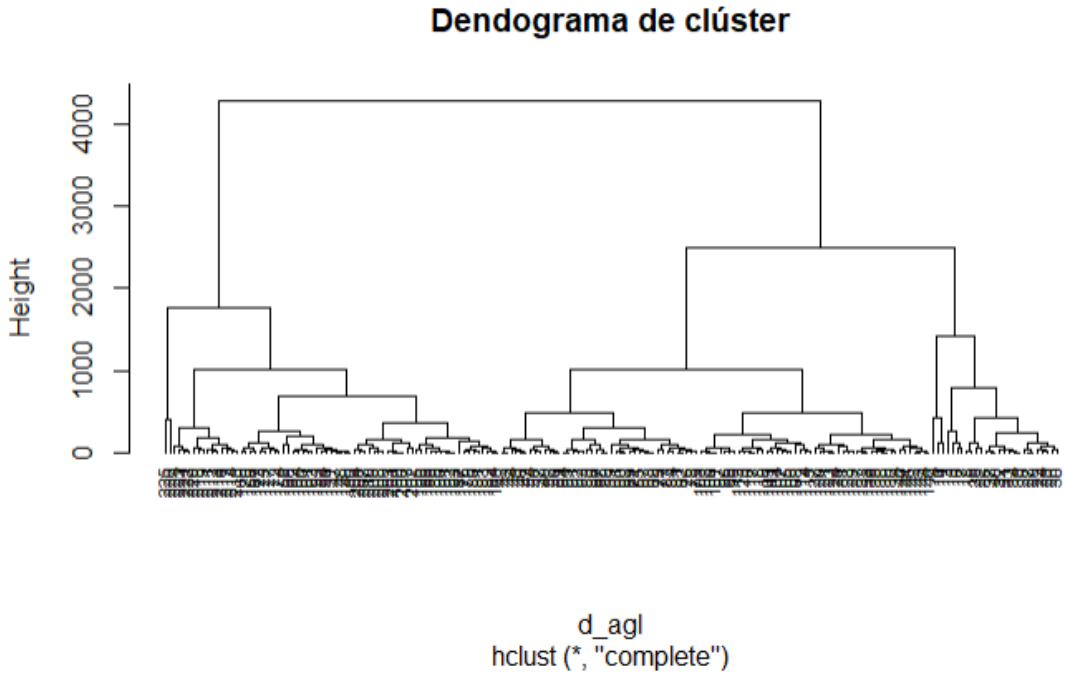


El primer componente 1 (Comp. 1) tiene una desviación estándar de aproximadamente 1.7027, que representa la mayor variabilidad de totas las componentes definidas y, además, representa el 72.96 %, aproximadamente, de la varianza total en los datos. Por lo tanto, la Comp. 1 es el componente más importante para explicar la variación de los datos.

Solución del ejercicio 4:

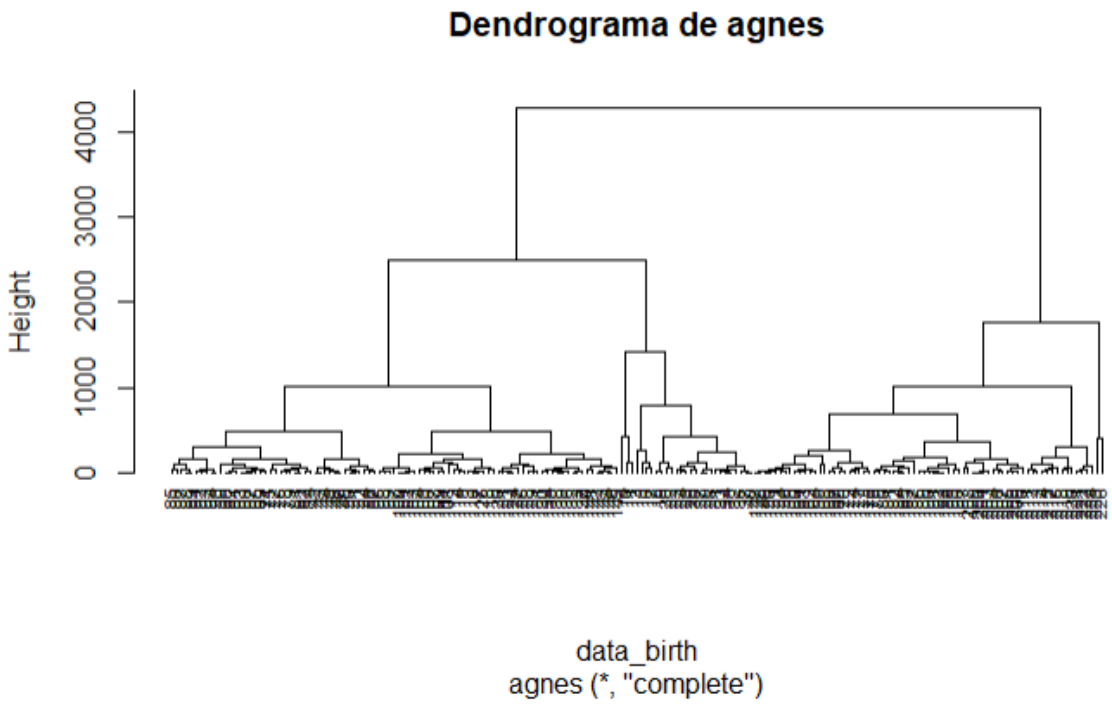
```r
##Agrupamiento jerárquico aglomerativo. Cada observación es un clúster que se va organizando hasta converger en una única rama central.
library(MASS)
library(cluster)
data("birthwt")
data_birth<-birthwt
d_agl<-dist(data_birth,method="euclidean") ##especificamos los valores de
```

```
distancia
hc_agl<-hclust(d_agl,method="complete") ##calculamos el clúster jerárquico
plot(hc_agl,cex=0.6,hang=-1, main ="Dendrograma de clúster") ##representamos
el dendograma
```



Si utilizamos la función agnes obtendremos, además del dendrograma, el coeficiente de aglomeración que indica el grado de fortaleza del agrupamiento, mejor cuanto más cercano a 1.

```
##Agrupamiento jerárquico aglomerativo
hc_ag<-agnes(data_birth,method="complete") ##calculamos los clústeres
jerárquicos
pltree(hc_ag,cex=0.6,hang=-1,main="Dendrograma de agnes") ##representamos
dendograma
```




```
hc_ag$ac ##calculamos el coeficiente de aglomeración

## [1] 0.9917573
```

Podemos observar que el coeficiente de aglomeración es muy cercano a 1, lo cual indica que la agrupación es fuerte.

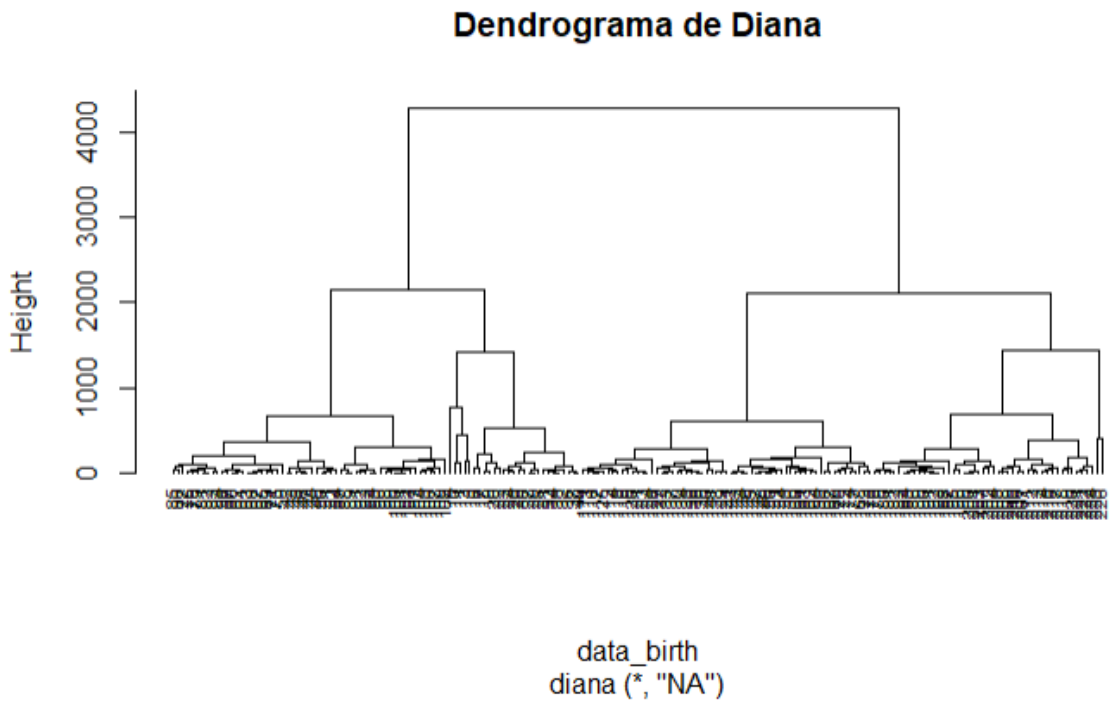
Solución del ejercicio 5:

Veamos ahora un ejemplo de agrupamiento divisional jerárquico utilizando la función diana. Cargamos inicialmente los datos correspondientes al conjunto de datos.

```
library(MASS)
library(cluster)
data_birth<-birthwt
```

Ahora realizamos el agrupamiento jerárquico:

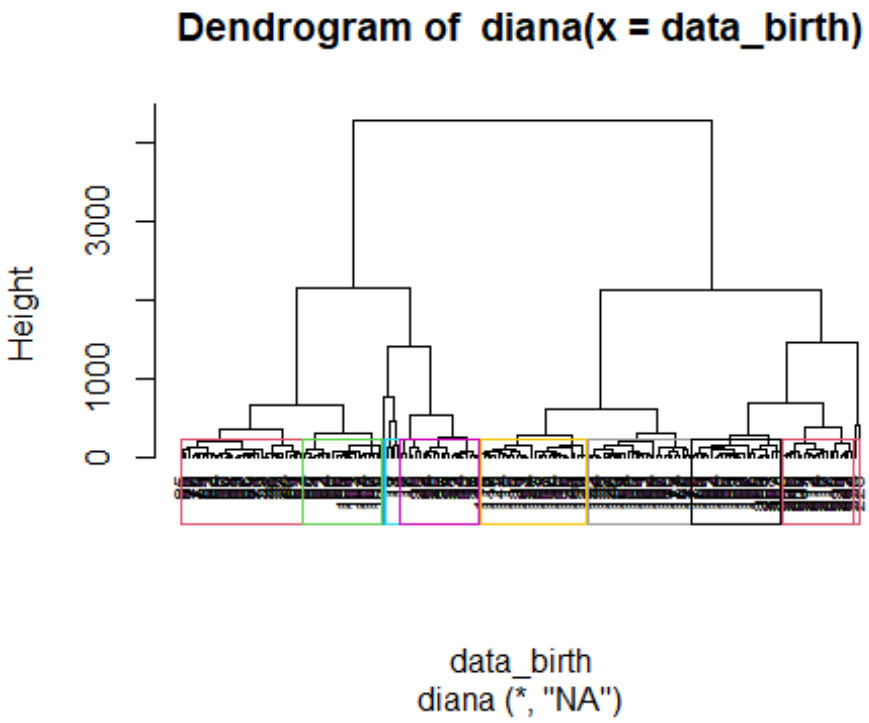
```
##Agrupamiento divisional jerárquico
hc_div<-diana(data_birth) ##calculamos los clústeres jerárquicos
pltree(hc_div,cex=0.6,hang=-1,main="Dendrograma de Diana") ##representamos el
dendrograma
```




```
hc_div$dc ##calculamos el coeficiente de división

## [1] 0.9907511
```

Igualmente, el coeficiente de división es un buen valor indicativo de la agrupación. Una vez creados los agrupamientos, asociamos los clústeres a las observaciones de datos. Una opción sencilla sería añadir a la anterior representación gráfica la división por clúster de la manera siguiente:

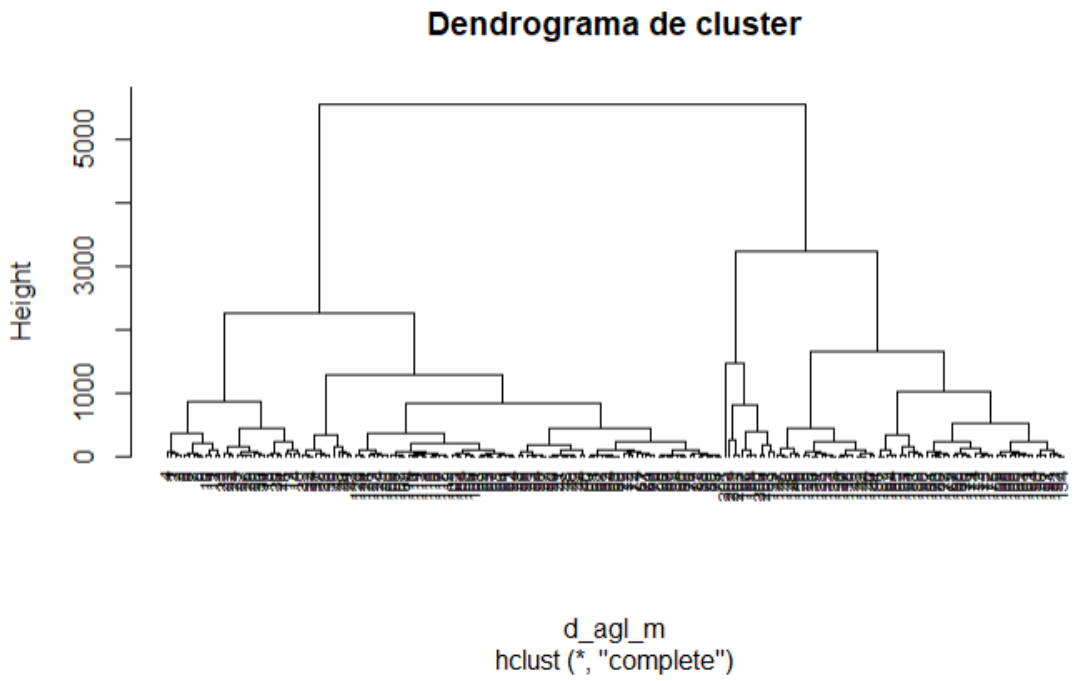
```
pltree(hc_div,hang=-1,cex=0.6)
rect.hclust(hc_div,k=10,border=2:10)
```



Solución del ejercicio 6:

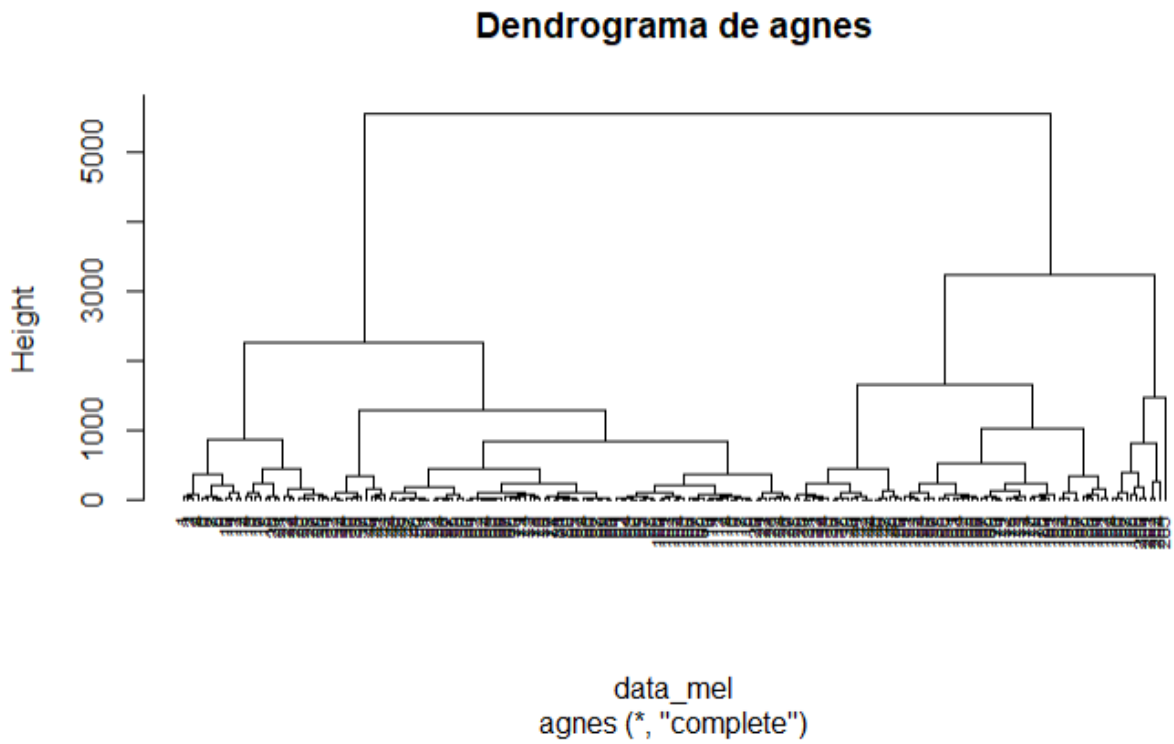
Cargamos el *dataset melanoma* t y realizamos el agrupamiento jerárquico aglomerativo:

```
##Agrupamiento jerárquico aglomerativo. Cada observación es un clúster que se va organizando hasta converger en una única rama central.
library(MASS)
library(cluster)
data_mel<-Melanoma
d_agl_m<-dist(data_mel,method="euclidean") ##especificamos los valores de distancia
hc_agl_m<-hclust(d_agl_m,method="complete") ##calculamos el clúster jerárquico
plot(hc_agl_m,cex=0.6,hang=-1, main ="Dendrograma de cluster")
##representamos el dendrograma
```



Repetimos el mismo proceso con la función *agnes()*:

```
hc_ag_m<-agnes(data_mel,method="complete") ##calculamos los clústeres
jerárquicos
pltree(hc_ag_m,cex=0.6,hang=-1,main="Dendrograma de agnes") ##representamos
dendrograma
```



Ahora calculamos el coeficiente de aglomeración:

```
hc_ag_m$ac ##calculamos el coeficiente de aglomeración

## [1] 0.9941753
```

Podemos observar que el coeficiente de aglomeración es muy cercano a 1, lo cual indica que la agrupación es fuerte.

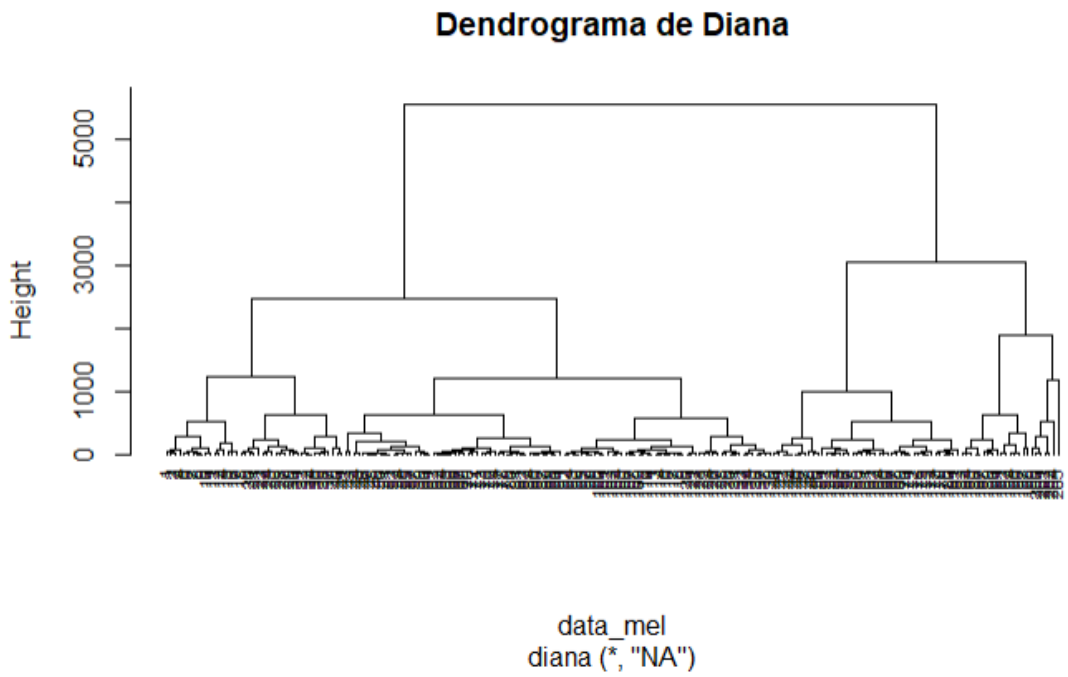
Solución del ejercicio 7:

Cargamos el *dataset* correspondiente:

```
library(MASS)
library(cluster)
data("Melanoma")
data_mel<-Melanoma
head(data_mel)

##   time status sex age year thickness ulcer
## 1   10      3   1  76 1972      6.76     1
## 2   30      3   1  56 1968      0.65     0
## 3   35      2   1  41 1977      1.34     0
## 4   99      3   0  71 1968      2.90     0
## 5  185      1   1  52 1965     12.08     1
## 6  204      1   1  28 1971      4.84     1
```

Realicemos ahora un ejemplo de agrupamiento divisional jerárquico utilizando la función *diana()*:

```
##Agrupamiento divisional jerárquico
hc_div_m<-diana(data_mel) ##calculamos los clústeres jerárquicos
pltree(hc_div_m,cex=0.6,hang=-1,main="Dendrograma de Diana") ##representamos
el dendrograma
```

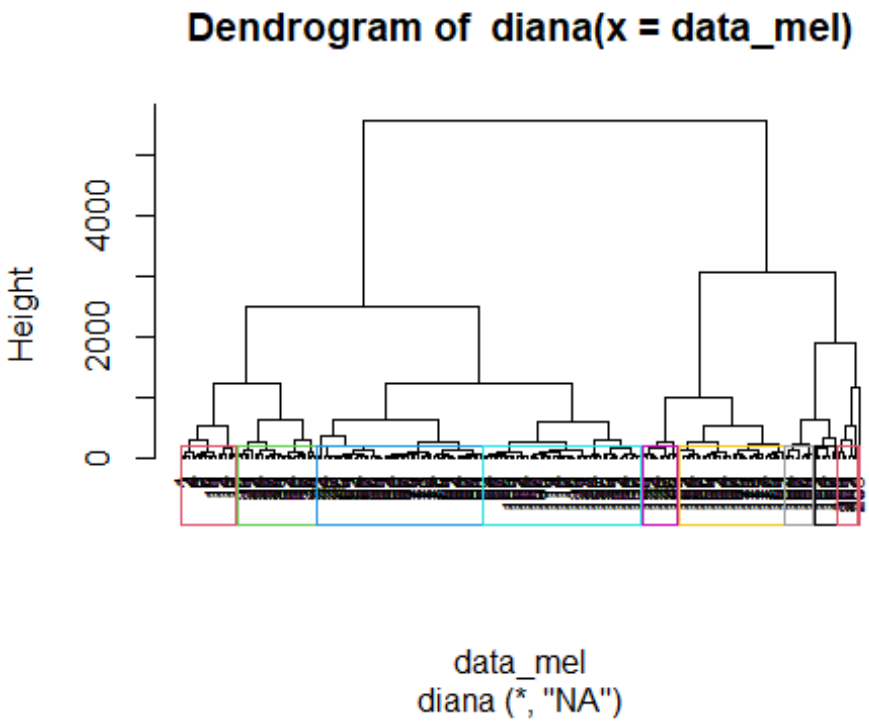


Calculamos el coeficiente de división:

```
hc_div_m$dc ##calculamos el coeficiente de división

## [1] 0.9940069
```

Una vez creados los agrupamientos, asociamos los clústeres a las observaciones de datos. Una opción sencilla sería añadir a la anterior representación gráfica la división por clúster de la siguiente manera:

```
pltree(hc_div_m,hang=-1,cex=0.6)
rect.hclust(hc_div_m,k=10,border=2:10)
```

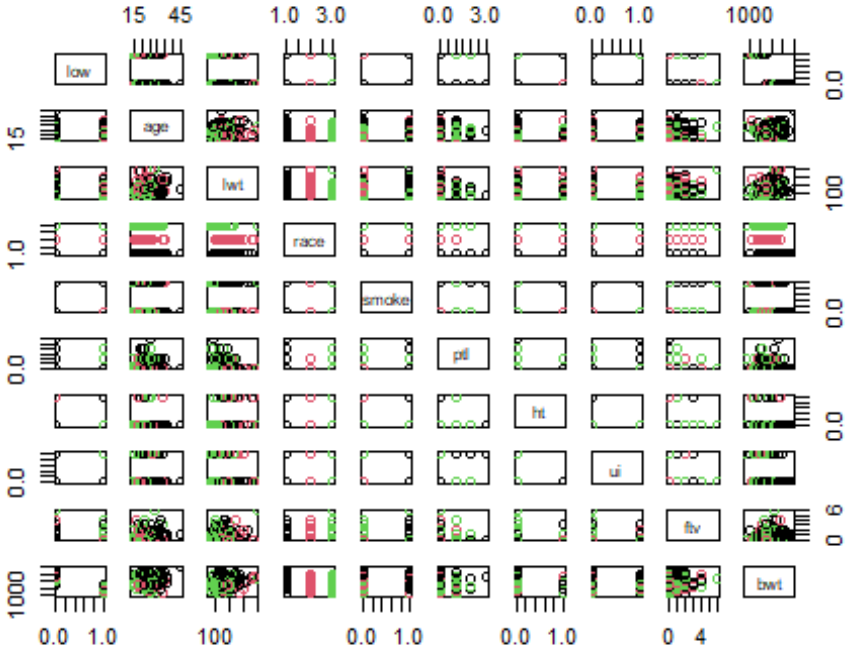


Solución del ejercicio 8:

Cargamos el conjunto de datos:

```
##invocamos el conjunto de datos birthwt
data_birth_k=MASS::birthwt
head(data_birth_k)

##     low age lwt race smoke ptl ht ui ftv  bwt
## 85    0  19 182    2     0   0  0  1   0 2523
## 86    0  33 155    3     0   0  0  0   3 2551
## 87    0  20 105    1     1   0  0  0   1 2557
## 88    0  21 108    1     1   0  0  1   2 2594
## 89    0  18 107    1     1   0  0  1   0 2600
## 91    0  21 124    3     0   0  0  0   0 2622

str(data_birth_k) ##comprobamos el tipo de variables. Hay 10 variables, todas
ellas numéricas, aunque race se podría considerar categórica

## 'data.frame':    189 obs. of  10 variables:
##  $ low  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ age  : int  19 33 20 21 18 21 22 17 29 26 ...
##  $ lwt  : int  182 155 105 108 107 124 118 103 123 113 ...
##  $ race : int  2 3 1 1 1 3 1 3 1 1 ...
##  $ smoke: int  0 0 1 1 1 0 0 0 1 1 ...
##  $ ptl  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ht   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ui   : int  1 0 0 1 1 0 0 0 0 0 ...
```

```
##  $ ftv  : int  0 3 1 2 0 0 1 1 1 0 ...
##  $ bwt  : int  2523 2551 2557 2594 2600 2622 2637 2637 2663 2665 ...

##consideramos race nuestra variable objetivo
##realizamos un gráfico de dispersión según la variable predictora
pairs(data_birth_k[-11],col=data_birth_k$race)
```




```
##comprobamos el grado de correlación entre el resto de variables numéricas
corr=cor(data_birth_k[-11])
corr

##                low         age         lwt         race       smoke
ptl
## low    1.00000000 -0.11893928 -0.16962694  0.137792751  0.16140431
0.196087267
## age   -0.11893928  1.00000000  0.18007315 -0.172817953 -0.04434618
0.071606386
## lwt   -0.16962694  0.18007315  1.00000000 -0.165048544 -0.04417908 -
0.140029003
## race   0.13779275 -0.17281795 -0.16504854  1.000000000 -0.33903074
0.007951293
## smoke  0.16140431 -0.04434618 -0.04417908 -0.339030745  1.00000000
0.187557063
## ptl    0.19608727  0.07160639 -0.14002900  0.007951293  0.18755706
1.000000000
## ht     0.15237025 -0.01583700  0.23636040  0.019929917  0.01340704 -
0.015399579
## ui     0.16904283 -0.07515558 -0.15276317  0.053602088  0.06215900
0.227585340
## ftv   -0.06296026  0.21539394  0.14052746 -0.098336254 -0.02801314 -
```

```
0.044429660
## bwt   -0.78480516  0.09031781  0.18573328 -0.194713487 -0.19044806 -
0.154653390
##                ht          ui         ftv         bwt
## low    0.15237025  0.16904283 -0.06296026 -0.78480516
## age   -0.01583700 -0.07515558  0.21539394  0.09031781
## lwt    0.23636040 -0.15276317  0.14052746  0.18573328
## race   0.01992992  0.05360209 -0.09833625 -0.19471349
## smoke  0.01340704  0.06215900 -0.02801314 -0.19044806
## ptl   -0.01539958  0.22758534 -0.04442966 -0.15465339
## ht     1.00000000 -0.10858506 -0.07237255 -0.14598189
## ui    -0.10858506  1.00000000 -0.05952341 -0.28392741
## ftv   -0.07237255 -0.05952341  1.00000000  0.05831777
## bwt   -0.14598189 -0.28392741  0.05831777  1.00000000
```

A partir de estos gráficos podemos intuir qué variables pueden aportar más información para la formación de grupos. No parece que haya ninguna combinación clara. Realizaremos la combinación con la variable *bwt*. Sería adecuado transformar la variable *race* de tipo entero en otra variable de tipo categórico.

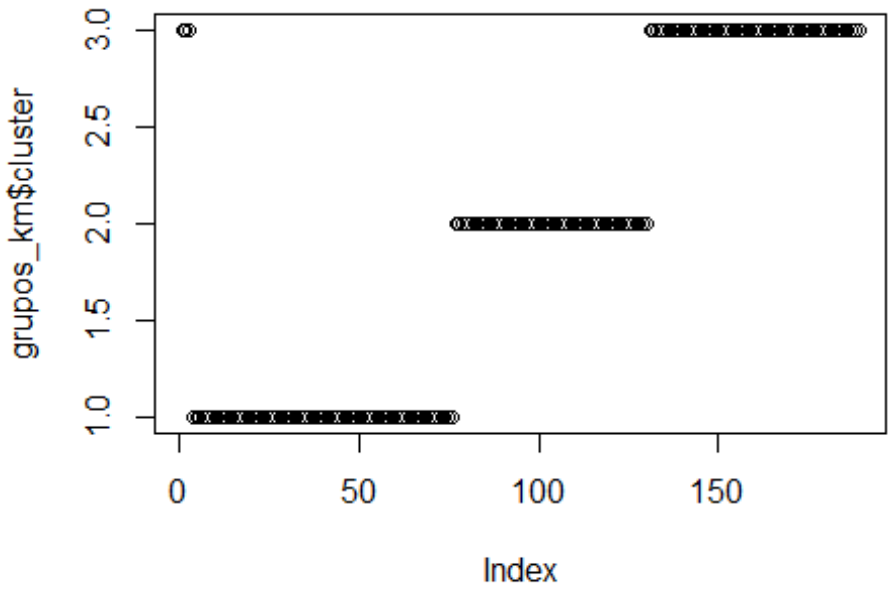
```
##Construimos los grupos de clústeres, crearemos 3 grupos que son los tipos
de raza (*race*)
grupos_km=kmeans(data_birth_k,3) ##10 variables, 3 tipos de raza
grupos_km

## K-means clustering with 3 clusters of sizes 73, 54, 62
##
## Cluster means:
##         low      age      lwt     race     smoke       ptl         ht
## 1 0.0000000 23.24658 128.9041 1.917808 0.4109589 0.1506849 0.02739726
## 2 0.0000000 24.20370 138.4630 1.537037 0.2407407 0.1111111 0.05555556
## 3 0.9516129 22.38710 123.3548 2.032258 0.5000000 0.3225806 0.11290323
##           ui       ftv      bwt
## 1 0.15068493 0.8904110 3009.986
## 2 0.03703704 0.7407407 3804.148
## 3 0.24193548 0.7258065 2118.935
##
## Clustering vector:
##  85  86  87  88  89  91  92  93  94  95  96  97  98  99 100 101 102 103
104 105
##   3   3   3   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
## 106 107 108 109 111 112 113 114 115 116 117 118 119 120 121 123 124 125
126 127
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
## 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145
146 147
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
```

```
## 148 149 150 151 154 155 156 159 160 161 162 163 164 166 167 168 169 170
172 173
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2   2
2   2
## 174 175 176 177 179 180 181 182 183 184 185 186 187 188 189 190 191 192
193 195
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
## 196 197 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214
215 216
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
## 217 218 219 220 221 222 223 224 225 226   4  10  11  13  15  16  17  18
19  20
##   2   2   2   2   2   2   2   2   2   2   3   3   3   3   3   3   3   3
3   3
##  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  40  42
43  44
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
3   3
##  45  46  47  49  50  51  52  54  56  57  59  60  61  62  63  65  67  68
69  71
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
3   3
##  75  76  77  78  79  81  82  83  84
##   3   3   3   3   3   3   3   3   3
##
## Within cluster sum of squares by cluster:
## [1] 3577022 4604811 9487300
##  (between_SS / total_SS =  82.4 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
"tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
grupos_km$cluster ##grupo al cual pertenecen los datos
```

```
##  85  86  87  88  89  91  92  93  94  95  96  97  98  99 100 101 102 103
104 105
##   3   3   3   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
## 106 107 108 109 111 112 113 114 115 116 117 118 119 120 121 123 124 125
126 127
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
## 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145
146 147
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
```

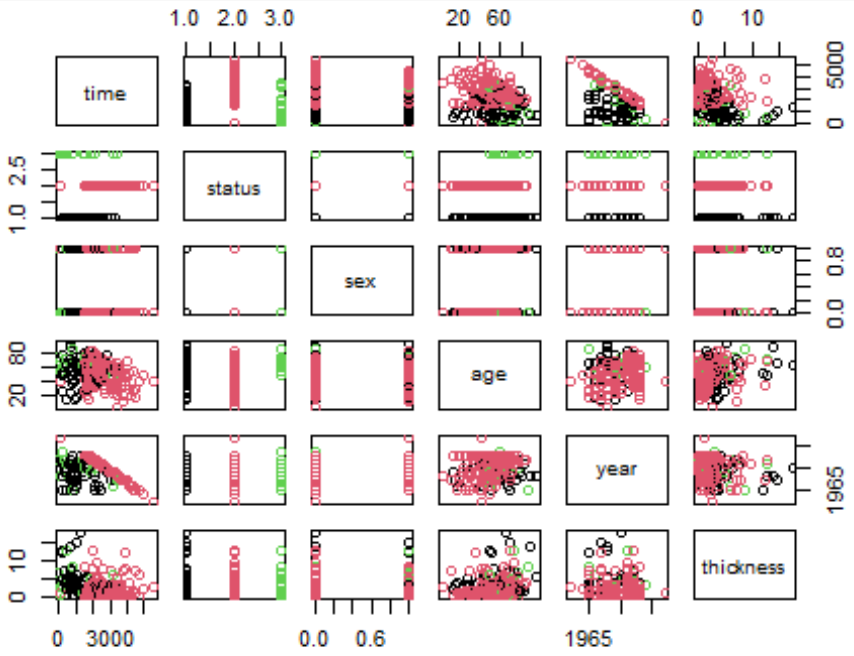
```
1   1
## 148 149 150 151 154 155 156 159 160 161 162 163 164 166 167 168 169 170
172 173
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   2   2
2   2
## 174 175 176 177 179 180 181 182 183 184 185 186 187 188 189 190 191 192
193 195
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
## 196 197 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214
215 216
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
## 217 218 219 220 221 222 223 224 225 226   4  10  11  13  15  16  17  18
19  20
##   2   2   2   2   2   2   2   2   2   2   3   3   3   3   3   3   3   3
3   3
##  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  40  42
43  44
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
3   3
##  45  46  47  49  50  51  52  54  56  57  59  60  61  62  63  65  67  68
69  71
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
3   3
##  75  76  77  78  79  81  82  83  84
##   3   3   3   3   3   3   3   3   3

plot(grupos_km$cluster)
```



Solución del ejercicio 9:

Cargamos el conjunto de datos:

```
##invocamos al conjunto de datos melanoma
data("Melanoma")
head(Melanoma)

##   time status sex age year thickness ulcer
## 1   10      3   1  76 1972      6.76     1
## 2   30      3   1  56 1968      0.65     0
## 3   35      2   1  41 1977      1.34     0
## 4   99      3   0  71 1968      2.90     0
## 5  185      1   1  52 1965     12.08     1
## 6  204      1   1  28 1971      4.84     1

str(Melanoma) ##comprobamos el tipo de variables. Hay 10 variables, todas
ellas numéricas, aunque estatus se podría considerar categórica

## 'data.frame':    205 obs. of  7 variables:
##  $ time     : int  10 30 35 99 185 204 210 232 232 279 ...
##  $ status   : int  3 3 2 3 1 1 1 3 1 1 ...
##  $ sex      : int  1 1 1 0 1 1 1 0 1 0 ...
##  $ age      : int  76 56 41 71 52 28 77 60 49 68 ...
##  $ year     : int  1972 1968 1977 1968 1965 1971 1972 1974 1968 1971 ...
##  $ thickness: num  6.76 0.65 1.34 2.9 12.08 ...
##  $ ulcer    : int  1 0 0 0 1 1 1 1 1 1 ...

##consideramos race nuestra variable objetivo
##realizamos un gráfico de dispersión según la variable predictora
pairs(Melanoma[-7],col=Melanoma$status)
```



```
##comprobamos el grado de correlación entre el resto de variables numéricas
corr_m=cor(Melanoma[-7])
corr_m
```

```
##                 time      status           sex         age         year
## time       1.0000000  0.31614601 -0.146499215 -0.30151794 -0.485504359
## status     0.3161460  1.00000000 -0.098967345  0.01596386  0.138166927
## sex       -0.1464992 -0.09896735  1.000000000  0.06833741 -0.002645159
## age       -0.3015179  0.01596386  0.068337413  1.00000000  0.188229089
## year      -0.4855044  0.13816693 -0.002645159  0.18822909  1.000000000
## thickness -0.2354087 -0.20472162  0.185412563  0.21247979 -0.133345424
##             thickness
## time       -0.2354087
## status     -0.2047216
## sex         0.1854126
## age         0.2124798
## year       -0.1333454
## thickness   1.0000000
```

A partir de estos gráficos podemos intuir qué variables pueden aportar más información para la formación de grupos.

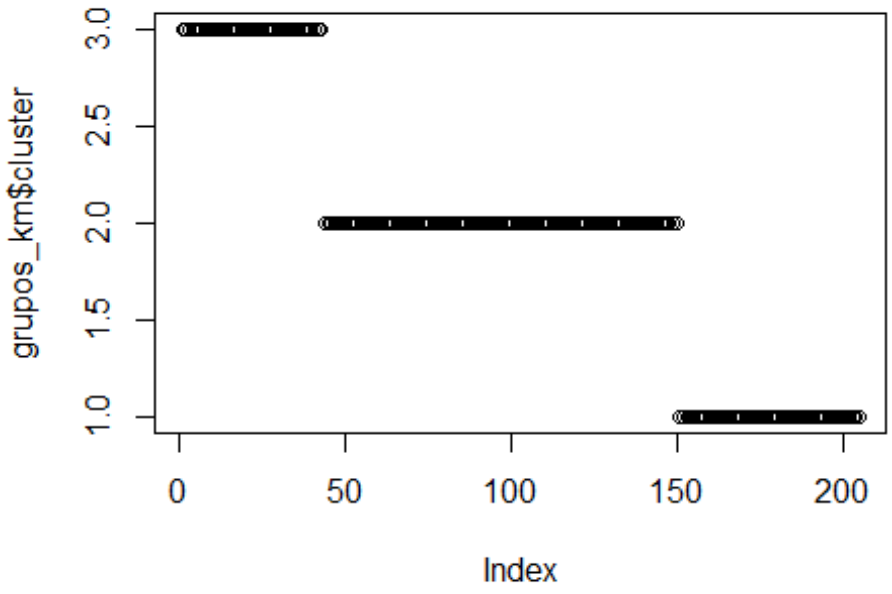
```
##Construimos los grupos de clústeres, crearemos 3 grupos que son los tipos estatus
grupos_km=kmeans(Melanoma[-7],3) ##10 variables, 3 tipos de estatus
grupos_km
```

```
## K-means clustering with 3 clusters of sizes 55, 107, 43
##
## Cluster means:
##        time   status       sex      age     year thickness
## 1 3644.4182 2.018182 0.3454545 44.36364 1967.345  2.446000
## 2 1984.4953 1.850467 0.3457944 54.62617 1971.421  2.224019
## 3  663.7209 1.348837 0.5348837 57.44186 1969.419  5.257442
##
## Clustering vector:
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
19  20
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
3   3
##  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38
39  40
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
3   3
##  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58
59  60
##   3   3   3   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
##  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78
79  80
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
```

```
2   2
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98
99 100
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
## 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118
119 120
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
## 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138
139 140
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
## 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158
159 160
##   2   2   2   2   2   2   2   2   2   2   1   1   1   1   1   1   1   1
1   1
## 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178
179 180
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
## 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198
199 200
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
## 201 202 203 204 205
##   1   1   1   1   1
##
## Within cluster sum of squares by cluster:
## [1] 16820233 13293543  6032057
##  (between_SS / total_SS =  85.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
"tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"

grupos_km$cluster ##grupo al cual pertenecen los datos

##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
19  20
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
3   3
##  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38
39  40
##   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3   3
3   3
##  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58
59  60
```

```
##   3   3   3   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
##  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78
79  80
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
##  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98
99 100
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
## 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118
119 120
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
## 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138
139 140
##   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2   2
2   2
## 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158
159 160
##   2   2   2   2   2   2   2   2   2   2   1   1   1   1   1   1   1   1
1   1
## 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178
179 180
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
## 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198
199 200
##   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1   1
1   1
## 201 202 203 204 205
##   1   1   1   1   1
```

```
plot(grupos_km$cluster)
```



Solución del ejercicio 10:

```
# Cargar el conjunto de datos iris
data(iris)

# Realizar un análisis de varianza (ANOVA) para comparar las longitudes de
pétalos entre especies
modelo_anova <- aov(Petal.Length ~ Species, data = iris)

# Resumen del ANOVA
summary(modelo_anova)

##              Df Sum Sq Mean Sq F value Pr(>F)
## Species       2  437.1  218.55    1180 <2e-16 ***
## Residuals   147   27.2    0.19
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Como podemos ver, existe una diferencia significativa en las longitudes de los pétalos entre al menos un par de especies de Iris («setosa,» «versicolor,» y «virginica»). El valor p es muy pequeño e indica que las diferencias observadas no son aleatorias y son estadísticamente significativas.

Solución caso práctico 1:

```
#Cargamos el paquete survival y, en caso de no estar instalado, sería
necesario instalarlo
library(survival)
data(ovarian)

## Warning in data(ovarian): data set 'ovarian' not found
```

```
#Accediendo a Packages / Survival / Ovarian, podemos visualizar la
descripción de este conjunto de datos referentes a datos de supervivencia de
cáncer de ovario
#Visualizamos algunos registros del conjunto de datos de Ovarian
head(ovarian)
```

```
##   futime fustat     age resid.ds rx ecog.ps
## 1     59      1 72.3315        2  1       1
## 2    115      1 74.4932        2  1       1
## 3    156      1 66.4658        2  1       2
## 4    421      0 53.3644        2  2       1
## 5    431      1 50.3397        2  1       1
## 6    448      0 56.4301        1  1       2
```

```
#Revisamos el tipo de variables de Ovarian y comprobamos que todas ellas son
numéricas
str(ovarian)
```

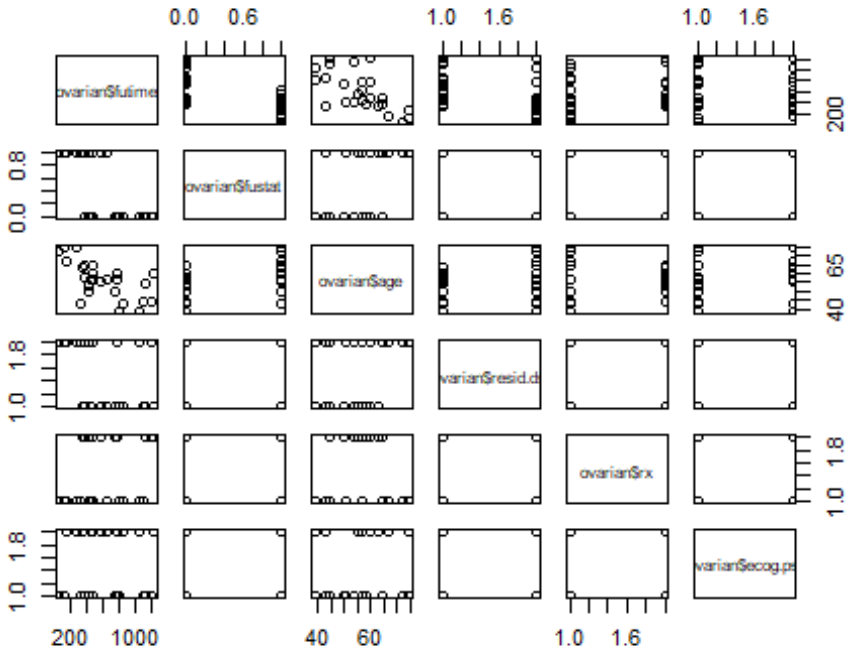
```
## 'data.frame':    26 obs. of  6 variables:
##  $ futime  : num  59 115 156 421 431 448 464 475 477 563 ...
##  $ fustat  : num  1 1 1 0 1 0 1 1 0 1 ...
##  $ age     : num  72.3 74.5 66.5 53.4 50.3 ...
##  $ resid.ds: num  2 2 2 2 2 1 2 2 2 1 ...
##  $ rx      : num  1 1 1 2 1 1 2 2 1 2 ...
##  $ ecog.ps : num  1 1 2 1 1 2 2 2 1 2 ...
```

```
#Mostramos un resumen estadístico de Ovarian
summary(ovarian)
```

```
##      futime           fustat            age           resid.ds
##  Min.   :  59.0   Min.   :0.0000   Min.   :38.89   Min.   :1.000
##  1st Qu.: 368.0   1st Qu.:0.0000   1st Qu.:50.17   1st Qu.:1.000
##  Median : 476.0   Median :0.0000   Median :56.85   Median :2.000
##  Mean   : 599.5   Mean   :0.4615   Mean   :56.17   Mean   :1.577
##  3rd Qu.: 794.8   3rd Qu.:1.0000   3rd Qu.:62.38   3rd Qu.:2.000
##  Max.   :1227.0   Max.   :1.0000   Max.   :74.50   Max.   :2.000
##        rx         ecog.ps
##  Min.   :1.0   Min.   :1.000
##  1st Qu.:1.0   1st Qu.:1.000
##  Median :1.5   Median :1.000
##  Mean   :1.5   Mean   :1.462
##  3rd Qu.:2.0   3rd Qu.:2.000
##  Max.   :2.0   Max.   :2.000
```

Realizamos los diagramas de puntos de la relación entre variables:

```
pairs(~ovarian$futime+ovarian$fustat+ovarian$age+ovarian$resid.ds+ovarian
$rx+ovarian$ecog.ps)
```



Este gráfico puede aportar una idea sobre qué variables podrían tener alguna relación; parece intuirse que *age* con *futime*. A continuación, calculamos el coeficiente de correlación de estas dos variables:

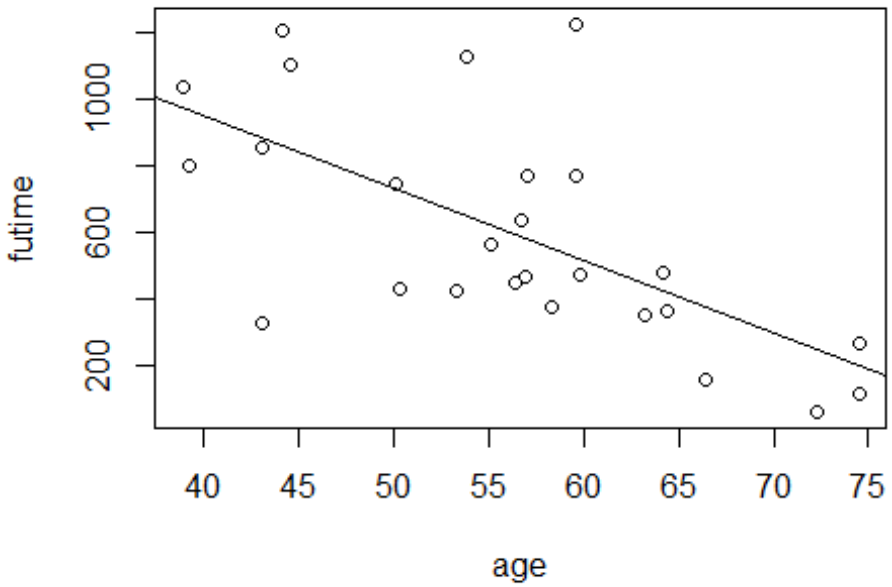
```
#Coeficiente de correlación age vs futime que hace referencia a la edad y el
tiempo de supervivencia
cor(ovarian[,c("age","futime")], use="complete")

##                age     futime
## age     1.0000000 -0.6483612
## futime -0.6483612  1.0000000
```

Vemos que el grado de correlación es de -0.648, que es un valor bastante aceptable para indicar que existe alguna correlación entre ambas variables. Generamos el modelo lineal correspondiente a la estimación por mínimos cuadrados:

```
RegModel.2<-lm(futime~age, data=ovarian)
summary(RegModel.2)

##
## Call:
## lm(formula = futime ~ age, data = ovarian)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -554.63 -160.13  -49.27   67.21  702.11
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1824.240    298.079   6.120 2.54e-06 ***
## age          -21.805      5.227  -4.172 0.000341 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 263.9 on 24 degrees of freedom
## Multiple R-squared:  0.4204, Adjusted R-squared:  0.3962
## F-statistic: 17.41 on 1 and 24 DF, p-value: 0.0003408

lm(formula = futime ~age, data = ovarian)

##
## Call:
## lm(formula = futime ~ age, data = ovarian)
##
## Coefficients:
## (Intercept)          age
##     1824.24       -21.81
```

El valor de *R-squared* nos da información acerca del grado de ajuste del modelo. Este será mayor cuanto más próximo a 1. En nuestro caso, dicho ajuste es del 0.4204. Por otra parte, el p-value es inferior a 0.05, lo cual parece indicar que pudiera haber una relación entre las dos variables. Dibujamos la recta de mínimos cuadrados:

```
plot(ovarian$age,ovarian$futime, xlab="age", ylab="futime")
abline(RegModel.2)
```



La anterior representación nos sigue reforzando la idea de que existe una correlación entre la variable *age* y *futime*. Realicemos ahora un modelo de regresión múltiple para comprobar si la supervivencia del cáncer de ovario viene determinada por algún factor más. Para ello, inicialmente calculamos el coeficiente de correlación de Pearson para las distintas combinaciones:

```
cor(x=ovarian,method="pearson")
```

```
##                futime     fustat         age    resid.ds          rx
ecog.ps
## futime    1.00000000 -0.6898795 -0.64836121 -0.38518677  0.24687108
0.01425371
## fustat   -0.68987946  1.0000000  0.49586611  0.32433749 -0.15430335
0.22619048
## age      -0.64836121  0.4958661  1.00000000  0.29339868  0.04372768
0.12927600
## resid.ds -0.38518677  0.3243375  0.29339868  1.00000000 -0.07784989 -
0.14414999
## rx        0.24687108 -0.1543033  0.04372768 -0.07784989  1.00000000
0.00000000
## ecog.ps   0.01425371  0.2261905  0.12927600 -0.14414999  0.00000000
1.00000000
```

De los datos anteriores parece derivarse que la variable *fustat* también presenta alguna relación con la variable *futime*.

```
mrm_ovarian<-lm(ovarian$futime
~ovarian$fustat+ovarian$age+ovarian$resid.ds+ovarian$rx+ovarian$ecog.ps,data=
ovarian)
summary(mrm_ovarian)
```

```
##
## Call:
```

```
## lm(formula = ovarian$futime ~ ovarian$fustat + ovarian$age +
##     ovarian$resid.ds + ovarian$rx + ovarian$ecog.ps, data = ovarian)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -374.48 -103.23  -12.18   77.76  384.41
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1276.931    327.157   3.903 0.000882 ***
## ovarian$fustat   -309.207    105.682  -2.926 0.008357 **
## ovarian$age       -14.373      5.083  -2.828 0.010401 *
## ovarian$resid.ds  -48.136     95.331  -0.505 0.619126
## ovarian$rx        125.638     87.254   1.440 0.165364
## ovarian$ecog.ps   109.503     90.484   1.210 0.240315
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 217.3 on 20 degrees of freedom
## Multiple R-squared:  0.6727, Adjusted R-squared:  0.5909
## F-statistic: 8.223 on 5 and 20 DF,  p-value: 0.0002355
```

El modelo anterior generado, una vez que hemos supuesto todas las variables como predictores, resulta un coeficiente de determinación *Multiple R-squared* bastante aceptable (0.6727), y el p-value también es significativo. Ahora, de entre todos los predictores posibles, deberíamos seleccionar los mejores. Para ello, hacemos lo siguiente:

```
step(object=mrm_ovarian,direction ="both", trace=1)
```

```
## Start:  AIC=285
## ovarian$futime ~ ovarian$fustat + ovarian$age + ovarian$resid.ds +
##     ovarian$rx + ovarian$ecog.ps
##
##                    Df Sum of Sq     RSS    AIC
## - ovarian$resid.ds  1     12034  956070 283.32
## - ovarian$ecog.ps   1     69130 1013166 284.83
## <none>                          944036 285.00
## - ovarian$rx        1     97865 1041901 285.56
## - ovarian$age       1    377400 1321436 291.74
## - ovarian$fustat    1    404069 1348105 292.26
##
## Step:  AIC=283.32
## ovarian$futime ~ ovarian$fustat + ovarian$age + ovarian$rx +
##     ovarian$ecog.ps
##
##                    Df Sum of Sq     RSS    AIC
## <none>                          956070 283.32
## - ovarian$ecog.ps   1     88920 1044991 283.64
## - ovarian$rx        1    101328 1057398 283.94
## + ovarian$resid.ds  1     12034  944036 285.00
```

```
## - ovarian$age       1    414281 1370351 290.69
## - ovarian$fustat    1    467936 1424007 291.68

##
## Call:
## lm(formula = ovarian$futime ~ ovarian$fustat + ovarian$age +
##     ovarian$rx + ovarian$ecog.ps, data = ovarian)
##
## Coefficients:
##     (Intercept)   ovarian$fustat      ovarian$age       ovarian$rx
##         1213.26          -322.41           -14.82           127.70
## ovarian$ecog.ps
##          120.52
```

Las variables potenciales de ser predictores son las especificadas en el anterior Call. Ejecutemos de nuevo el modelo:

```
mrm_ovarian_predictores<-(lm(formula = ovarian$futime
~ovarian$fustat+ovarian$age +
ovarian$rx +ovarian$ecog.ps, data = ovarian))
summary(mrm_ovarian_predictores)

##
## Call:
## lm(formula = ovarian$futime ~ ovarian$fustat + ovarian$age +
##     ovarian$rx + ovarian$ecog.ps, data = ovarian)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -377.09  -94.77  -16.40   73.50  400.66
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1213.263    296.477   4.092 0.000521 ***
## ovarian$fustat  -322.407    100.565  -3.206 0.004244 **
## ovarian$age      -14.824      4.914  -3.017 0.006568 **
## ovarian$rx       127.701     85.599   1.492 0.150606
## ovarian$ecog.ps  120.524     86.240   1.398 0.176842
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 213.4 on 21 degrees of freedom
## Multiple R-squared:  0.6686, Adjusted R-squared:  0.6054
## F-statistic: 10.59 on 4 and 21 DF, p-value: 7.383e-05
```

En teoría, y con los datos de que disponemos, este sería el modelo que explicaría qué factores influyen en la supervivencia del cáncer de ovario.

Solución caso práctico 2:

Cargamos el conjunto de datos:

```
#Cargamos el paquete survival y el conjunto de datos lung
library(survival)
data("lung")

## Warning in data("lung"): data set 'lung' not found

#En Packages / Survival / lung podemos visualizar la descripción
#Visualizamos los primeros registros
head(lung)

##   inst time status age sex ph.ecog ph.karno pat.karno meal.cal wt.loss
## 1    3  306      2  74   1       1       90       100     1175      NA
## 2    3  455      2  68   1       0       90        90     1225      15
## 3    3 1010      1  56   1       0       90        90       NA      15
## 4    5  210      2  57   1       1       90        60     1150      11
## 5    1  883      2  60   1       0      100        90       NA       0
## 6   12 1022      1  74   1       1       50        80      513       0
```

Revisamos la estructura de datos:

```
#Analizamos el tipo de variables y observamos que todas son numéricas
str(lung)

## 'data.frame':    228 obs. of  10 variables:
##  $ inst     : num  3 3 3 5 1 12 7 11 1 7 ...
##  $ time     : num  306 455 1010 210 883 ...
##  $ status   : num  2 2 1 2 2 1 2 2 2 2 ...
##  $ age      : num  74 68 56 57 60 74 68 71 53 61 ...
##  $ sex      : num  1 1 1 1 1 1 2 2 1 1 ...
##  $ ph.ecog  : num  1 0 0 1 0 1 2 2 1 2 ...
##  $ ph.karno : num  90 90 90 90 100 50 70 60 70 70 ...
##  $ pat.karno: num  100 90 90 60 90 80 60 80 80 70 ...
##  $ meal.cal : num  1175 1225 NA 1150 NA ...
##  $ wt.loss  : num  NA 15 15 11 0 0 10 1 16 34 ...

#Eliminamos los valores NA
data_lung=na.omit(lung)

#Realizamos un breve resumen estadístico
summary(lung)

##       inst             time           status           age       
##  Min.   : 1.00   Min.   :   5.0   Min.   :1.000   Min.   :39.00  
##  1st Qu.: 3.00   1st Qu.: 166.8   1st Qu.:1.000   1st Qu.:56.00  
##  Median :11.00   Median : 255.5   Median :2.000   Median :63.00  
##  Mean   :11.09   Mean   : 305.2   Mean   :1.724   Mean   :62.45  
##  3rd Qu.:16.00   3rd Qu.: 396.5   3rd Qu.:2.000   3rd Qu.:69.00  
##  Max.   :33.00   Max.   :1022.0   Max.   :2.000   Max.   :82.00
```

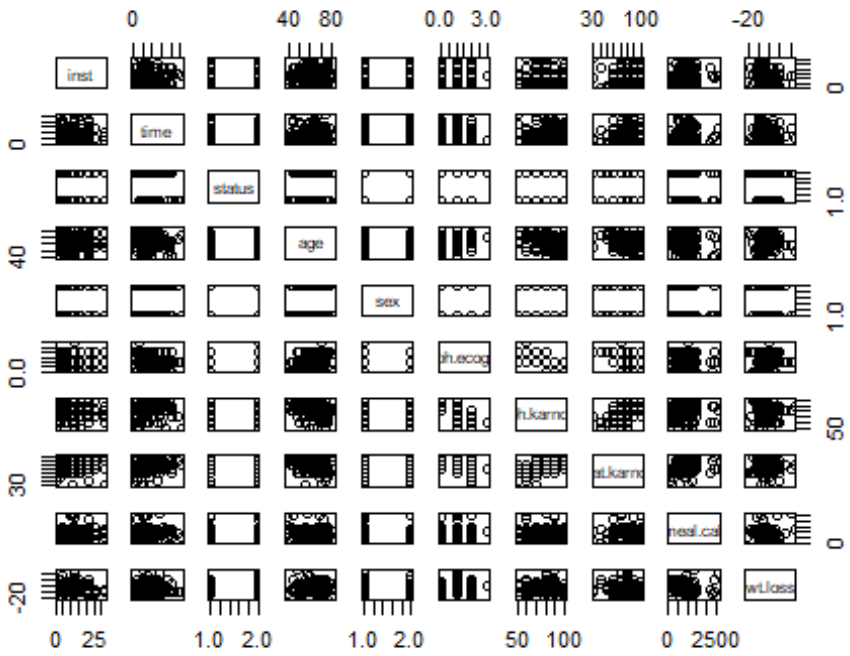
```
##  NA's   :1
##       sex           ph.ecog          ph.karno        pat.karno
##  Min.   :1.000   Min.   :0.0000   Min.   : 50.00   Min.   : 30.00
##  1st Qu.:1.000   1st Qu.:0.0000   1st Qu.: 75.00   1st Qu.: 70.00
##  Median :1.000   Median :1.0000   Median : 80.00   Median : 80.00
##  Mean   :1.395   Mean   :0.9515   Mean   : 81.94   Mean   : 79.96
##  3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.: 90.00   3rd Qu.: 90.00
##  Max.   :2.000   Max.   :3.0000   Max.   :100.00   Max.   :100.00
##                  NA's   :1        NA's   :1        NA's   :3
##     meal.cal         wt.loss
##  Min.   :  96.0   Min.   :-24.000
##  1st Qu.: 635.0   1st Qu.:  0.000
##  Median : 975.0   Median :  7.000
##  Mean   : 928.8   Mean   :  9.832
##  3rd Qu.:1150.0   3rd Qu.: 15.750
##  Max.   :2600.0   Max.   : 68.000
##  NA's   :47       NA's   :14
```

Una vez que tengamos una primera vista del conjunto de datos *lung*, procederemos a valorar si existe alguna correlación clara entre las variables de este conjunto de datos.

```
#calculamos la matriz de correlaciones
cor(x=data_lung,method="pearson")
```

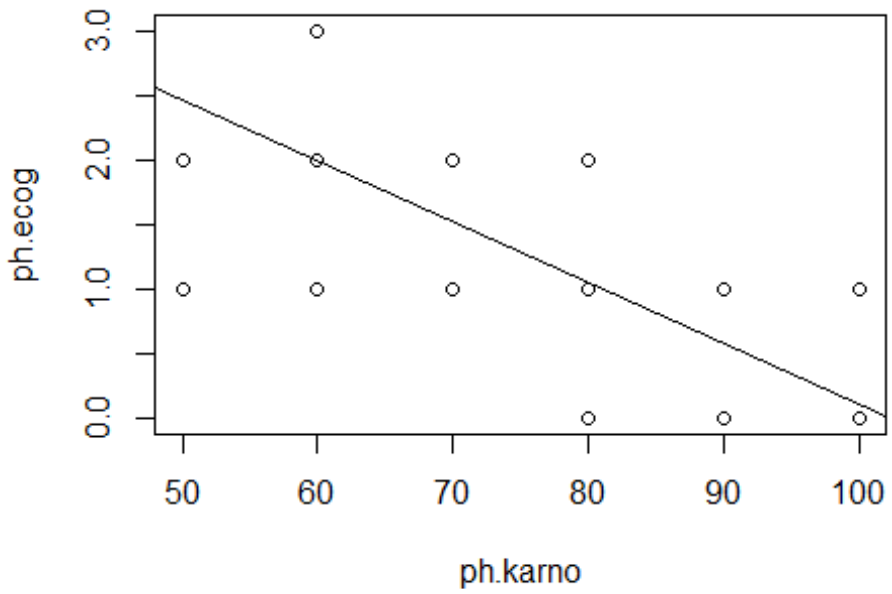
```
##                  inst        time      status         age          sex
## inst       1.00000000  0.02462876 -0.12719801  0.04859452  0.084362910
## time       0.02462876  1.00000000 -0.16217237 -0.07854153  0.114149616
## status    -0.12719801 -0.16217237  1.00000000  0.15911933 -0.218780030
## age        0.04859452 -0.07854153  0.15911933  1.00000000 -0.125280356
## sex        0.08436291  0.11414962 -0.21878003 -0.12528036  1.000000000
## ph.ecog    0.05947203 -0.19116847  0.23805821  0.30865378 -0.005363288
## ph.karno  -0.02252266  0.09487913 -0.16127595 -0.32261297 -0.019623924
## pat.karno  0.04147893  0.17505701 -0.18542442 -0.23989736  0.071014942
## meal.cal   0.09869124  0.07467151  0.02483564 -0.23958240 -0.171044801
## wt.loss   -0.17485406  0.03342528  0.04868879  0.04286056 -0.169892775
##                ph.ecog    ph.karno   pat.karno    meal.cal     wt.loss
## inst       0.059472025 -0.02252266  0.04147893  0.09869124 -0.17485406
## time      -0.191168469  0.09487913  0.17505701  0.07467151  0.03342528
## status     0.238058213 -0.16127595 -0.18542442  0.02483564  0.04868879
## age        0.308653782 -0.32261297 -0.23989736 -0.23958240  0.04286056
## sex       -0.005363288 -0.01962392  0.07101494 -0.17104480 -0.16989278
## ph.ecog    1.000000000 -0.82269739 -0.54719617 -0.10563039  0.17125740
## ph.karno  -0.822697393  1.00000000  0.53502749  0.05385409 -0.12524032
## pat.karno -0.547196168  0.53502749  1.00000000  0.17465190 -0.18213953
## meal.cal  -0.105630385  0.05385409  0.17465190  1.00000000 -0.11134425
## wt.loss    0.171257402 -0.12524032 -0.18213953 -0.11134425  1.00000000
```

```
#Representamos la relación entre las variables
pairs(data_lung)
```



Revisando los coeficientes de Pearson de cada par de variables, quizá dos variables sobre las que podemos calcular el modelo de regresión lineal son ph.ecog y ph.karno.

```
#Realizamos el modelo de regresión lineal
lung_rl=lm(formula=ph.ecog ~ph.karno,data=data_lung)
#Mostramos el resultado
summary(lung_rl)

##
## Call:
## lm(formula = ph.ecog ~ ph.karno, data = data_lung)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4658 -0.1127 -0.0539  0.4167  1.0049
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.818865   0.210176   22.93   <2e-16 ***
## ph.karno    -0.047062   0.002532  -18.59   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4168 on 165 degrees of freedom
## Multiple R-squared:  0.6768, Adjusted R-squared:  0.6749
## F-statistic: 345.6 on 1 and 165 DF, p-value: < 2.2e-16

#Representamos el diagrama de puntos
plot(data_lung$ph.karno,data_lung$ph.ecog,xlab="ph.karno",ylab="ph.ecog")
abline(lung_rl)
```



El modelo de regresión muestra un *R-squared* de 0.6768, lo cual indica un buen ajuste de dicho modelo. A continuación, realizaremos el modelo de regresión múltiple evaluando cuáles son los posibles predictores para la variable ph.ecog, para ello, ejecutamos las siguientes instrucciones:

```
#Modelo de regresión múltiple
lung_rm<-lm(data_lung$ph.ecog
~data_lung$inst+data_lung$time+data_lung$status+data_lung$age+data_lung$ph.ka
rno+data_lung$pat.karno+data_lung$meal.cal+data_lung$wt.loss,data=data_lung)
summary(lung_rm)

##
## Call:
## lm(formula = data_lung$ph.ecog ~ data_lung$inst + data_lung$time +
##     data_lung$status + data_lung$age + data_lung$ph.karno +
data_lung$pat.karno +
##     data_lung$meal.cal + data_lung$wt.loss, data = data_lung)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.07182 -0.20748  0.00247  0.29444  0.92569
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           4.523e+00  4.099e-01  11.033   <2e-16 ***
## data_lung$inst        6.881e-03  3.901e-03   1.764   0.0797 .
## data_lung$time       -3.099e-04  1.516e-04  -2.044   0.0426 *
## data_lung$status      1.499e-01  7.177e-02   2.089   0.0383 *
## data_lung$age         8.339e-04  3.686e-03   0.226   0.8213
## data_lung$ph.karno   -4.154e-02  2.963e-03 -14.020   <2e-16 ***
## data_lung$pat.karno -5.185e-03  2.514e-03  -2.063   0.0408 *
## data_lung$meal.cal   -7.251e-05  7.894e-05  -0.919   0.3598
```

```
## data_lung$wt.loss    3.698e-03  2.394e-03   1.545   0.1244
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3973 on 158 degrees of freedom
## Multiple R-squared:  0.7188, Adjusted R-squared:  0.7046
## F-statistic: 50.49 on 8 and 158 DF, p-value: < 2.2e-16
```

Una vez calculados todos los coeficientes de los posibles predictores, evaluamos cuáles son los mejores predictores:

```
#Analizamos los mejores predictores
step(object=lung_rm,direction="both",trace=1)
```

```
## Start:  AIC=-299.54
## data_lung$ph.ecog ~ data_lung$inst + data_lung$time + data_lung$status +
##     data_lung$age + data_lung$ph.karno + data_lung$pat.karno +
##     data_lung$meal.cal + data_lung$wt.loss
##
##                       Df Sum of Sq    RSS     AIC
## - data_lung$age        1    0.0081 24.951 -301.48
## - data_lung$meal.cal   1    0.1332 25.076 -300.65
## <none>                             24.942 -299.54
## - data_lung$wt.loss    1    0.3767 25.319 -299.03
## - data_lung$inst       1    0.4911 25.434 -298.28
## - data_lung$time       1    0.6597 25.602 -297.18
## - data_lung$pat.karno  1    0.6716 25.614 -297.10
## - data_lung$status     1    0.6888 25.631 -296.99
## - data_lung$ph.karno   1   31.0307 55.973 -166.55
##
## Step:  AIC=-301.48
## data_lung$ph.ecog ~ data_lung$inst + data_lung$time + data_lung$status +
##     data_lung$ph.karno + data_lung$pat.karno + data_lung$meal.cal +
##     data_lung$wt.loss
##
##                       Df Sum of Sq    RSS     AIC
## - data_lung$meal.cal   1     0.158 25.109 -302.43
## <none>                             24.951 -301.48
## - data_lung$wt.loss    1     0.375 25.326 -300.99
## - data_lung$inst       1     0.506 25.456 -300.13
## + data_lung$age        1     0.008 24.942 -299.54
## - data_lung$time       1     0.661 25.612 -299.11
## - data_lung$pat.karno  1     0.677 25.628 -299.01
## - data_lung$status     1     0.720 25.671 -298.73
## - data_lung$ph.karno   1    33.216 58.166 -162.13
##
## Step:  AIC=-302.43
## data_lung$ph.ecog ~ data_lung$inst + data_lung$time + data_lung$status +
##     data_lung$ph.karno + data_lung$pat.karno + data_lung$wt.loss
##
```

```
##                         Df Sum of Sq    RSS     AIC
## <none>                                25.109 -302.43
## - data_lung$wt.loss      1     0.414 25.523 -301.70
## + data_lung$meal.cal     1     0.158 24.951 -301.48
## - data_lung$inst         1     0.462 25.571 -301.38
## + data_lung$age          1     0.033 25.076 -300.65
## - data_lung$status       1     0.674 25.783 -300.00
## - data_lung$time         1     0.703 25.812 -299.82
## - data_lung$pat.karno    1     0.802 25.911 -299.17
## - data_lung$ph.karno     1    33.085 58.194 -164.05

##
## Call:
## lm(formula = data_lung$ph.ecog ~ data_lung$inst + data_lung$time +
##     data_lung$status + data_lung$ph.karno + data_lung$pat.karno +
##     data_lung$wt.loss, data = data_lung)
##
## Coefficients:
##         (Intercept)       data_lung$inst       data_lung$time
##           4.5535875            0.0066252           -0.0003192
##    data_lung$status   data_lung$ph.karno  data_lung$pat.karno
##           0.1466834           -0.0415905           -0.0055947
##   data_lung$wt.loss
##           0.0038653
```

En el resultado del *call* anterior se aprecian cuáles serían los mejores predictores. Calculamos de nuevo el modelo de regresión sobre estos predictores:

```
#Modelo de regresión con los mejores predictores
lung_rm_p<-lm(formula = data_lung$ph.ecog ~data_lung$inst
+data_lung$time+data_lung$status +data_lung$ph.karno +data_lung$pat.karno
+data_lung$wt.loss, data =data_lung)
summary(lung_rm_p)
```

```
##
## Call:
## lm(formula = data_lung$ph.ecog ~ data_lung$inst + data_lung$time +
##     data_lung$status + data_lung$ph.karno + data_lung$pat.karno +
##     data_lung$wt.loss, data = data_lung)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.16209 -0.22765 -0.01216  0.30052  0.92432
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          4.5535875  0.2816769  16.166   <2e-16 ***
## data_lung$inst       0.0066252  0.0038613   1.716   0.0881 .
## data_lung$time      -0.0003192  0.0001509  -2.116   0.0359 *
## data_lung$status     0.1466834  0.0707557   2.073   0.0398 *
## data_lung$ph.karno  -0.0415905  0.0028644 -14.520   <2e-16 ***
```

```
## data_lung$pat.karno -0.0055947  0.0024744  -2.261   0.0251 *
## data_lung$wt.loss    0.0038653  0.0023803   1.624   0.1064
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3961 on 160 degrees of freedom
## Multiple R-squared:  0.7169, Adjusted R-squared:  0.7063
## F-statistic: 67.54 on 6 and 160 DF, p-value: < 2.2e-16
```

Se observa que el ajuste del modelo *R-squared* (0.7169) se ha incrementado respecto a anteriores cálculos. Realicemos ahora el test ANOVA para las variables *sex* y *time*.

```
#definimos los vectores de las variables sex y time
sex_lung<-c(data_lung$sex)
time_lung<-c(data_lung$time)
#definimos el data frame que guarde estos dos vectores
df_lung<-data.frame(time_lung,sex_lung)
#mostramos los primeros registros
head(df_lung)
```

```
##   time_lung sex_lung
## 1       455        1
## 2       210        1
## 3      1022        1
## 4       310        2
## 5       361        2
## 6       218        1
```

```
#mostramos el número de observaciones por cada grupo (sexo)
table(df_lung$sex_lung)
```

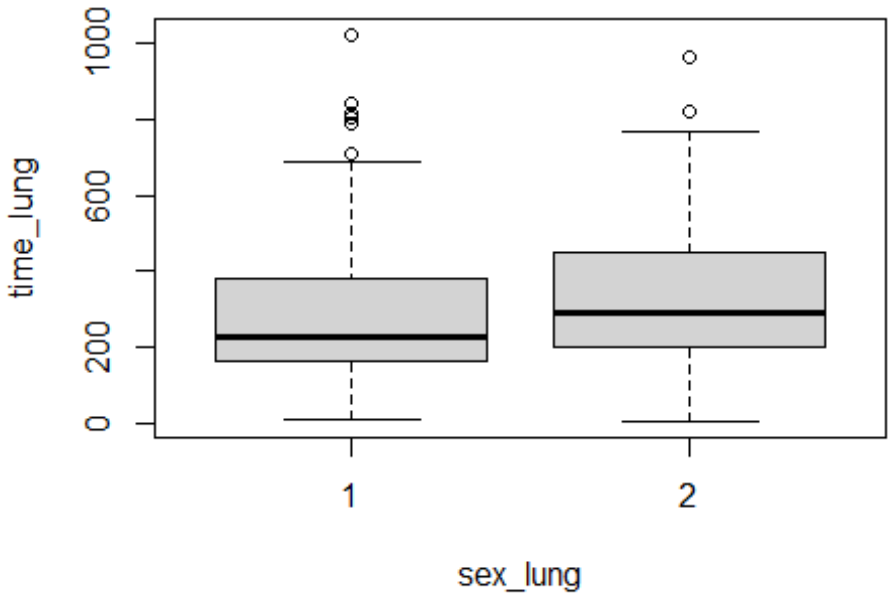
```
##
##   1   2
## 103  64
```

Se observa que el número de observaciones es desigual en cada grupo. Veamos la media del tiempo de supervivencia en función del sexo.

```
#Calculamos la media del tiempo de supervivencia en función del ph.ecog
aggregate(time_lung~sex_lung,data=df_lung,FUN=mean)
```

```
##   sex_lung time_lung
## 1        1  291.1456
## 2        2  340.1719
```

```
#Calculamos el diagrama de cajas
boxplot(time_lung~sex_lung,data=df_lung,id.method="y")
```



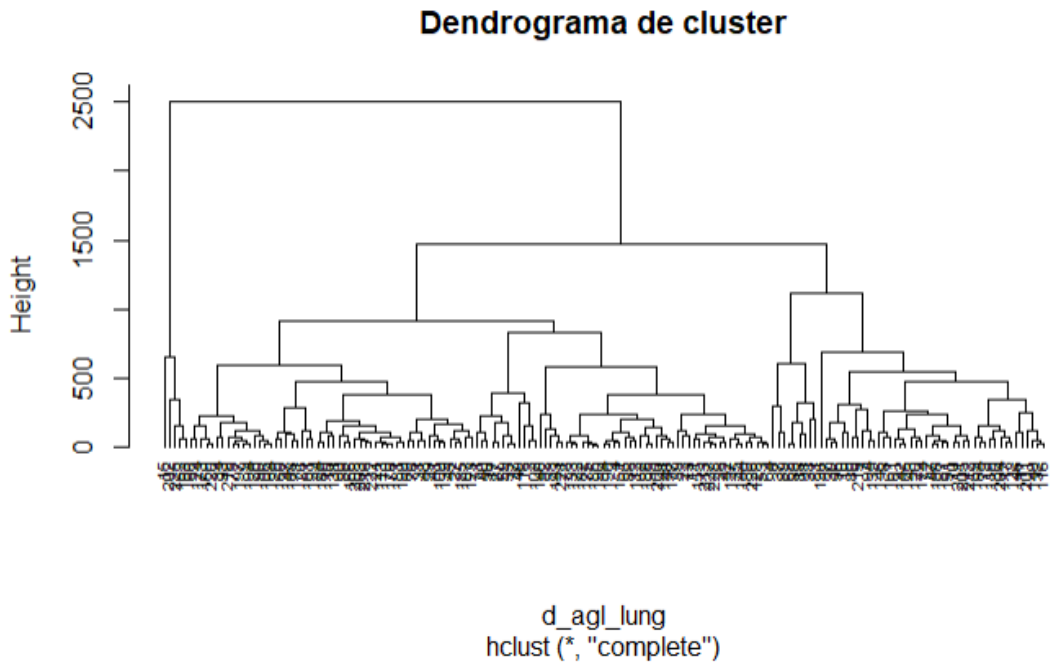
Para determinar la normalidad aplicamos el test de Kolmogorov-Smirnov, ya que disponemos de más de 50 observaciones.

```
require(nortest)
by(data=df_lung,INDICES =
df_lung$sex_lung,FUN=function(x){lillie.test(x$time_lung)})
```

```
## df_lung$sex_lung: 1
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  x$time_lung
## D = 0.16663, p-value = 1.904e-07
##
## ------------------------------------------------------------
## df_lung$sex_lung: 2
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  x$time_lung
## D = 0.12018, p-value = 0.02246
```

Se puede observar que ambos grupos tienen un p-value inferior a 0.05, con lo cual se determina que ninguno de los dos grupos cumple las condiciones de normalidad. Por tanto, no tiene sentido realizar el test de ANOVA. Por último, realizaremos un test de *clustering* y, para ello, comenzamos con un agrupamiento jerárquico aglomerativo.

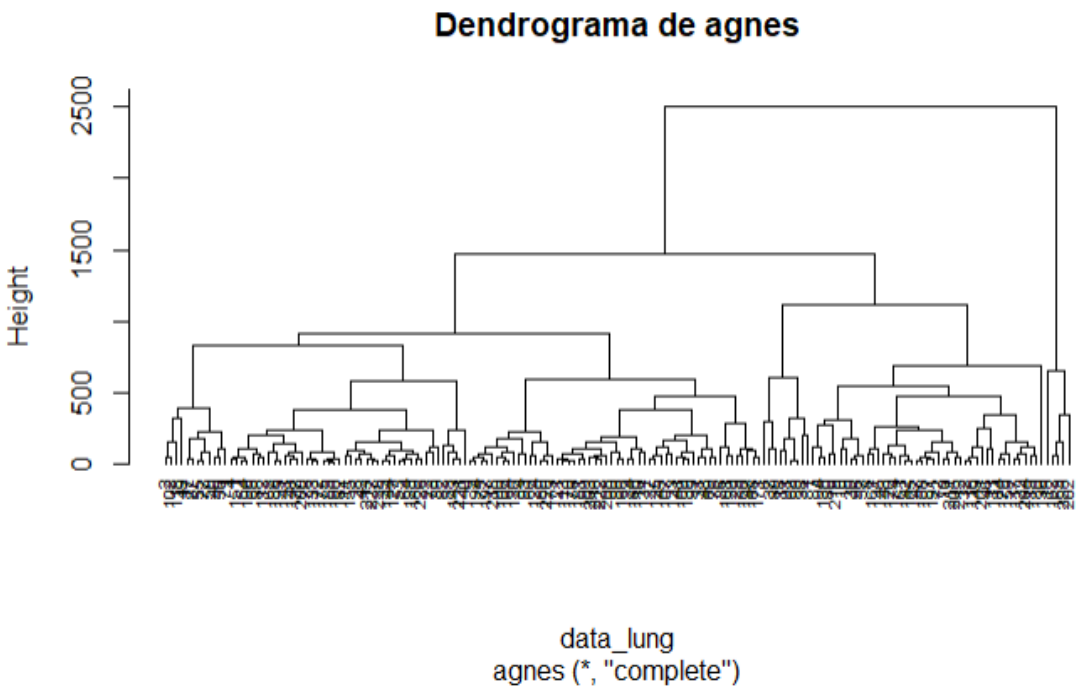
```
##Agrupamiento jerárquico aglomerativo. Cada observación es un clúster que se va organizando hasta converger en una única rama central.
library(MASS)
library(cluster)
d_agl_lung<-dist(data_lung,method="euclidean") ##especificamos los valores de distancia
hc_agl_lung<-hclust(d_agl_lung,method="complete") ##calculamos el clúster jerárquico
plot(hc_agl_lung,cex=0.6,hang=-1, main ="Dendrograma de cluster")
```




```
##representamos el dendrograma
```

Ahora realizamos un agrupamiento jerárquico aplicando la función *agnes()*:

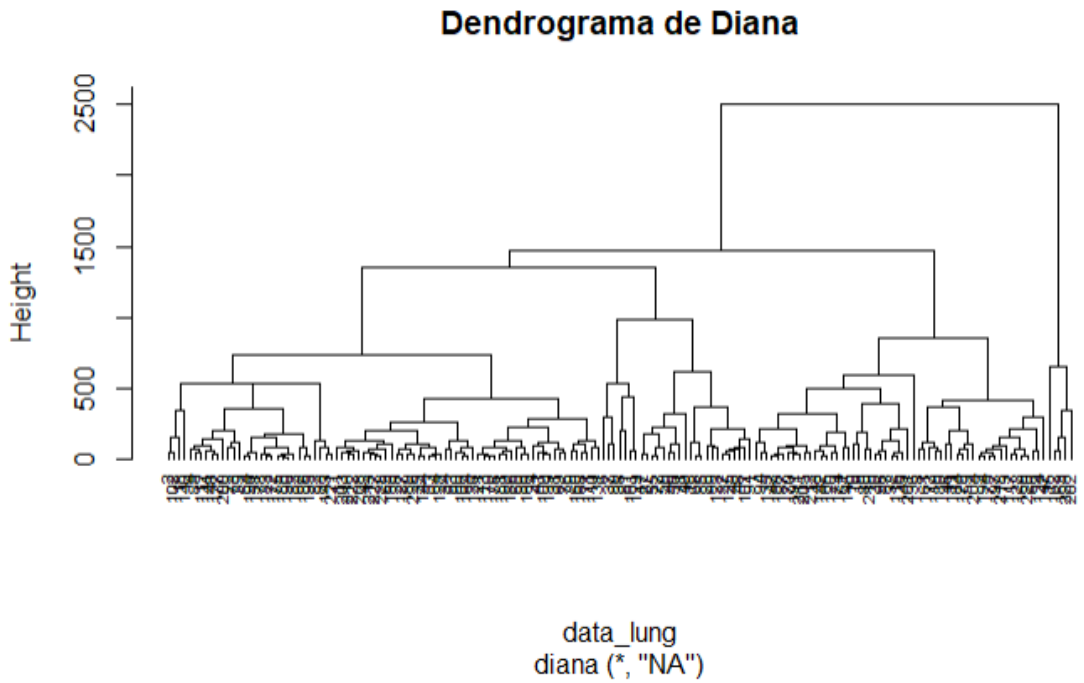
```
hc_ag_lung<-agnes(data_lung,method="complete") ##calculamos los clústeres jerárquicos
pltree(hc_ag_lung,cex=0.6,hang=-1,main="Dendrograma de agnes")
```




```
##representamos dendrograma

hc_ag_lung$ac ##calculamos el coeficiente de aglomeración

## [1] 0.9723202
```

El coeficiente es próximo a 1, con lo que la agrupación es fuerte. Ahora repetimos el proceso utilizando la función diana.

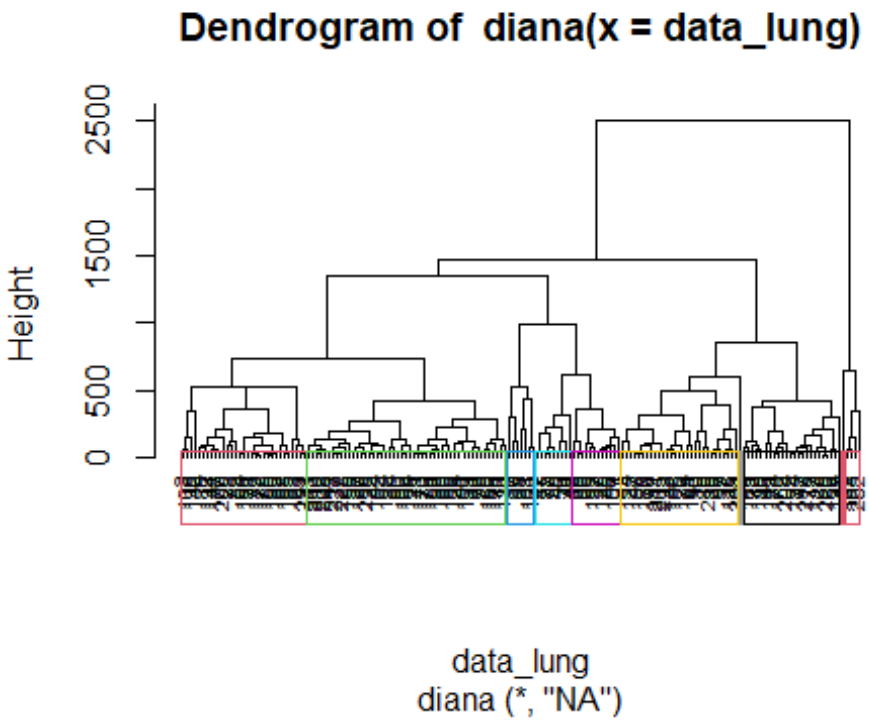
```
##Agrupamiento divisional jerárquico
hc_div_lung<-diana(data_lung) ##calculamos los clústeres jerárquicos
pltree(hc_div_lung,cex=0.6,hang=-1,main="Dendrograma de Diana")
```




```
##representamos el dendograma

hc_div_lung$dc ##calculamos el coeficiente de división

## [1] 0.9705904
```

El coeficiente también es muy próximo a 1. Para asociar los agrupamientos a los datos, hacemos:

```
pltree(hc_div_lung,hang=-1,cex=0.6)
rect.hclust(hc_div_lung,k=10,border=2:10)
```



Solución caso práctico 3:

Definimos el conjunto de datos del estudio que realizar y extraemos información de este:

```
#Definición de los datos de estudio de birthwt
library(MASS)
data_birth<-MASS::birthwt
data_birth$race<-
factor(data_birth$race,levels=c(1,2,3),labels=c("white","black","other"))
#1->white,2->black,3->other
head(data_birth)

##    low age lwt  race smoke ptl ht ui ftv  bwt
## 85   0  19 182 black     0   0  0  1   0 2523
## 86   0  33 155 other     0   0  0  0   3 2551
## 87   0  20 105 white     1   0  0  0   1 2557
## 88   0  21 108 white     1   0  0  1   2 2594
## 89   0  18 107 white     1   0  0  1   0 2600
## 91   0  21 124 other     0   0  0  0   0 2622

raza<-c(data_birth$race)
peso<-c(data_birth$bwt)
df_birth_anova<-data.frame(raza,peso) #definimos el conjunto de datos
head(df_birth_anova,4) #mostramos la información del data frame

##    raza peso
## 1 black 2523
## 2 other 2551
```

```
## 3 white 2557
## 4 white 2594

table(df_birth_anova$raza) #número de observaciones por cada grupo

##
## white black other
##    96    26    67
```

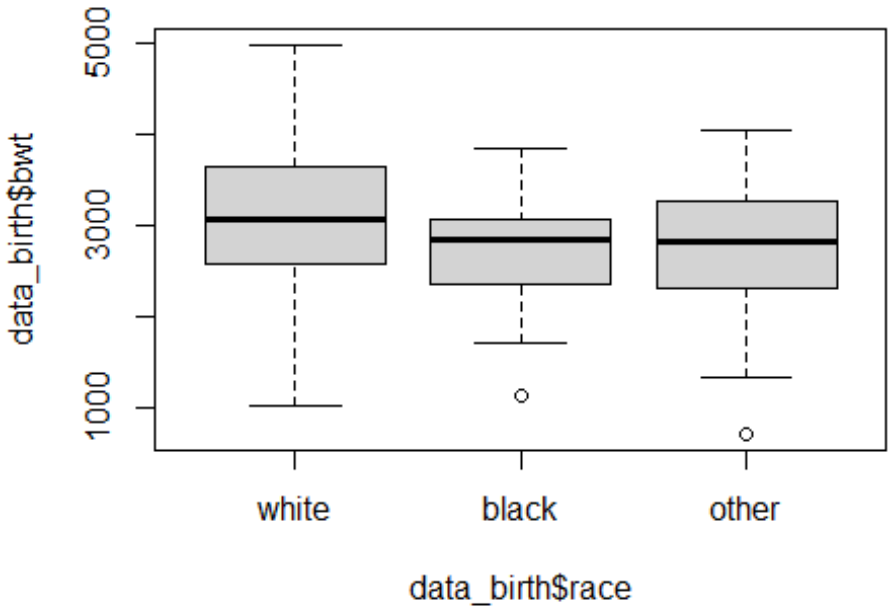
El número de observaciones no es constante para cada grupo y, entre alguno de ellos, las diferencias son significativas.

Calculemos ahora la media y desviación típica por cada grupo.

```
##Media de pesos por raza
aggregate(peso~raza,data=df_birth_anova,FUN=mean)

##    raza     peso
## 1 white 3102.719
## 2 black 2719.692
## 3 other 2805.284

##Desviación típica de pesos por raza
aggregate(peso~raza,data=df_birth_anova,FUN=sd)

##    raza     peso
## 1 white 727.8861
## 2 black 638.6839
## 3 other 722.1944
```

Veamos en una representación gráfica si pudieran existir asimetrías, datos atípicos u otras diferencias significativas:

```
boxplot(data_birth$bwt~data_birth$race, data=data_birth, id.method="y")
```



Aunque se observan valores atípicos y pequeñas diferencias entre el tamaño de las cajas, los tres grupos parecen seguir una distribución bastante simétrica. Seguimos adelante con el estudio y ahora comprobemos las condiciones para aplicar un ANOVA. Por una parte, disponemos de una muestra de estudio donde las observaciones de cada grupo son independientes entre sí.

A continuación, realizaremos un test que acabará de determinar la normalidad de la distribución. Normalmente, se aplica el test de Shapiro-Wilk, si la muestra es inferior a 50 observaciones, o el test de Kolmogorov-Smirnov, si dicha muestra es superior. En nuestro caso aplicaremos este último test.

```
##Test Kolmogorov-Smirnov
##Instalaremos el paquete nortest que contiene las funciones que aplicar
require(nortest)
## Loading required package: nortest
by(data=df_birth_anova,INDICES
=df_birth_anova$raza,FUN=function(x){lillie.test(x$peso)})

## df_birth_anova$raza: white
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  x$peso
## D = 0.090389, p-value = 0.05113
##
## ------------------------------------------------------------
## df_birth_anova$raza: black
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  x$peso
```

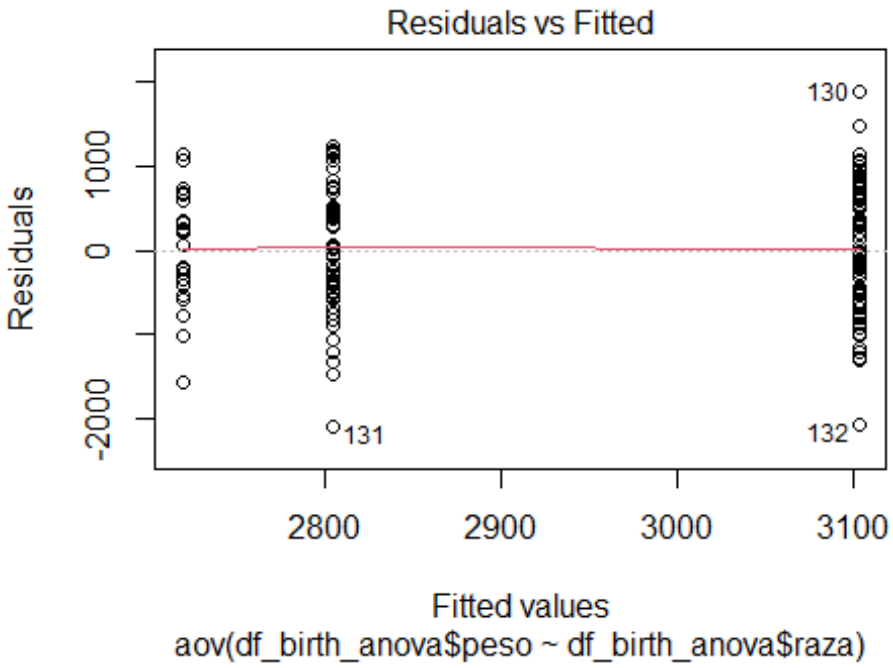
```
## D = 0.1231, p-value = 0.3946
##
## ------------------------------------------------------------
## df_birth_anova$raza: other
##
##  Lilliefors (Kolmogorov-Smirnov) normality test
##
## data:  x$peso
## D = 0.11598, p-value = 0.02578
```

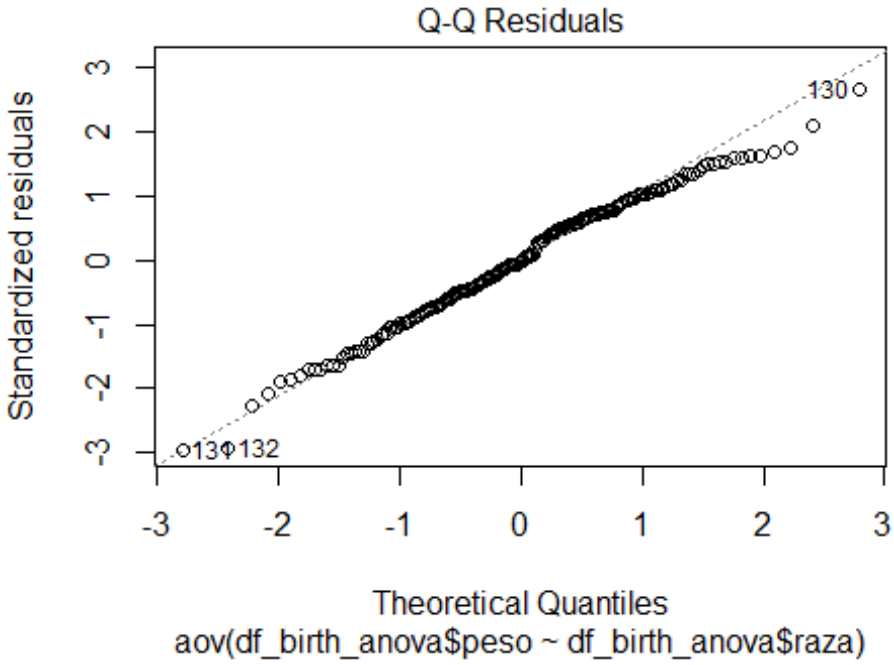
Los anteriores test de normalidad muestran falta de normalidad en algún grupo, por ejemplo, el grupo 3. Esto viene determinado por el valor del p-value. En general, se acepta la hipótesis de normalidad si este valor es superior a 0.05. Podemos comprobar, también, que el grupo 1 estaría al límite de aceptar dicha normalidad. A continuación, debemos evaluar la varianza constante entre los grupos, denominada homocedasticidad. Para ello, existen diferentes test que podemos aplicar en función de los resultados de normalidad; si estos no son claramente aceptables, desestimamos utilizar el test de Fisher y el test de Bartlett y aplicamos el test de Levene (aconsejable si tenemos variables categóricas) o el test de Fligner-Killeen.

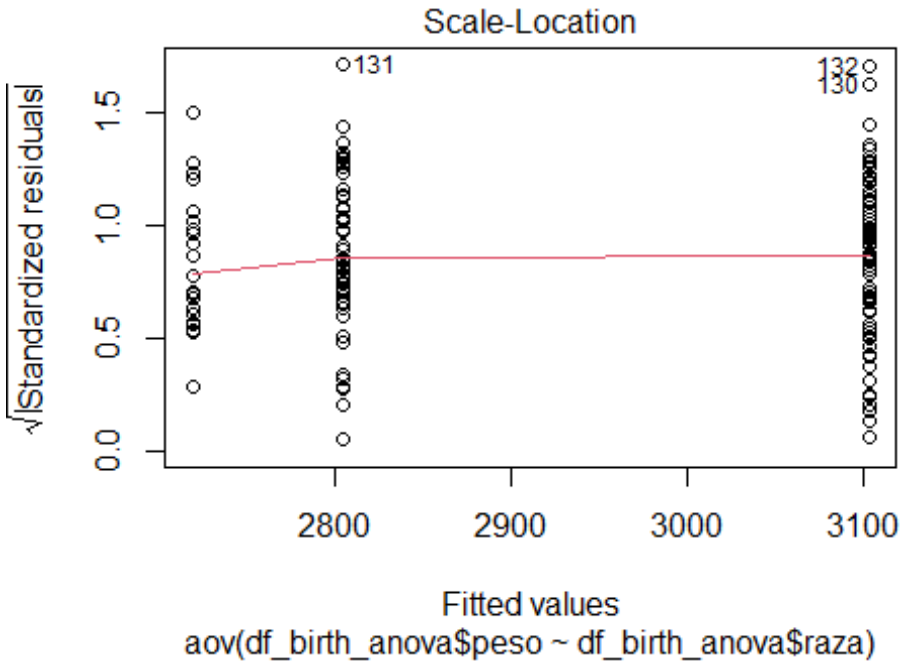
```
##Test Fligner-Killen diagnosis homocedasticidad
fligner.test(df_birth_anova$peso~df_birth_anova$raza,df_birth_anova)

##
## Fligner-Killeen test of homogeneity of variances
##
## data:  df_birth_anova$peso by df_birth_anova$raza
## Fligner-Killeen:med chi-squared = 1.0086, df = 2, p-value = 0.6039
```

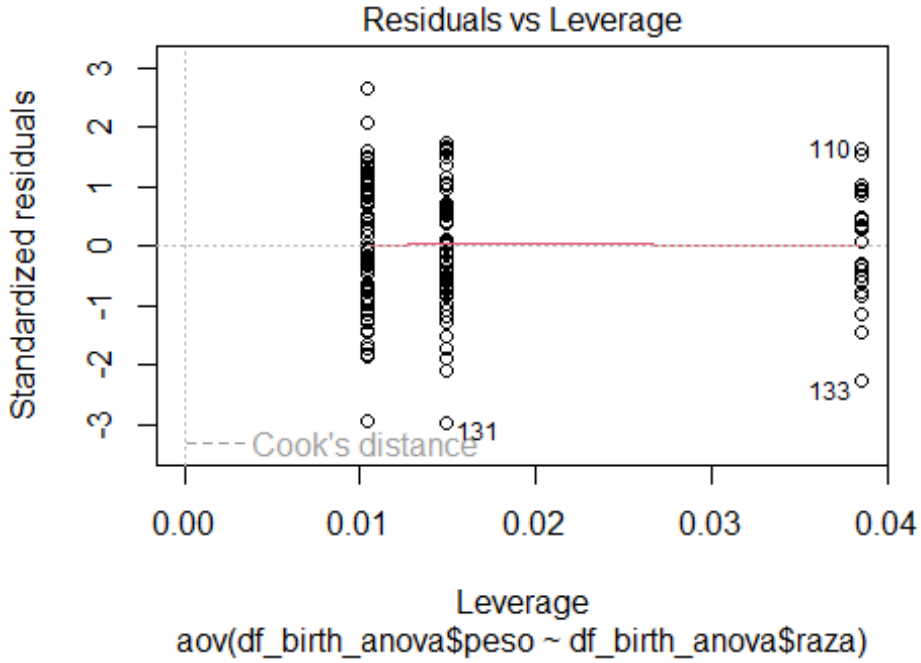
No se detecta falta de homocedasticidad. *A priori*, si no se cumplen las condiciones de ANOVA, no sería necesario continuar con el estudio. No obstante, excepto casos concretos, conviene seguir avanzando, ya que posteriores estudios nos pueden aportar más información.

```
##Estudio anova
anova_birthwt<-
aov(df_birth_anova$peso~df_birth_anova$raza,data=df_birth_anova)
summary(anova_birthwt)

##                       Df   Sum Sq Mean Sq F value  Pr(>F)
## df_birth_anova$raza    2  5015725 2507863   4.913 0.00834 **
## Residuals            186 94953931  510505
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

plot(anova_birthwt)
```


Residuals vs Fitted
Residuals
1000
0
-2000
130
131
132
2800
2900
3000
3100
Fitted values
aov(df_birth_anova$peso ~ df_birth_anova$raza)


Q-Q Residuals
Standardized residuals
3
2
1
0
-1
-2
-3
130
131
132
-3
-2
-1
0
1
2
3
Theoretical Quantiles
aov(df_birth_anova$peso ~ df_birth_anova$raza)





De los resultados anteriores se desprende que sí puede haber diferencias estadísticamente significativas entre las razas de las madres. En los casos en los que se detectan diferencias significativas entre las medias de los grupos, se utilizarían métodos de comparaciones múltiples y correcciones.

Las más recomendadas son Corrección de Holm y TukeyHSD.

```
pairwise.t.test(x=df_birth_anova$peso,g=df_birth_anova$raza,p.adjust.method
="holm",pool.sd=TRUE,paired=FALSE,alternative="two.sided")

##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  df_birth_anova$peso and df_birth_anova$raza
##
##       white black
## black 0.033 -
## other 0.029 0.605
##
## P value adjustment method: holm
```

Información adicional

Algunos de los recursos materiales utilizados y recomendados para trabajar este **LAB5** son los siguientes:

An introduction to Machine Learning with R

Machine Learning con R y Caret

Data Science, Analitys and Machine learning with R