Técnicas de Análisis de Datos Económicos con R

2024-12-05

**Autores:**

Miguel Ángel Tarancón Morán. Catedrático de Economía Aplicada. Universidad de Castilla - La Mancha.

# Prefacio

Este libro recoge las diversas prácticas que se han ido desarrollando a lo largo de multitud de cursos en varias asignaturas de grado y máster relacionadas con el análisis de datos, especialmente de tipo económico, en la Facultad de Derecho y Ciencias Sociales de Ciudad Real.

Gracias a tantas y tantas personas y compañeros/as que hacen posible la construcción y mejora del libro.

# 1 Introducción.

## 1.1 Llámalo Estadística.

Todo el mundo habla de las estadísticas. Constantemente se hace referencia a estas en los medios de comunicación, todo está medido y estructurado por estos entes que convierten la realidad en una amalgama de números. Y más aún en el campo del comportamiento humano, es decir, lo que conocemos como *Ciencias Sociales*. A diario nos llegan las estadísticas sobre la intención de voto cuando hay unas elecciones, del crecimiento de la economía en términos del *PIB*, del comportamiento de los precios medido mediante el concepto de *inflación*…

El secreto de la relevancia que les damos a las estadísticas subyace en que, de partida, suponen una forma sintética y objetiva de representar la realidad que nos rodea, de manera que podemos abarcar el conocimiento de tal realidad de un modo más o menos plausible. Y esta representación de la realidad es a priori *objetiva* porque las estadísticas se elaboran siguiendo unas metodologías que se apoyan en un lenguaje universal: las matemáticas.

Sí. El lenguaje matemático es un lenguaje que pueden entender todas las personas, tengan la procedencia que tengan, y sean de la condición que sean. Si necesitas comunicarte con *casi* cualquier persona del mundo, habla en inglés. Si necesitas comunicarte con cualquier persona del mundo, hazlo mediante las matemáticas, aunque sean matemáticas más o menos elementales.

Por ello, las estadísticas se expresan en lenguaje matemático.

Pero lo que comúnmente entendemos como *estadísticas* no son más que unos resultados, unos outputs de la Estadística. La Estadística en realidad es algo mucho más complejo. Es una Ciencia. Las estadísticas son construidas usando el método estadístico; pero la Estadística se utiliza para muchas más cosas que para publicar estadísticas.

### 1.1.1 Concepto de Estadística.

El término “Estadística” proviene de la palabra latina *status*, “el Estado”, y fue acuñado por Achenwall a mediados del siglo XVIII con el significado de “recogida, procesamiento y utilización de datos por parte del Estado”.

Sin embargo, tal y como se entiende hoy en día, es decir, en el sentido de ***Ciencia Estadística***, surgió como resultado de la integración de dos disciplinas: la *Aritmética Política*, en ese sentido de la cuantificación del Estado; y del *Cálculo de Probabilidades*, que nace en el siglo XVII como Teoría Matemática de los juegos de azar y que podríamos asociar al sentido de *Estadística Matemática*.

Ciñéndonos pues a este último sentido, a lo largo de la Historia se han dado múltiples definiciones de Estadística. Fisher propone una definición quizá demasiado generalista al decir que la Ciencia Estadística es esencialmente una rama de las matemáticas aplicada a los datos observados. Una reflexión que puede ayudar a delimitar la definición de la Ciencia Estadística es la que realiza ([Peña 1983](#ref-Peña1983)) cuando realiza la siguiente reflexión:

*“La Estadística como disciplina científica ocupa un lugar muy singular en el conjunto de las ciencias. La Física, la Medicina o la Sociología tienen un área sustantiva de conocimiento y cuando utilizan modelos matemáticos, los subordinan al objeto principal de hacer avanzar el conocimiento en su parcela de estudio de la realidad. El objetivo de la Matemática, en contraposición, es ampliar la concepción y generalidad de sus propias herramientas analíticas, con absoluta independencia de la posible relación entre los entes matemáticos abstractos y los fenómenos reales. La Estadística participa de esos dos objetivos, aunque con rasgos muy peculiares. Su campo de estudio son los fenómenos aleatorios que están presentes, en mayor o menor medida, en toda actividad humana de adquisición de conocimiento empírico.”*

En este mismo sentido, ([Martín-Pliego 2004](#ref-MartinPliego2004)) apunta:

*“La Estadística, por tanto, se configura como la tecnología del método científico que proporciona instrumentos para la toma de decisiones cuando éstas se adoptan en ambiente de incertidumbre, siempre que esa incertidumbre pueda ser medida en términos de probabilidad. Por ello, la Estadística se preocupa de los métodos de recogida y descripción de datos, así como de generar técnicas para el análisis de esta información.”*

En definitiva, la Estadística reúne tanto la concepción derivada de la *Aritmética Política*, entendida como recopilación sistemática de datos cara a la descripción de la realidad (“hacer” *estadísticas*); como la concepción *probabilística*, entendida como la modelización de dicha realidad cuando está inscrita en un ambiente de incertidumbre, con el objeto de acotar dicha incertidumbre y servir de ayuda en la toma decisiones (*representar matemáticamente el comportamiento de fenómenos sujetos a incertidumbre*, cuando contamos con **datos** que caracterizan a esos fenómenos).

### 1.1.2 El método estadístico.

En cuanto al método seguido por la Ciencia Estadística, prima el r**azonamiento *inductivo****:* las hipótesis que se plantean en la investigación implican propiedades observables en un conjunto de casos, cuyo análisis lleva a formular hipótesis más generales, aplicables ya a un conjunto mayor de casos. El método estadístico consiste, en definitiva, en sistematizar y organizar este procedimiento de aprendizaje que parte de lo particular para llegar a lo general.

En la aplicación del método estadístico podemos diferenciar una serie de **etapas básicas** que se exponen a continuación:

*a)* **Planteamiento del problema.** Consiste en definir el objeto de la investigación, (¿qué quiero obtener? ¿a dónde quiero llegar?), para lo cual debemos precisar la **población** de referencia y determinar las características que debemos observar y cómo serán recogidas. El resultado de esta fase es un sistema de **características** de interés observadas en un subconjunto de la población representativo de esta, al que llamamos **muestra**. Estas características se llamarán variables si están en escala métrica; o atributos, variables cualitativas o factores si están en escala no-métrica (nominal u ordinal). Las variables toman valores para cada elemento o caso de la muestra. Los atributos adoptan una categoría o nivel para cada uno de los casos que integran la muestra. Según los objetivos planteados en la investigación, el tamaño de la muestra, tipo de características, etc., se podrá hacer una primera selección de los posibles tipos de técnicas y modelos estadíticos a aplicar.

*b)* **Recogida y preparación de la información muestral**. Los datos, que son los valores (en caso de trabajar con variables) o categorías o niveles (en el caso de trabajar con atributos o factores) que adoptan los distintos casos que constituyen la muestra en relación con las características de interés de la población; han de ser obtenidos de las fuentes disponibles. Estas fuentes pueden ser primarias, cuando somos los propios investigadores los que generamos los datos (a través de la observación o la realización de encuestas), o secundarias, cuando estos datos ya han sido generados y/o recopilados por otros investigadores o instituciones. En cualquier caso, la muestra debe ser lo suficientemente amplia como para extraer conclusiones válidas para toda la población, y los datos deben ser de calidad, pues son la materia prima con la que trabajamos. Para ello, un requisito importante es que las fuentes de datos sean fiables.

*c)* **Depuración de los datos**. Antes de utilizar los datos muestrales conviene aplicar un análisis descriptivo que permitirá detectar posibles inconsistencias en los datos identificando los valores anómalos, posibles errores, etc. En esta fase es clave tanto identificar las carencias de datos existentes (datos faltantes o *missing data*), como identificar aquellos elementos de la muestra que no representan bien a la población, puesto que presentan comportamientos extraños en alguna o algunas de las variables o atributos en estudio (casos atípicos u *outliers*).

*d)* **Aplicación de técnicas o modelos estadísticos** para obtener resultados generalizables al conjunto de la población. Una vez se tienen claros los objetivos de la investigación y las características de la información muestral de la que se dispone (datos), y se han depurado convenientemente los datos, será el momento de plantear qué técnica o modelo estadístico aplicar. Aquí podemos distinguir, a su vez, distintas subetapas.

* Por un lado, la aplicación correcta de ciertas técnicas o modelos de naturaleza inferencial, requiere del **cumplimiento por parte de los datos de ciertos patrones de comportamiento** (por ejemplo, el cumplimiento por parte de las variables de un comportamiento acorde con una Ley Normal). Así, deberán aplicarse una serie de pruebas para comprobar hasta qué punto los datos de partida cumplen con estos patrones.
* Tras superar el punto anterior, podrá aplicarse la técnica o modelo a los datos para obtener los resultados que contribuyan a cubrir los objetivos de la investigación (usualmente, esta etapa se corresponde con la de **estimación** del modelo estadístico aplicado).
* Por último, los resultados deben ser sometidos a una subetapa de **validación y contraste**, en la que se valora hasta qué punto los resultados representan el comportamiento real de los casos estudiados (estudio de la bondad del modelo), y el grado de aptitud técnica del modelo, en el sentido de si el modelo estimado cumple con los requisitos que garantizan la calidad de los resultados (por ejemplo, si se cumplen ciertas hipótesis básicas que garanticen que los coeficientes estimados del modelo gozan de las mejores propiedades estadísticas, como insesgadez, eficiencia y consistencia).
* En esta etapa, además, se intentará simplificar el modelo, es decir, conseguir un modelo tan sencillo como sea posible, sin más parámetros de los necesariosy, que represente la realidad sin mucha pérdida de calidad con respecto a otro modelo más complejo, o sea, ciñéndose al ***principio de parsimonia*** de la modelización.

*e)* **Crítica y diagnosis del modelo**. Si una vez culminada la fase anterior se considera que el modelo es válido y técnicamente correcto, podrá ser adoptado para ayudar a la toma de decisiones, mediante análisis estructural, realización de previsiones o planteamiento de simulaciones. En caso contrario, si el modelo no se considera válido y/o correcto, deberemos reformular dicho modelo repitiendo las etapas anteriores hasta obtener un modelo que represente la realidad en estudio más adecuado.

En definitiva, el método estadístico sigue el método científico en cuanto a que tiene unas etapas bien delimitadas en las que se trata el **conocimiento *a priori*** (teoría) para obtener un **conocimiento *a posteriori*,** lo que pasa a engrosar el cuerpo de la Ciencia.

Es relevante destacar cómo, a su vez, el método científico, al ser aplicado al resto de ciencias, y a la propia Ciencia Estadística, recurre al método estadístico en su ejecución. Así, por ejemplo, en la etapa de recogida de evidencias observables (datos), a fin de verificar las consecuencias o hipótesis que se desprenden de una teoría previa, la Estadística interviene tanto a partir de la *Teoría de Muestras* como del *Diseño de Experimentos* para garantizar la validez y coherencia de los datos. En una fase posterior del método científico, se pasaría a verificar la nueva teoría que se desprende de las hipótesis articuladas a partir de la teoría preexistente. Nuevamente aquí interviene la Estadística como herramienta auxiliar, mediante la *modelización inferencial*. Además, en todo el proceso, que abarca tanto la observación de la realidad como a la generalización de los resultados como modo de confirmar una nueva teoría, aparece la incertidumbre en mediciones y resultados, por lo que el papel de la Estadística como procedimiento para la medición de dicha incertidumbre es indispensable.

De lo dicho se desprende una característica que hace de la Estadística una ciencia singular: su carácter de ***ciencia instrumental*** que auxilia al resto de ciencias en el desarrollo de sus cuerpos de conocimiento. De ahí que la Estadística es aplicada en la totalidad de las ciencias, bien sean naturales, jurídicas o sociales, y en todos los campos del saber, desde las áreas más técnicas hasta en las propias humanidades. Es decir, la Estadística es una herramienta fundamental en todo el proceso de adquisición de conocimientos a través de datos empíricos y, desde este punto de vista, podemos referirnos a la afirmación de ([Mood 1963](#ref-Mood&Graybill1963)):

*“La Estadística es la tecnología del método científico”.*

Esta extensión de la Ciencia Estadística como ciencia auxiliar de otras ciencias, junto con su crecimiento y madurez metodológica, ha permitido el nacimiento de áreas con un cuerpo de conocimiento específico que pueden ser consideradas, a su vez, como entidades con la categoría de ciencia, como pueden ser la Psicometría, la Estadística Económica y la Econometría[[1]](#Xc217c44f9d6f9d647df03178eb8b64fb8f54252). Así, a continuación, nos centraremos en la Estadística Económica, rama que ha ocupado un papel primordial en el desarrollo de la propia Ciencia Estadística desde el principio de sus orígenes.

[[1]](#Xf4c8113f07524640efcbf4d5822e1fc633a9e60) En nuestra opinión, no existe una delimitación clara entre *Estadística Económica* y *Econometría*, siendo la diferencia en todo caso un matiz dependiente de las técnicas y el enfoque empleado al enfrentarse a un determinado estudio. Quizá ambas disciplinas pudieran englobarse en otra disciplina más general que podría ser llamada ‘Economía Cuantitativa’. Véase en relación con este respecto ([Hernández-Alonso 2000](#ref-Hernandez2000)).

### 1.1.3 Economía y Estadística.

La aplicación del método estadístico a la Economía puede entenderse como el proceso de representación de los sistemas económicos, constituidos por los distintos agentes que operan en las economías, y las relaciones que los ligan. La Economía suele especificar dichas relaciones dándoles forma de teorías económicas. No obstante, las teorías económicas con frecuencia son demasiado imprecisas a la hora de plantear modelos económicos verificables. Como Paul Samuelson apunta (([Samuelson 2006](#ref-Samuelson&Nordhaus2006))):

“*Solo en una muy pequeña parte de las obras de Economía teóricas o aplicadas se ha tratado la derivación de los teoremas significativos operacionalmente. En parte, por lo menos, tal situación se debe a los malos preconceptos metodológicos, según los cuales, las leyes económicas deducidas de los supuestos a priori poseen rigor y validez, independientemente de cualquier conducta humana real… De hecho, las obras de economía rebosan de malas generalizaciones.”*

La aplicación de los instrumentos estadísticos, y en concreto del Método Estadístico, permite dotar a la Teoría Económica del grado de concreción necesario para verificar en los sistemas reales el cumplimiento y la validez de dichas teorías. Este proceso de representación de sistemas reales puede llegar a tal grado de especificación que se puedan cuantificar las consecuencias en los cambios provocados en los elementos y relaciones del sistema (([Intriligator 1996](#ref-Intriligatoretal1996)), capítulo II). Sin embargo, por muy alto que sea el nivel de especificación del modelo que representa la realidad económica, este deberá llevar implícito cierta carga de abstracción de la realidad a la que representa, para poder ser abarcable. La realidad económica, el sistema económico, supera necesariamente en complejidad a cualquier modelo propuesto por la Teoría Económica, ya que el sistema económico depende, en última instancia, de fenómenos inmersos en cierto grado de incertidumbre; lo que es atribuible, a su vez, a su vinculación con el comportamiento humano. De este hecho se deduce la necesidad de incluir en la modelización de la realidad económica elementos estocásticos, lo que origina una visión no determinista, sino probabilista de la realidad económica.

Como señala ([Martín-Pliego 2004](#ref-MartinPliego2004)), parte del conjunto de técnicas estadísticas aplicadas a la investigación económica es común a otras ciencias, mientras que otra parte es específica de este tipo de investigación, fruto de una evolución de la aplicación de la disciplina en el tratamiento de los temas económicos. Entre estas metodologías específicas se encuentran el estudio de las series temporales económicas, de la distribución de la renta, la construcción y análisis de números índices, la modelización regional, el análisis input-output e intersectorial, las técnicas demográficas e incluso, en nuestra opinión, la propia Econometría.

## 1.2 ¿Qué es R y cómo nos ayuda a analizar datos desde el punto de vista estadístico?

En los apartados anteriores hemos partido del concepto de Estadística como ciencia instrumental hasta llegar a la Estadística Económica, como aquel cuerpo de la Ciencia Económica que se sirve de las herramientas que ofrece la Estadística para profundizar en el conocimiento de la realidad económica.

Pero claro, lo interesante de esto es llevarlo a la práctica. Se necesita un *soporte de hardware y software* para poder aplicar las técnicas estadísticas a los datos económicos, con el objetivo de crear conocimiento a partir de dichos datos. Este conocimeinto se traducirá en una reducción de la incertidumbre que inevitablemente viene aparejada a los fenómenos económicos, lo que redundará en una mejor toma de decisiones.

En los últimos tiempos se ha producido una evolución de hardware sin precedentes, lo que ha dado soporte al desarrollo de un potente software dedicado al análisis de datos (todo tipo de datos, no solamente económicos). Este software permite a cualquier investigador aplicar las últimas técnicas de análisis estadístico a cualquier masa de datos, lo que ha supuesto una verdadera revolución. A su vez, esta realidad se ha retroalimentado, de modo que se ha producido un constante avance en el desarrollo de técnicas y tecnologías de análisis de datos cada vez más complejas. Así, podemos hablar de técnicas de aprendizaje automático o *machine learning* (supervisado, no-supervisado o reforzado) o, más recientemente, de modelos de análisis basados en la *inteligencia artificial*.

En este caldo de cultivo, en el que se dispone de grandes masas de datos, de hardware capaz de procesarlas, y de técnicas capaces de extraer información de las mismas, se ha desarrollado un software cada vez más potente que une todos estos elementos para modelizar la realidad. Este software se concreta en aplicaciones y plataformas diversas: SPSS, Stata, SAS… Y también lenguajes de programación orientados al análisis estadístico y matemático, como pueden ser Python, Matlab, Julia o… R.

Sí. R no es solo una aplicación al uso. Es todo un lenguaje de programación, orientado principalmente a la analítica de datos, sobre todo desde una perspectiva estadística. R es un proyecto de GNU, por lo que los usuarios son libres de modificarlo y extenderlo. R se distribuye como software libre bajo la licencia GNU y es multiplataforma, lo que ha facilitado su difusión y la existencia de una comunidad muy activa de ususarios y desarrolladores.

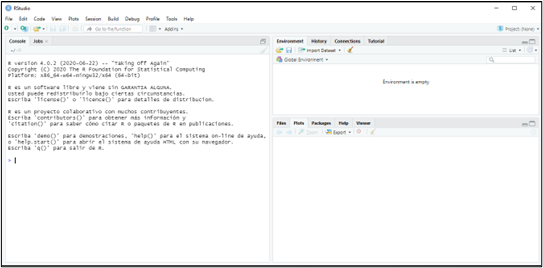
## 1.3 Instalación de R y R-Studio.

Como ya se ha mencionado, R es un software o lenguaje de uso y difusión gratuitos, bajo licencia GNU. El modo de instalar R es sencillo: basta con ir a la web [*CRAN*](https://cran.r-project.org/) (*Comprehensive R Archive Network*) y descargar la última versión disponible en el sistema operativo del que se sea usuario (en este manual, Microsoft® Windows®). Se ejecutará el archivo descargado, y se completará la instalación.

Una limitación de R es la interfaz o IDE (entorno de desarrollo integrado) que incorpora. Es decir, el “software” con el que se interactúa con el lenguaje R. Esta IDE es muy poco amigable. Para superar esta limitación, existen IDEs alternativas, entre las que destaca RStudio, desarrollada por Posit® Software. Esta IDE es gratuita. De nuevo, simplemente tendremos que ir a la web de[RStudio](https://posit.co/download/rstudio-desktop/) y descargar e instalar la versión gratuita.

## 1.4 R y RStudio. Comienzo: Proyectos.

Tras instalar R y su IDE RStudio, podremos comenzar a trabajar. Para ello, abriremos RStudio pulsando en el icono correspondiente. Aparecerá la siguiente ventana:



IDE de RStudio.

La parte izquierda de la ventana es la **consola**. La consola es la sección de RStudio donde podemos manejar R mediante la introducción de código. Por ejemplo, podemos escribir 2+2 después del cursor (signo “>”), y pulsar Enter. La propia consola nos devolverá el valor 4:

2+2

## [1] 4

De todos modos, la forma más eficiente de trabajar es mediante “proyectos” y “scripts”.

Un **proyecto** básicamente viene asociado a la carpeta donde R trabajará, buscando los datos que sean sus “inputs”, y, en su caso, enviando sus resultados u “outputs”. Dicho de otro modo, es una carpeta más de nuestro sistema de carpetas o directorios; pero a la que dotamos de una característica especial: ser un proyecto de R. Si abrimos desde RStudio el proyecto, estaremos diciendo a R que, por defecto, preferentemente busque todos los archivos e inputs (datos, etc.) que necesite en esa carpeta de proyecto; y que, en su caso, guarde en tal carpeta los outputs que genere.

Para crear un nuevo proyecto, seguiremos la instrucción File → New Project, luego se nos preguntará si se crea el proyecto en una nueva carpeta o en una ya existente. Vamos a crearlo, por ejemplo, en el disco extraíble D, carpeta R, subcarpeta “explora”, que ya está creada. Nos saldrá una ventana para buscar la carpeta y, cuando la encontremos, pulsaremos Open y Create Project. Ya tendremos creado nuestro proyecto. Si nos vamos al explorador de Windows®, y buscamos la carpeta “explora”, encontraremos que en tal carpeta aparece un archivo de nombre “explora”, con un icono de un cubo con una “R”. Ese archivo lo que está haciendo es actuar como un “faro” que le dice a R que, cuando trabajemos en el proyecto “explora”, todos los archivos de datos necesarios estarán en esa carpeta (también llamada “explora”, porque el proyecto adopta el nombre de la carpeta donde lo localizamos). Y que, si nuestro trabajo aporta algún fichero de “output”, también se depositará en esa carpeta del proyecto.

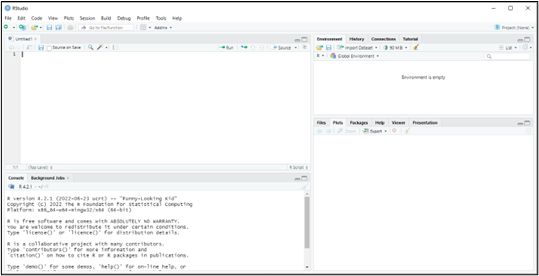
En futuras sesiones, si queremos trabajar en el mismo proyecto, en lugar de seguir la ruta File → New Project, tendremos que hacer File → Open Project.

## 1.5 Scripts.

En cuanto a los **scripts**, son programas o rutinas donde varias instrucciones se ejecutan secuencialmente. Para crear un script, se seguirá la ruta File → New File → R Script. Y si el script lo guardamos, ¿dónde lo hará? Pues en la carpeta “explora”, que es la del proyecto en el que estamos trabajando.

Informáticamente, un script es simplemente un archivo de texto plano. Se puede modificar con cualquier editor de texto. Afortunadamente, para no estar entrando y saliendo de R-Studio, esta interfaz incorpora un **editor** de scripts, lo cual es muy cómodo.

Vemos cómo ahora, a la izquierda de RStudio, ha aparecido, en la parte superior, una nueva ventana, pasando la consola a ocupar la parte inferior. Es la ventana del “editor”:



El editor de Scripts de RStudio.

Igual que con los proyectos, podemos crear desde RStudio un script nuevo, o abrir uno preexistente; y modificarlo, ejecutarlo, o volverlo a guardar.

Vamos a comenzar a escribir nuestro script. Si queremos hacer un comentario que no ejecute ninguna instrucción, éste irá precedido del símbolo almohadilla o *hashtag* “#”. Luego, vamos a ordenar a R que haga la operación de suma: 2+2. Escribimos, por tanto, en el editor:

#Ejemplo de Script  
2+2 #este script hace una simple suma.

Si pulsamos Control + Mayúsculas + ENTER o al desplegable de Source → Source with Echo, se ejecutará el script (para ejecutar solo la línea donde está el cursor, pulsaremos Control + ENTER o el botón de Run; y para ejecutar varias líneas, hemos de sombrearlas y pulsar Control + ENTER o el botón de Run). En la consola aparecerá:

## [1] 4

Podemos guardar el script con File → Save As… ¿Dónde se guardará por defecto? Pues en la carpeta “explora”, que es la de nuestro proyecto. Una vez nuestro script ya tiene nombre, podemos ir guardándolo de vez en cuando pulsando simplemente en el botón del “disquete” del editor. Vamos a llamarlo, por ejemplo, “explorando”. Si vamos, en el explorador de Windows®, a nuestra carpeta de proyecto, veremos que hay un archivo de texto llamado “explorando” con extensión “.R” (explorando.R). Este script lo podremos ejecutar cuantas veces queramos sin tener que escribir nada, o reescribirlo si vemos que no funciona o que necesitamos hacer modificaciones. Esa es la ventaja de trabajar con scripts.

Para recuperar un script en una nueva sesión de trabajo simplemente tenemos que seguir las instrucciones File → Open File… y seleccionarlo.

## 1.6 Funciones.

R trabaja con datos y funciones, principalmente. Pero, ¿qué es una **función**?

Una función es un conjunto o **sistema de instrucciones** que convierten unos datos de entrada o **inputs** en otros datos de salida, resultados, u **outputs**. Una función puede ser muy sencilla o ser verdaderamente compleja. Por otro lado, no todas las funciones están integradas en “paquetes”; sino que el usuario puede crear sus propias funciones (por ejemplo, escribiéndolas en un script) y ejecutarlas.

Las partes básicas de una función son:

* **Entradas, inputs o argumentos:** son las diversas informaciones necesarias para realizar el procedimiento de la función. Los argumentos pueden ser introducidos por el usuario, o pueden venir dados por defecto, lo que quiere decir que, si el usuario no dota de valor a un argumento, este tomará automáticamente un valor prestablecido.
* **Cuerpo:** está formado por un conjunto de instrucciones que transforman los *inputs* o entradas en los *outputs* o salidas. Si el cuerpo de la función está formado por varias instrucciones, éstas deben escribirse entre llaves { }.
* **Salidas:** son los resultados u *output* de la función. Si una función ofrece como salida varios tipos de *objetos,* estos objetos suelen ser almacenados en una estructura de almacenaje de *lista*.

Como ejemplo, vamos a integrar en nuestro script una función, llamada “suma”. Esta función requerirá de dos entradas o argumentos (dos números cualesquiera), y ofrecerá, como resultado, salida u output; la suma de tales entradas. El código es:

suma <- function(x, y) {  
 resultado <- x + y  
 return(resultado)  
}

Ahora, una vez ejecutado el código anterior; si queremos sumar, por ejemplo, los números 12 y 16, solo tendremos que teclear en la consola, o escribir en el script y hacer run, a la línea:

suma(x=12, y=16)

## [1] 28

## 1.7 Paquetes (packages).

R es un lenguaje de programación en torno al cual se ha desarrollado una cantidad casi inimaginable de recursos: funciones, bases de datos, utilidades… Tal es la cantidad de recursos, que no sería operativo abrir R (directamente, o a través de una IDE, como RStudio) y tener inmediatamente todos esos recursos activos y preparados para ser utilizados. Además, R debería ser actualizado de un modo casi constante.

Por todo ello, todos los recursos disponibles están organizados mediante “paquetes” (“packages” en inglés). Un **paquete** es una colección de funciones y/o un conjunto de datos desarrollados por la comunidad de R. Estos incrementan el potencial de R ampliando sus capacidades básicas, o añadiendo otras nuevas.

De hecho, cuando abrimos R, algunos de estos paquetes, que se han instalado junto al propio lenguaje, se activan. Pero solo algunos. Un ejemplo es el paquete {base} o el paquete {stats} ([R Core Team 2024](#ref-R-base)).

La mayor parte de los paquetes disponibles no forman parte, por “defecto”, en la misma instalación de R. Se encuentran en diversos servidores llamados repositorios. El más importante, es [CRAN](https://cran.r-project.org/), que es el “repositorio oficial” y que alberga más de 10.000 paquetes. Pero existen otros repositorios, a destacar, por ejemplo, GitHub.

Para **instalar** un paquete en nuestra máquina que esté albergado en CRAN, un modo sencillo es, dentro de R-Studio, pulsar en la ventana inferior / izquierda sobre la pestaña “Packages”, y sobre el botón “Install”. Emergerá entonces una ventana donde hay un campo para escribir el nombre del paquete (al comenzar a escribirlo, el propio R-Studio te sugerirá los paquetes disponibles). Esto equivale a usar (bien directamente en la consola, o bien como línea de código insertada en un script) la instrucción install.packages(), con el nombre del paquete entre comillas (si son varios, pues irán separados por comas.

Una vez se tiene instalado el paquete, ya no habrá que volver a instalarlo para utilizarlo; sino **activarlo**. De hecho, todos los paquetes que no se encuentran por defecto en la propia instalación de R, deben ser activados para poder usar sus funcionalidades y/o datos. Para hacerlo, se debe utilizar la instrucción library(), y el nombre del paquete dentro del paréntesis.

Del nombre de esta instrucción surge la confusión común de tomar como sinónimos las palabras “paquete” y “librería” en el entorno de R. Si nos referimos a estas colecciones de funcionalidades y/o datos; lo correcto es “paquete”, ya que “librería” tiene más que ver con la organización informática de un software.

## 1.8 Help! (sistema de ayuda).

A veces podemos albergar dudas sobre la correcta utilización de las funcionalidades y herramientas que nos proporciona un paquete. Hay varias fuentes de ayuda para intentar encontrar respuesta a las cuestiones que se nos plantean.

Una opción, para obtener información general sobre un paquete, es utilizar la función help(), con el argumento “package”. Por ejemplo:

help(package="base")

Observaremos como en la ventana inferior / izquierda de R-Studio nos saldrá la información correspondiente. De hecho, en tal ventana existe una pestaña **“Help”** para obtener la ayuda sin teclear código.

Además, cada función puede ser consultada individualmente mediante help("nombre de la función") o help(function, package = "package") si el paquete no ha sido cargado. Estas instrucciones nos mostrarán la descripción de la función y sus argumentos acompañados de ejemplos de utilización. Por ejemplo:

help("rm", package="base")

La instrucción anterior nos aporta la documentación sobre la función rm() del paquete {base} de R (nota: este paquete se activa por defecto al abrir R o R-Studio; por lo que el segundo argumento, con el nombre del paquete que contiene la instrucción no es necesario).

Otra opción para mostrar información de ayuda es la exploración de las “viñetas” (vignettes). Las **viñetas** son documentos que muestran de un modo más detallado las funcionalidades de un paquete. La información de las viñetas de un paquete están disponibles en el archivo “documentation”. Puede obtenerse una lista de las viñetas de nuestros paquetes instalados con la función browseVignettes(). Si solo queremos consultar las viñetas de un paquete concreto pasaremos como argumento a la función el nombre del mismo: browseVignettes(package = "packagename"). En ambos casos, una ventana del navegador se abrirá para que podamos fácilmente explorar el documento.

Si optamos por permanecer en la consola, la instrucción vignette() nos mostrará una lista de viñetas, vignette(package = "packagename") las viñetas incluidas en el paquete, y una vez identificada la viñeta de interés podremos consultarla mediante vignette("vignettename").

# 2 Almacenando y manipulando datos.

## 2.1 Objetos. Datos.

Como vimos en el capítulo 1, tras ejecutar un sencillo script (o al escribir instrucciones directamente desde la consola), R es interactivo: responde a las entradas que recibe. Las entradas o **expresiones** pueden ser, básicamente:

* Expresiones aritméticas.
* Expresiones lógicas.
* Llamadas a funciones.
* Asignaciones.

Las expresiones realizan acciones sobre **objetos** de R. Los objetos en R son entes que tienen ciertas características, *metadatos*, llamados atributos. No todos los objetos tienen los mismos atributos y, ni tan siquiera, todos los objetos tienen atributos que los caractericen.

Los objetos más importantes en R son ciertas estructuras o *contenedores* diseñados para almacenar elementos:

* Vectores.
* Matrices.
* Listas.
* Data frames.
* Factores.

Los elementos almacenados en los objetos se dividen en *clases*. Entre las diferentes clases, destacan las clases referidas a **datos**, que pueden ser de diferentes *modos*: logical (verdadero/falso), numeric (números) o character (cadena de texto). El modo numeric puede ser, a la vez, de tipo integer (número entero) o double (número real). En el caso de logical y carácter, modo y tipo coinciden.

Vamos a profundizar un poco en algunas de estos contenedores de datos. Vamos a suponer que trabajamos en el proyecto que creamos en el capítulo anterior (proyecto “explora”), y que vamos a editar el script que también creamos en tal capítulo (script “explorando.R”, que se encontrará ubicado en la carpeta del proyecto “explora”).

### 2.1.1 Vectores.

Los **vectores**, son conjuntos de elementos **de la misma clase**. Vamos a definir por ejemplo el vector x = (1,3,5,8). Para ello, vamos a escribir en nuestro script:

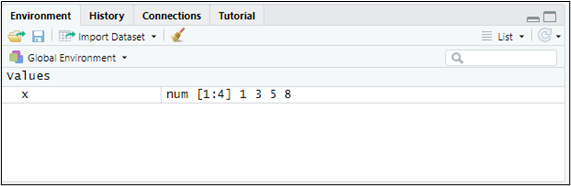
x <- c(1,3,5,8)

Ejecutamos la línea (situando el cursor en algún lugar de ella, dentro del script; y pulsando a la vez las teclas Control + Enter o pinchando con el ratón en el botón Run del editor). Ya tenemos nuestro primer objeto de tipo *vector* en memoria. Por cierto, lo que hemos hecho es una **asignación**, que se escribe con una flecha creada mediante los signos “<” y “-”. Hemos asignado a un vector llamado “x” los elementos 1, 3, 5 y 8.

Para ver el vector simplemente escribimos en la consola (o en el script) el nombre del vector, “x”. El resultado será:

## [1] 1 3 5 8

Además, si miramos en la ventana superior-derecha de R-Studio, veremos que en el ***Global Environment*** se muestra nuestro vector y que, además, se nos informa de que tiene *modo* *numérico*. El *Global Environment* nos informa de los objetos que R tiene en memoria:



Nuestro vector en memoria.

Si queremos obtener un vector de números consecutivos del 2 al 6, basta con ejecutar en la “consola” (o escribir y ejecutar en el script):

y <- c(2:6)

Al escribir el nombre del vector “y” en la “consola” obtendremos:

y

## [1] 2 3 4 5 6

Si queremos saber la longitud de un vector, usaremos la *función* length(). Por ejemplo, length(y) nos devolverá el valor 5. Escribamos en el script y ejecutemos:

length(y)

## [1] 5

Un vector puede incluir, además de números, caracteres o grupos de caracteres alfanuméricos; siempre entrecomillados (lo fundamental es que sean elementos de la misma clase). Por ejemplo, el vector “genero” (¡no pongamos tildes o podemos tener problemas!). Así, si ejecutamos estas dos líneas de código:

genero<-c("Mujer","Hombre")  
genero

Se habrá creado el vector “genero”:

## [1] "Mujer" "Hombre"

Podemos obtener la clase de los elementos almacenados en nuestro vector con la función class():

class(genero)

## [1] "character"

Si falta un dato en un vector, habrá que escribir “NA” (not available). Por ejemplo, si falta el tercer dato de este vector “z”, este vector se escribirá como:

Z <- c(1,2,NA,2,8)

Para **seleccionar un elemento** concreto de un vector, indicaremos entre corchetes la posición en la que se encuentra. Por ejemplo, refiriéndonos al vector “x”, para obtener el valor de su tercer elemento, haremos:

x[3]

## [1] 5

Si queremos que se nos muestre los elementos del vector x del 2º al 4º:

x[2:4]

## [1] 3 5 8

Por último, si queremos sacar en pantalla los elementos 1º y 4º, tendremos que incluir una “c” seguida de un paréntesis que recoja el orden de los elementos que queremos seleccionar:

x[c(1,4)]

## [1] 1 8

### 2.1.2 Matrices.

Las **matrices**, internamente en R, son vectores; pero con dos atributos adicionales: número de filas y número de columnas. Se definen mediante la función matrix(). Por ejemplo, para definir la matriz “a”:

Tendremos que escribir:

a <- matrix(c(1,2,3,4,5,6,7,8,9),nrow=3)  
a

## [,1] [,2] [,3]  
## [1,] 1 4 7  
## [2,] 2 5 8  
## [3,] 3 6 9

El número de filas de la matriz (y por tanto, el número de columnas) se fija con el argumento nrow = . También podríamos fijar el número de columnas, con ncol = .

Como vemos, por defecto, R va “cortando” el vector por columnas (si lo preferimos, lo puede hacer también por filas, añadiendo a la función matrix() el argumento by row = true; pero, en nuestro ejemplo, obtendríamos la matriz traspuesta a la que queremos almacenar).

Las dimensiones (número de filas y de columnas) de la matriz pueden obtenerse mediante la función dim():

dim(a)

## [1] 3 3

3 filas y 3 columnas.

Si queremos seleccionar elementos concretos de una matriz, lo haremos utilizando corchetes para indicar filas y columnas. Hemos de tener en cuenta que, trabajando con matrices, siempre tenemos

Si se deja en blanco el espacio entre el corchete inicial y la coma, esto querrá decir que consideramos todas las filas. Y si no insertamos nada entre la coma y el corchete de cierre, esto significará que consideramos todas las columnas. A continuación tenemos varios ejemplos de código, con el resultado obtenido en la consola:

a[2,3]

## [1] 8

a[1:2,2:3]

## [,1] [,2]  
## [1,] 4 7  
## [2,] 5 8

a[,c(1,3)]

## [,1] [,2]  
## [1,] 1 7  
## [2,] 2 8  
## [3,] 3 9

a[c(1,3),]

## [,1] [,2] [,3]  
## [1,] 1 4 7  
## [2,] 3 6 9

Tanto para vectores como para matrices, funcionan las operaciones suma y diferencia sin más complicaciones. En el caso del producto, sin embargo, hay que tener en cuenta que, por ejemplo, a\*a devuelve la multiplicación elemento a elemento, es decir:

a\*a

Devuelve la multiplicación elemento a elemento (en este caso, el cuadrado de cada número, al multiplicar la matriz a por sí misma):

## [,1] [,2] [,3]  
## [1,] 1 16 49  
## [2,] 4 25 64  
## [3,] 9 36 81

Para hacer el verdadero producto matricial deberá introducirse:

a%\*%a

## [,1] [,2] [,3]  
## [1,] 30 66 102  
## [2,] 36 81 126  
## [3,] 42 96 150

### 2.1.3 Data frames.

Un ***data frame*** es un objeto que almacena datos organizados mediante la clase data.frame. Esta organización consiste en que, por filas, se disponen los diferentes casos o sujetos; mientras que por columnas se posicionan las variables. Así:

* Es similar a una matriz en el sentido de que tiene dos dimensiones. Podemos acceder a sus elementos con corchetes, tenemos nombres de filas y columnas, y podemos operar con ellas.
* Cada columna tiene un nombre, de manera que podemos acceder a una columna concreta con el símbolo **$**. Todas las columnas (variables) son vectores con la misma longitud.
* Cada columna puede ser un vector numérico, factor, de tipo carácter o lógico.

Por ejemplo, vamos a crear el *data frame* **“datos”**, con tres variables: “peso”, “altura”, y “color de ojos”, llamadas “Peso”, “Altura” y “Cl.ojos”, respectivamente; para 3 individuos o casos. Una opción es crear primero las tres variables como vectores, y luego crear el *data frame* mediante la función dataframe():

Peso<-c(68,75,88)  
Altura<-c(1.6,1.8,1.9)  
Cl.ojos<-c("azules","marrones","marrones")  
datos<-data.frame(Peso,Altura,Cl.ojos)

Si ahora ejecutamos una línea con el nombre de nuestro *data frame*, lo obtendremos como resultado en la consola:

datos

## Peso Altura Cl.ojos  
## 1 68 1.6 azules  
## 2 75 1.8 marrones  
## 3 88 1.9 marrones

Para obtener los nombres de las variables (es decir, el nombre de cada columna) teclearemos la función:

names(datos)

Obteniéndose:

## [1] "Peso" "Altura" "Cl.ojos"

Para obtener solo los datos de la columna (variable) color de ojos teclearemos datos$Cl.ojos:

datos$Cl.ojos

## [1] "azules" "marrones" "marrones"

Y para obtener los datos de peso: datos$Peso:

datos$Peso

## [1] 68 75 88

Para saber el número de filas y de columnas de una hoja de datos utilizaremos las funciones nrow() y ncol():

nrow(datos)

## [1] 3

ncol(datos)

## [1] 3

Para seleccionar elementos de un data frame, se pueden seguir las mismas reglas que para la selección de elementos de una matriz (con el número de cada fila, que es cada individuo; y el número de cada columna, que es cada variable. Para elegir una variable, no obstante, ya hemos visto que es posible usar su nombre; aunque precedido del nombre del data frame y el signo $. Por ejemplo, si ejecutamos:

datos[,2]

## [1] 1.6 1.8 1.9

datos$Altura

## [1] 1.6 1.8 1.9

Obtenemos el mismo resultado.

## 2.2 Importando datos.

Lo más frecuente es que no tecleemos los datos, como hemos hecho hasta ahora; sino que los importemos a R desde algún contenedor externo (archivo de texto, hoja de cálculo, base de datos…). Nosotros vamos a importar nuestros datos desde Microsoft® Excel®. Vamos a cerrar el script que hemos estado construyendo en los apartados anteriores (para conservarlo hay que guardarlo antes), aunque vamos a seguir trabajando en el mismo proyecto (que habíamos llamado “explora”). Iremos a la carpeta del proyecto y guardaremos en ella los dos archivos de esta práctica (obtén el enlace a los archivos en la sección final del capítulo):

* Un archivo de Microsoft® Excel® llamado “eolica\_20.xlsx”
* Un script con las instrucciones que vamos a mostrar a continuación, y que se llama “explora\_eolica.R”

Si abrimos el archivo de Microsoft® Excel® comprobaremos que se compone de tres hojas. La primera muestra el criterio de búsqueda de casos en la un aviso sobre el uso exclusivo que se debe dar a los datos incorporados; la segunda recoge la descripción de las variables consideradas; y la tercera (hoja “Top 20”) guarda los datos que debemos **importar** desde R-Studio. Estos datos se corresponden con diferentes variables económico-financieras de 20 empresas productoras de electricidad mediante generación eólica. Luego vamos a cerrar el archivo de Microsoft® Excel® y volveremos a R-Studio. Vamos a abrir nuestro script “explora\_eolica.R” con File → Open File… Este script contiene el programa que vamos a ir ejecutando en la práctica. La primera línea / instrucción en los scripts suele ser:

rm(list = ls())

La instrucción tiene como objeto limpiar el *Environment* (memoria) de objetos de anteriores sesiones de trabajo. Para importar los datos localizados en el archivo de Excel “eolica\_20.xlsx” el código es:

# DATOS  
library(readxl)  
eolica\_20 <- read\_excel("eolica\_20.xlsx", sheet = "Top 20")

¡Atención! Si nunca se ha utilizado el paquete {readxl} (que contiene el código necesario para importar datos de un archivo de Microsoft® Excel®), cuando la intentemos activar con la función library() nos dará un error o nos dirá que previamente hay que importarla. En ese caso, iremos a la ventana inferior-derecha y pulsaremos la pestaña Packages, pulsaremos en Install, y emergerá una ventana donde dejaremos el “repositorio” que viene por defecto y, en el campo “Packages”, escribiremos el nombre del “paquete” que contiene la librería que nos hace falta (normalmente coincide con el nombre de la propia librería, en nuestro caso {readxl}. Una vez descargado el “paquete”, podremos ejecutar el código anterior sin problemas.

Otra cuestión importante a tener en cuenta es que, en la hoja de cálculo del ejemplo, los “valores perdidos” o *missing values* (celdas en las que no hay datos), venían en blanco. Pero, en ocasiones, pueden contener algún tipo de anotación, como por ejemplo, “n.d.” (no disponible). En tal caso, deberá incluirse un argumento más que informe de estas celdas que, sin estar en blanco, no tienen dato:

eolica\_20 <- read\_excel("eolica\_20.xlsx", sheet = "Top 20", na = c("n.d."))

Volviendo a nuestro ejemplo, podemos observar cómo en el *Environment* ya aparece un objeto. Este objeto es una estructura de datos tipo *data frame*, se llama “eolica\_20” y contiene 11 columnas, una por cada una de las variables almacenadas en el archivo de Microsoft® Excel®. De estas variables, tres son de tipo cualitativo, formadas por cadenas de caracteres: el nombre de la empresa, “NOMBRE”; y el nombre del grupo empresarial matriz al que pertenece, “MATRIZ”. Puede explorarse el contenido del *data frame* y los principales estadísticos con la función summary():

summary (eolica\_20)

## NOMBRE RES ACTIVO FPIOS   
## Length:20 Min. : -5662 Min. : 109024 Min. : -77533   
## Class :character 1st Qu.: 2865 1st Qu.: 187240 1st Qu.: 27615   
## Mode :character Median : 7388 Median : 271636 Median : 77740   
## Mean : 50754 Mean : 1183599 Mean : 563678   
## 3rd Qu.: 21206 3rd Qu.: 813816 3rd Qu.: 219345   
## Max. :727548 Max. :13492812 Max. :6904824   
## NA's :1   
##   
## RENECO RENFIN LIQUIDEZ MARGEN   
## Min. :-2.8130 Min. :-359.773 Min. :0.0780 Min. :-302.03   
## 1st Qu.: 0.8765 1st Qu.: 1.664 1st Qu.:0.7342 1st Qu.: 12.39   
## Median : 3.6150 Median : 10.812 Median :1.2345 Median : 21.42   
## Mean : 2.9399 Mean : -3.450 Mean :1.4200 Mean : 16.40   
## 3rd Qu.: 4.7735 3rd Qu.: 25.312 3rd Qu.:1.5615 3rd Qu.: 38.56   
## Max. : 8.5860 Max. : 52.261 Max. :5.3300 Max. : 208.36   
## NA's :1   
##   
## SOLVENCIA APALANCA MATRIZ   
## Min. :-40.74 Min. :-6265.50 Length:20   
## 1st Qu.: 11.26 1st Qu.: 16.13 Class :character   
## Median : 23.68 Median : 145.93 Mode :character   
## Mean : 32.68 Mean : -17.17   
## 3rd Qu.: 52.62 3rd Qu.: 504.74   
## Max. : 99.08 Max. : 1019.62

Veremos cómo aparecen 11 variables con algunos estadísticos básicos.

R ha considerado la primera columna como una variable de tipo cualitativo (atributo). En realidad no es una variable, sino el nombre de los individuos o casos. Para evitar que R tome los nombres de los casos como una variable, podemos redefinir nuestro *data frame* diciéndole que considere esa primera columna como los *nombres de los individuos o filas*:

eolica\_20 <- data.frame(eolica\_20, row.names = 1)

En la línea anterior hemos asignado al data frame “eolica\_20” los propios datos de “eolica\_20”; pero indicando que la primera columna de datos no es una variable; sino el nombre de los casos. Si hacemos ahora el summary():

## RES ACTIVO FPIOS RENECO   
## Min. : -5662 Min. : 109024 Min. : -77533 Min. :-2.8130   
## 1st Qu.: 2865 1st Qu.: 187240 1st Qu.: 27615 1st Qu.: 0.8765   
## Median : 7388 Median : 271636 Median : 77740 Median : 3.6150   
## Mean : 50754 Mean : 1183599 Mean : 563678 Mean : 2.9399   
## 3rd Qu.: 21206 3rd Qu.: 813816 3rd Qu.: 219345 3rd Qu.: 4.7735   
## Max. :727548 Max. :13492812 Max. :6904824 Max. : 8.5860   
## NA's :1 NA's :1   
##   
## RENFIN LIQUIDEZ MARGEN SOLVENCIA   
## Min. :-359.773 Min. :0.0780 Min. :-302.03 Min. :-40.74   
## 1st Qu.: 1.664 1st Qu.:0.7342 1st Qu.: 12.39 1st Qu.: 11.26   
## Median : 10.812 Median :1.2345 Median : 21.42 Median : 23.68   
## Mean : -3.450 Mean :1.4200 Mean : 16.40 Mean : 32.68   
## 3rd Qu.: 25.312 3rd Qu.:1.5615 3rd Qu.: 38.56 3rd Qu.: 52.62   
## Max. : 52.261 Max. :5.3300 Max. : 208.36 Max. : 99.08   
##   
## APALANCA MATRIZ   
## Min. :-6265.50 Length:20   
## 1st Qu.: 16.13 Class :character   
## Median : 145.93 Mode :character   
## Mean : -17.17   
## 3rd Qu.: 504.74   
## Max. : 1019.62

Vemos que ya no aparece “NOMBRE” como variable, y en el *Environment* ya aparece el data frame “eolica\_20” con 20 observaciones (casos), pero con 10 variables (una menos).

Antes de seguir con la manipulación de nuestros datos, es preciso decir que existen otros muchos formatos de datos que pueden ser importados. Por ejemplo, con el paquete {readr} se pueden importar datos de archivos de texto de tipo tabular (por ejemplo, archivos \*.csv). Con el paquete {haven} se pueden capturar los datos almacenados en archivos de SPSS® (.sav), Stata® (.dta), SAS® (.sas7bdat), etc. Finamente, se pueden capturar datos almacenados en páginas web (archivos en formato JSON o XML, o en tablas HTML)) o en bases de datos gestionadas mediante diversos sistemas (SQLite, MySQL, MariaDB, PostgreSQL, Oracle®).

## 2.3 {dplyr}.

### 2.3.1 El Tidyverse. Cargando {dplyr}.

El ***Tidyverse*** es un conjunto de paquetes / librerías con una filosofía común, como es el uso de ciertas estructuras gramaticales, que facilitan muchas de las tareas y análisis que podrían hacerse con el lenguaje R estándar. Una buena obra para profundizar en el Tidyverse es Wickham and Grolemund ([2017](#ref-Wickham2017R)).

Uno de esos paquetes es {dplyr}, que proporciona una gramática más sencilla que la del lenguaje R convencional para **manipular** los objetos de estructuras de datos conocidos como ***data frames***.

Los data frames, como ya sabemos, son estructuras en las que se almacenan datos de modo que, por columnas, se disponen las variables del análisis; y por filas los casos que conforman la muestra / población.

Vamos a suponer que trabajamos dentro del proyecto que hemos creado previamente, de nombre “explora” (ver capítulo 1). Dentro de la carpeta del proyecto guardaremos el script llamado “explora\_dplyr.R” y el archivo de Microsoft® Excel® llamado “eolica\_20.xlsx”. Si abrimos este último archivo, comprobaremos que se compone de tres hojas. La hoja “Top 20” contiene los datos a importar desde R-Studio. Estos datos se corresponden con diferentes variables económico-financieras de 20 empresas productoras de electricidad mediante generación eólica.

Luego cerraremos el archivo de Microsoft® Excel®, “eolica\_20.xlsx”, y volveremos a R-Studio. Después, abriremos nuestro script “explora\_dplyr.R” con File → Open File… Este script contiene el programa que vamos a ir ejecutando en la práctica.

La primera línea / instrucción en los scripts suele ser:

rm(list = ls())

La instrucción tiene como objeto limpiar el *Environment* (memoria) de objetos de anteriores sesiones de trabajo.

Para importar los datos que hay en la hoja “Top 20” del archivo de Microsoft® Excel® llamado “eolica\_20.xlsx”, ejecutaremos el código:

# DATOS  
  
library(readxl)  
eolica\_20 <- read\_excel("eolica\_20.xlsx", sheet = "Top 20")

Podemos observar como en el *Environment* ya aparece un objeto. Este objeto es una estructura de datos tipo *data frame*, se llama “eolica\_20” y contiene 11 columnas. R ha considerado la primera columna como una variable de tipo cualitativo. En realidad, la primera columna no es una variable, sino que está formada por el nombre (identificador) de los diferentes casos u observaciones. Para evitar que R tome los nombres de los casos como una variable más, podemos redefinir nuestro *data frame* diciéndole que tome esa primera columna como los *nombres de los individuos*:

eolica\_20 <- data.frame(eolica\_20, row.names = 1)

En la línea anterior hemos asignado al *data frame* “eolica\_20” los propios datos de “eolica\_20”; pero indicando que la primera columna de datos no es una variable; sino el nombre de los casos.

A continuación, cargaremos el paquete {dplyr}. Si nunca antes se ha utilizado este paquete, cuando lo intentemos activar con la función library() nos dará un error o nos dirá que previamente hay que importarlo. En ese caso, iremos a la ventana inferior-derecha y pulsaremos la pestaña “Packages”, pulsaremos en Install, y emergerá una ventana donde dejaremos el “repositorio” que viene por defecto y, en el campo Packages, escribiremos el nombre del “paquete” (en nuestro caso {dplyr}). Una vez descargado el “paquete”, podremos ejecutar el código sin problemas:

#Cargando dplyr  
  
library (dplyr)

Para entender mejor la **sintaxis** que siguen las funciones o instrucciones a las que da acceso {dplyr}, hay que tener en cuenta lo siguiente:

* El primer argumento que tiene una función de {dplyr} es el *data frame* con el que se va a trabajar.
* Los otros argumentos describen qué hay que hacer con el *data frame* especificado en el primer argumento. Es posible referirse a las columnas (variables) del *data frame* con su nombre, **sin utilizar el operador $**.
* El valor de retorno es un **nuevo** *data frame*.

En los siguientes subapartados practicaremos con algunas de las principales funciones que aporta {dplyr}.

### 2.3.2 Seleccionando columnas de un data frame.

La función clave de {dplyr} para seleccionar una o varias columnas (variables) de un *data frame* es la función select().

Así, vamos a imaginar por ejemplo que queremos eliminar de nuestro *data frame* la variable (de tipo “carácter”) MATRIZ. Podremos ejecutar la asignación:

#Seleccionando variables  
  
eolica\_20 <-select(eolica\_20, -MATRIZ)  
summary (eolica\_20)

## RES ACTIVO FPIOS   
## Min. : -5662 Min. : 109024 Min. : -77533   
## 1st Qu.: 2865 1st Qu.: 187240 1st Qu.: 27615   
## Median : 7388 Median : 271636 Median : 77740   
## Mean : 50754 Mean : 1183599 Mean : 563678   
## 3rd Qu.: 21206 3rd Qu.: 813816 3rd Qu.: 219345   
## Max. :727548 Max. :13492812 Max. :6904824   
## NA's :1   
##   
## RENECO RENFIN LIQUIDEZ   
## Min. :-2.8130 Min. :-359.773 Min. :0.0780   
## 1st Qu.: 0.8765 1st Qu.: 1.664 1st Qu.:0.7342   
## Median : 3.6150 Median : 10.812 Median :1.2345   
## Mean : 2.9399 Mean : -3.450 Mean :1.4200   
## 3rd Qu.: 4.7735 3rd Qu.: 25.312 3rd Qu.:1.5615   
## Max. : 8.5860 Max. : 52.261 Max. :5.3300   
## NA's :1   
##   
## MARGEN SOLVENCIA APALANCA   
## Min. :-302.03 Min. :-40.74 Min. :-6265.50   
## 1st Qu.: 12.39 1st Qu.: 11.26 1st Qu.: 16.13   
## Median : 21.42 Median : 23.68 Median : 145.93   
## Mean : 16.40 Mean : 32.68 Mean : -17.17   
## 3rd Qu.: 38.56 3rd Qu.: 52.62 3rd Qu.: 504.74   
## Max. : 208.36 Max. : 99.08 Max. : 1019.62

Podemos verificar que, en el *Environment,* el *data frame* ha pasado a tener una variable menos (9), ya que hemos eliminado la variable MATRIZ. Es decir, con el guión “-” se pueden eliminar directamente variables de un *data frame*.

Ahora, suponemos que queremos visualizar las variables del *data frame* “eolica\_20”: ACTIVO, FPIOS, LIQUIDEZ, MARGEN, SOLVENCIA y APALANCA (es decir, todas las variables menos RES, RENECO, RENFIN). Para ello, ejecutaremos el código:

select(eolica\_20, ACTIVO, FPIOS, LIQUIDEZ, MARGEN, SOLVENCIA, APALANCA)

## ACTIVO FPIOS LIQUIDEZ MARGEN  
## Holding De Negocios De GAS SL. 13492812.0 6904824.000 1.020 91.152  
## Global Power Generation SA. 2002458.0 1740487.000 2.006 22.403  
## Naturgy Renovables SLU 1956869.0 318475.000 1.263 20.442  
## EDP Renovables España SLU 1275939.0 726783.000 1.596 47.193  
## Corporacion Acciona Eolica SL 864606.0 136064.000 0.788 20.091  
## Saeta Yield SA. 796886.4 665319.556 2.687 16.258  
## Elawan Energy SL. 443467.0 186302.006 0.595 208.357  
## Olivento SL 381207.0 58340.998 0.771 16.629  
## Parque Eolico La Boga SL. 303904.4 29316.797 1.407 1.001  
## Naturgy Wind, S.L. 273542.0 28418.000 1.364 39.575  
## Viesgo Renovables SL. 269730.0 177707.000 0.272 11.818  
## Al-Andalus Wind Power SL 249853.8 21466.121 1.550 12.582  
## Innogy Spain SA. 230338.5 85447.212 1.416 -18.025  
## Guzman Energia SL 190287.0 -77532.698 0.078 -19.193  
## Acciona Eolica Del Levante SL 188354.0 21769.000 2.855 27.520  
## Biovent Energia SA 183899.0 70033.000 1.206 22.792  
## Esquilvent SL 157630.6 48769.130 5.330 39.476  
## Eolica La Janda SL 153429.4 25206.748 1.184 38.256  
## Parque Eolico Santa Catalina SL 147742.5 -1664.755 0.388 31.780  
## WPD Wind Investment SL. 109023.8 108023.826 0.624 -302.027  
##   
## SOLVENCIA APALANCA  
## Holding De Negocios De GAS SL. 51.174 91.964  
## Global Power Generation SA. 86.917 1.044  
## Naturgy Renovables SLU 16.274 494.729  
## EDP Renovables España SLU 56.960 67.028  
## Corporacion Acciona Eolica SL 15.737 422.263  
## Saeta Yield SA. 83.489 17.067  
## Elawan Energy SL. 42.010 123.771  
## Olivento SL 15.304 534.761  
## Parque Eolico La Boga SL. 9.646 921.591  
## Naturgy Wind, S.L. 10.388 824.537  
## Viesgo Renovables SL. 65.883 13.330  
## Al-Andalus Wind Power SL 8.591 1019.616  
## Innogy Spain SA. 37.096 150.688  
## Guzman Energia SL -40.745 -343.542  
## Acciona Eolica Del Levante SL 11.557 743.754  
## Biovent Energia SA 38.082 141.163  
## Esquilvent SL 30.938 218.275  
## Eolica La Janda SL 16.428 480.122  
## Parque Eolico Santa Catalina SL -1.126 -6265.496  
## WPD Wind Investment SL. 99.082 0.000

Como no hemos asignado el resultado de la función a ningún “nombre”, R simplemente saca el resultado en pantalla; pero no guarda ningún objeto en el *Environment*. Si asignamos un select() a un “nombre”, se creará un *data frame* con ese nombre, y las variables seleccionadas:

eolica\_20A <-select(eolica\_20, ACTIVO, FPIOS, LIQUIDEZ, MARGEN, SOLVENCIA, APALANCA)  
summary (eolica\_20A)

## RES ACTIVO FPIOS   
## Min. : -5662 Min. : 109024 Min. : -77533   
## 1st Qu.: 2865 1st Qu.: 187240 1st Qu.: 27615   
## Median : 7388 Median : 271636 Median : 77740   
## Mean : 50754 Mean : 1183599 Mean : 563678   
## 3rd Qu.: 21206 3rd Qu.: 813816 3rd Qu.: 219345   
## Max. :727548 Max. :13492812 Max. :6904824   
## NA's :1   
##   
## RENECO RENFIN LIQUIDEZ   
## Min. :-2.8130 Min. :-359.773 Min. :0.0780   
## 1st Qu.: 0.8765 1st Qu.: 1.664 1st Qu.:0.7342   
## Median : 3.6150 Median : 10.812 Median :1.2345   
## Mean : 2.9399 Mean : -3.450 Mean :1.4200   
## 3rd Qu.: 4.7735 3rd Qu.: 25.312 3rd Qu.:1.5615   
## Max. : 8.5860 Max. : 52.261 Max. :5.3300   
## NA's :1   
##   
## MARGEN SOLVENCIA APALANCA   
## Min. :-302.03 Min. :-40.74 Min. :-6265.50   
## 1st Qu.: 12.39 1st Qu.: 11.26 1st Qu.: 16.13   
## Median : 21.42 Median : 23.68 Median : 145.93   
## Mean : 16.40 Mean : 32.68 Mean : -17.17   
## 3rd Qu.: 38.56 3rd Qu.: 52.62 3rd Qu.: 504.74   
## Max. : 208.36 Max. : 99.08 Max. : 1019.62

Podemos comprobar en el *Environment* cómo hay otro objeto *data frame* llamado “eolica\_20A”, con 6 variables (y los mismos 20 casos). Este *data frame* lo podríamos haber creado, también, eliminando del *data frame* original (“eolica\_20”), las variables que nos sobran:

eolica\_20A <-select(eolica\_20, -RES, -RENECO, -RENFIN)

Más aún, si nos fijamos bien, los nombres de todas las variables que hemos excluido empiezan por “RE”, a diferencia de las incluidas. Podríamos haber hecho también:

eolica\_20A <-select(eolica\_20, -(starts\_with("RE")))

Y de nuevo obtendríamos el mismo resultado. El argumento starts\_with() permite seleccionar variables cuyos nombres comienzan por cierta cadena de caracteres. También se puede hacer mismo con los caracteres finales (ends\_with()) o contenidos en alguna posición del nombre (contains()).

Otra posibilidad que tenemos es hacer una copia de un *data frame* rápidamente con el argumento everything(). Por ejemplo:

eolica\_20\_replica <-select(eolica\_20, everything())

Se ha creado el *date frame* “eolica\_20\_replica” que es una copia exacta de “eolica\_20”.

### 2.3.3 Seleccionando casos de un *data frame*.

Además de seleccionar variables, con {dplyr} también se pueden seleccionar casos que cumplan ciertas condiciones. La función para realizar este cometido es filter(). Por ejemplo, si queremos seleccionar las empresas eólicas con un resultado (variable RES) mayor o igual a 50.000 y presentarlas en pantalla, la instrucción será:

filter(eolica\_20, RES >= 50000)

## RES ACTIVO FPIOS RENECO RENFIN  
## Holding De Negocios De GAS SL. 727548 13492812 6904824 5.264 10.287  
## EDP Renovables España SLU 67033 1275939 726783 6.458 11.338  
##   
## LIQUIDEZ MARGEN SOLVENCIA APALANCA  
## Holding De Negocios De GAS SL. 1.020 91.152 51.174 91.964  
## EDP Renovables España SLU 1.596 47.193 56.960 67.028

Se pueden incluir varias condiciones en un mismo filtro. Por ejemplo, vamos a construir un nuevo *data frame* llamado “eolica\_20B” con las empresas que posean un resultado mayor o igual a 50000 y una rentabilidad económica (variable RENECO) inferior al 6%:

eolica\_20B <-filter(eolica\_20, RES >= 50000 & RENECO < 6)  
eolica\_20B

## RES ACTIVO FPIOS  
## Holding De Negocios De GAS SL. 727548 13492812 6904824  
##   
## RENECO RENFIN LIQUIDEZ  
## Holding De Negocios De GAS SL. 5.264 10.287 1.02  
##   
## MARGEN SOLVENCIA APALANCA  
## Holding De Negocios De GAS SL. 91.152 51.174 91.964

En el *Environment* aparecerá el *data frame* “eolica\_9B” con solo un caso: la empresa que cumple con ambas condiciones, introducidas mediante el operador lógico relacional “**&**”, que es el equivalente a la conjunción “y” o, dicho de otro modo, la intersección. Otro operador lógico relacional muy utilizado es la barra vertical “**|**”, que es el equivalente a la conjunción “o”, es decir, la unión.

Los filtros más usuales son >, <, >=, <=, == (igual, ojo, con dos símbolos de igualdad seguidos) y != (no igual).

### 2.3.4 Ordenando casos de un *data frame*.

Además de seleccionar determinados casos u observaciones (filas) de un *data frame*, con las funciones de {dplyr} también se pueden ordenar estos casos a partir de los valores de ciertas variables (columnas). La función a utilizar es arrange(). Esta función, por defecto, ordena los casos de modo **ascendente**. Por ejemplo:

arrange(eolica\_20, RENECO)

## RES ACTIVO FPIOS  
## Guzman Energia SL -5661.463 190287.0 -77532.698  
## Innogy Spain SA. -5268.573 230338.5 85447.212  
## WPD Wind Investment SL. -850.068 109023.8 108023.826  
## Parque Eolico La Boga SL. 11.940 303904.4 29316.797  
## Saeta Yield SA. 2084.476 796886.4 665319.556  
## Global Power Generation SA. 39995.000 2002458.0 1740487.000  
## Naturgy Renovables SLU 42737.000 1956869.0 318475.000  
## Al-Andalus Wind Power SL 4403.214 249853.8 21466.121  
## Olivento SL 7388.175 381207.0 58340.998  
## Elawan Energy SL. 12818.975 443467.0 186302.006  
## Naturgy Wind, S.L. 8500.000 273542.0 28418.000  
## Parque Eolico Santa Catalina SL 3645.278 147742.5 -1664.755  
## Biovent Energia SA NA 183899.0 70033.000  
## Corporacion Acciona Eolica SL 29592.000 864606.0 136064.000  
## Acciona Eolica Del Levante SL 6853.000 188354.0 21769.000  
## Holding De Negocios De GAS SL. 727548.000 13492812.0 6904824.000  
## EDP Renovables España SLU 67033.000 1275939.0 726783.000  
## Esquilvent SL 9010.214 157630.6 48769.130  
## Eolica La Janda SL 9880.091 153429.4 25206.748  
## Viesgo Renovables SL. 4609.000 269730.0 177707.000  
##   
## RENECO RENFIN LIQUIDEZ  
## Guzman Energia SL -2.813 6.904 0.078  
## Innogy Spain SA. -2.708 -7.302 1.416  
## WPD Wind Investment SL. -1.040 -1.049 0.624  
## Parque Eolico La Boga SL. 0.162 1.684 1.407  
## Saeta Yield SA. 0.360 0.432 2.687  
## Global Power Generation SA. 1.393 1.603 2.006  
## Naturgy Renovables SLU 1.959 12.043 1.263  
## Al-Andalus Wind Power SL 2.349 27.350 1.550  
## Olivento SL 2.553 16.684 0.771  
## Elawan Energy SL. 3.615 8.605 0.595  
## Naturgy Wind, S.L. 3.949 38.018 1.364  
## Parque Eolico Santa Catalina SL 4.053 -359.773 0.388  
## Biovent Energia SA 4.551 11.952 1.206  
## Corporacion Acciona Eolica SL 4.562 28.990 0.788  
## Acciona Eolica Del Levante SL 4.985 43.139 2.855  
## Holding De Negocios De GAS SL. 5.264 10.287 1.020  
## EDP Renovables España SLU 6.458 11.338 1.596  
## Esquilvent SL 7.621 24.633 5.330  
## Eolica La Janda SL 8.586 52.261 1.184  
## Viesgo Renovables SL. NA 3.200 0.272  
##   
## MARGEN SOLVENCIA APALANCA  
## Guzman Energia SL -19.193 -40.745 -343.542  
## Innogy Spain SA. -18.025 37.096 150.688  
## WPD Wind Investment SL. -302.027 99.082 0.000  
## Parque Eolico La Boga SL. 1.001 9.646 921.591  
## Saeta Yield SA. 16.258 83.489 17.067  
## Global Power Generation SA. 22.403 86.917 1.044  
## Naturgy Renovables SLU 20.442 16.274 494.729  
## Al-Andalus Wind Power SL 12.582 8.591 1019.616  
## Olivento SL 16.629 15.304 534.761  
## Elawan Energy SL. 208.357 42.010 123.771  
## Naturgy Wind, S.L. 39.575 10.388 824.537  
## Parque Eolico Santa Catalina SL 31.780 -1.126 -6265.496  
## Biovent Energia SA 22.792 38.082 141.163  
## Corporacion Acciona Eolica SL 20.091 15.737 422.263  
## Acciona Eolica Del Levante SL 27.520 11.557 743.754  
## Holding De Negocios De GAS SL. 91.152 51.174 91.964  
## EDP Renovables España SLU 47.193 56.960 67.028  
## Esquilvent SL 39.476 30.938 218.275  
## Eolica La Janda SL 38.256 16.428 480.122  
## Viesgo Renovables SL. 11.818 65.883 13.330

En cambio, para ordenar de modo descendente, hay que utilizar el argumento desc():

arrange(eolica\_20, desc(RENECO))

## RES ACTIVO FPIOS  
## Eolica La Janda SL 9880.091 153429.4 25206.748  
## Esquilvent SL 9010.214 157630.6 48769.130  
## EDP Renovables España SLU 67033.000 1275939.0 726783.000  
## Holding De Negocios De GAS SL. 727548.000 13492812.0 6904824.000  
## Acciona Eolica Del Levante SL 6853.000 188354.0 21769.000  
## Corporacion Acciona Eolica SL 29592.000 864606.0 136064.000  
## Biovent Energia SA NA 183899.0 70033.000  
## Parque Eolico Santa Catalina SL 3645.278 147742.5 -1664.755  
## Naturgy Wind, S.L. 8500.000 273542.0 28418.000  
## Elawan Energy SL. 12818.975 443467.0 186302.006  
## Olivento SL 7388.175 381207.0 58340.998  
## Al-Andalus Wind Power SL 4403.214 249853.8 21466.121  
## Naturgy Renovables SLU 42737.000 1956869.0 318475.000  
## Global Power Generation SA. 39995.000 2002458.0 1740487.000  
## Saeta Yield SA. 2084.476 796886.4 665319.556  
## Parque Eolico La Boga SL. 11.940 303904.4 29316.797  
## WPD Wind Investment SL. -850.068 109023.8 108023.826  
## Innogy Spain SA. -5268.573 230338.5 85447.212  
## Guzman Energia SL -5661.463 190287.0 -77532.698  
## Viesgo Renovables SL. 4609.000 269730.0 177707.000  
##   
## RENECO RENFIN LIQUIDEZ  
## Eolica La Janda SL 8.586 52.261 1.184  
## Esquilvent SL 7.621 24.633 5.330  
## EDP Renovables España SLU 6.458 11.338 1.596  
## Holding De Negocios De GAS SL. 5.264 10.287 1.020  
## Acciona Eolica Del Levante SL 4.985 43.139 2.855  
## Corporacion Acciona Eolica SL 4.562 28.990 0.788  
## Biovent Energia SA 4.551 11.952 1.206  
## Parque Eolico Santa Catalina SL 4.053 -359.773 0.388  
## Naturgy Wind, S.L. 3.949 38.018 1.364  
## Elawan Energy SL. 3.615 8.605 0.595  
## Olivento SL 2.553 16.684 0.771  
## Al-Andalus Wind Power SL 2.349 27.350 1.550  
## Naturgy Renovables SLU 1.959 12.043 1.263  
## Global Power Generation SA. 1.393 1.603 2.006  
## Saeta Yield SA. 0.360 0.432 2.687  
## Parque Eolico La Boga SL. 0.162 1.684 1.407  
## WPD Wind Investment SL. -1.040 -1.049 0.624  
## Innogy Spain SA. -2.708 -7.302 1.416  
## Guzman Energia SL -2.813 6.904 0.078  
## Viesgo Renovables SL. NA 3.200 0.272  
##   
## MARGEN SOLVENCIA APALANCA  
## Eolica La Janda SL 38.256 16.428 480.122  
## Esquilvent SL 39.476 30.938 218.275  
## EDP Renovables España SLU 47.193 56.960 67.028  
## Holding De Negocios De GAS SL. 91.152 51.174 91.964  
## Acciona Eolica Del Levante SL 27.520 11.557 743.754  
## Corporacion Acciona Eolica SL 20.091 15.737 422.263  
## Biovent Energia SA 22.792 38.082 141.163  
## Parque Eolico Santa Catalina SL 31.780 -1.126 -6265.496  
## Naturgy Wind, S.L. 39.575 10.388 824.537  
## Elawan Energy SL. 208.357 42.010 123.771  
## Olivento SL 16.629 15.304 534.761  
## Al-Andalus Wind Power SL 12.582 8.591 1019.616  
## Naturgy Renovables SLU 20.442 16.274 494.729  
## Global Power Generation SA. 22.403 86.917 1.044  
## Saeta Yield SA. 16.258 83.489 17.067  
## Parque Eolico La Boga SL. 1.001 9.646 921.591  
## WPD Wind Investment SL. -302.027 99.082 0.000  
## Innogy Spain SA. -18.025 37.096 150.688  
## Guzman Energia SL -19.193 -40.745 -343.542  
## Viesgo Renovables SL. 11.818 65.883 13.330

En el supuesto de que, por ejemplo, hubiera varias empresas con la misma rentabilidad económica (RENECO), podría añadirse otro criterio de ordenación con otra variable, que afectaría a tales empresas para deshacer el “empate” en rentabilidad económica. Por ejemplo, para ordenar de modo ascendente por rentabilidad y, en caso de que haya rentabilidades iguales, por liquidez (variable LIQUIDEZ), se ejecutaría:

arrange(eolica\_20, RENECO, LIQUIDEZ)

## RES ACTIVO FPIOS  
## Guzman Energia SL -5661.463 190287.0 -77532.698  
## Innogy Spain SA. -5268.573 230338.5 85447.212  
## WPD Wind Investment SL. -850.068 109023.8 108023.826  
## Parque Eolico La Boga SL. 11.940 303904.4 29316.797  
## Saeta Yield SA. 2084.476 796886.4 665319.556  
## Global Power Generation SA. 39995.000 2002458.0 1740487.000  
## Naturgy Renovables SLU 42737.000 1956869.0 318475.000  
## Al-Andalus Wind Power SL 4403.214 249853.8 21466.121  
## Olivento SL 7388.175 381207.0 58340.998  
## Elawan Energy SL. 12818.975 443467.0 186302.006  
## Naturgy Wind, S.L. 8500.000 273542.0 28418.000  
## Parque Eolico Santa Catalina SL 3645.278 147742.5 -1664.755  
## Biovent Energia SA NA 183899.0 70033.000  
## Corporacion Acciona Eolica SL 29592.000 864606.0 136064.000  
## Acciona Eolica Del Levante SL 6853.000 188354.0 21769.000  
## Holding De Negocios De GAS SL. 727548.000 13492812.0 6904824.000  
## EDP Renovables España SLU 67033.000 1275939.0 726783.000  
## Esquilvent SL 9010.214 157630.6 48769.130  
## Eolica La Janda SL 9880.091 153429.4 25206.748  
## Viesgo Renovables SL. 4609.000 269730.0 177707.000  
##   
## RENECO RENFIN LIQUIDEZ  
## Guzman Energia SL -2.813 6.904 0.078  
## Innogy Spain SA. -2.708 -7.302 1.416  
## WPD Wind Investment SL. -1.040 -1.049 0.624  
## Parque Eolico La Boga SL. 0.162 1.684 1.407  
## Saeta Yield SA. 0.360 0.432 2.687  
## Global Power Generation SA. 1.393 1.603 2.006  
## Naturgy Renovables SLU 1.959 12.043 1.263  
## Al-Andalus Wind Power SL 2.349 27.350 1.550  
## Olivento SL 2.553 16.684 0.771  
## Elawan Energy SL. 3.615 8.605 0.595  
## Naturgy Wind, S.L. 3.949 38.018 1.364  
## Parque Eolico Santa Catalina SL 4.053 -359.773 0.388  
## Biovent Energia SA 4.551 11.952 1.206  
## Corporacion Acciona Eolica SL 4.562 28.990 0.788  
## Acciona Eolica Del Levante SL 4.985 43.139 2.855  
## Holding De Negocios De GAS SL. 5.264 10.287 1.020  
## EDP Renovables España SLU 6.458 11.338 1.596  
## Esquilvent SL 7.621 24.633 5.330  
## Eolica La Janda SL 8.586 52.261 1.184  
## Viesgo Renovables SL. NA 3.200 0.272  
##   
## MARGEN SOLVENCIA APALANCA  
## Guzman Energia SL -19.193 -40.745 -343.542  
## Innogy Spain SA. -18.025 37.096 150.688  
## WPD Wind Investment SL. -302.027 99.082 0.000  
## Parque Eolico La Boga SL. 1.001 9.646 921.591  
## Saeta Yield SA. 16.258 83.489 17.067  
## Global Power Generation SA. 22.403 86.917 1.044  
## Naturgy Renovables SLU 20.442 16.274 494.729  
## Al-Andalus Wind Power SL 12.582 8.591 1019.616  
## Olivento SL 16.629 15.304 534.761  
## Elawan Energy SL. 208.357 42.010 123.771  
## Naturgy Wind, S.L. 39.575 10.388 824.537  
## Parque Eolico Santa Catalina SL 31.780 -1.126 -6265.496  
## Biovent Energia SA 22.792 38.082 141.163  
## Corporacion Acciona Eolica SL 20.091 15.737 422.263  
## Acciona Eolica Del Levante SL 27.520 11.557 743.754  
## Holding De Negocios De GAS SL. 91.152 51.174 91.964  
## EDP Renovables España SLU 47.193 56.960 67.028  
## Esquilvent SL 39.476 30.938 218.275  
## Eolica La Janda SL 38.256 16.428 480.122  
## Viesgo Renovables SL. 11.818 65.883 13.330

Obviamente, en este ejemplo concreto el resultado es el mismo que se obtuvo con arrange(eolica\_20, RENECO), puesto que no hay rentabilidades iguales entre las 20 empresas de la muestra.

### 2.3.5 Cambiando el nombre de las variables de un *data frame*.

{dplyr} cuenta con una función que cambia fácilmente el nombre de una variable o columna de un *data frame*: la función rename(). Por ejemplo, si queremos cambiar el nombre de la variable SOLVENCIA por SOLVE, simplemente ejecutaremos:

#Renombrando variables  
eolica\_20 <- rename(eolica\_20, SOLVE = SOLVENCIA)

Podemos comprobar en el *Environment*, despegando el objeto “eolica\_20”, cómo ya no aparece la variable SOLVENCIA; pero sí SOLVE en su lugar (obviamente, con los mismos datos). Es necesario tener en cuenta que en el **lado izquierdo** de la igualdad hay que poner el **nuevo nombre**, y en la derecha el antiguo. Además, en el mismo rename() se pueden cambiar los nombres de **varias variables**, separando las igualdades correspondientes con comas.

### 2.3.6 **Añadiendo variables como transformación de otras variables en un *data frame*.**

El paquete {dplyr} también permite añadir a un *data frame* variables que son el resultado de **someter a otras variables a diversas transformaciones**. La función para realizar este cometido es mutate().

Así, por ejemplo, imaginemos que necesitamos calcular una variable como el cociente entre los resultados obtenidos y el activo. A esta nueva variable la denominaremos RATIO. El código será:

# Añadiendo variables como transformacion de otras variables  
eolica\_20 <- mutate (eolica\_20, RATIO = RES / ACTIVO)  
summary(eolica\_20)

## RES ACTIVO FPIOS RENECO   
## Min. : -5662 Min. : 109024 Min. : -77533 Min. :-2.8130   
## 1st Qu.: 2865 1st Qu.: 187240 1st Qu.: 27615 1st Qu.: 0.8765   
## Median : 7388 Median : 271636 Median : 77740 Median : 3.6150   
## Mean : 50754 Mean : 1183599 Mean : 563678 Mean : 2.9399   
## 3rd Qu.: 21206 3rd Qu.: 813816 3rd Qu.: 219345 3rd Qu.: 4.7735   
## Max. :727548 Max. :13492812 Max. :6904824 Max. : 8.5860   
## NA's :1 NA's :1   
##   
## RENFIN LIQUIDEZ MARGEN SOLVE   
## Min. :-359.773 Min. :0.0780 Min. :-302.03 Min. :-40.74   
## 1st Qu.: 1.664 1st Qu.:0.7342 1st Qu.: 12.39 1st Qu.: 11.26   
## Median : 10.812 Median :1.2345 Median : 21.42 Median : 23.68   
## Mean : -3.450 Mean :1.4200 Mean : 16.40 Mean : 32.68   
## 3rd Qu.: 25.312 3rd Qu.:1.5615 3rd Qu.: 38.56 3rd Qu.: 52.62   
## Max. : 52.261 Max. :5.3300 Max. : 208.36 Max. : 99.08   
##   
## APALANCA RATIO   
## Min. :-6265.50 Min. :-0.029752   
## 1st Qu.: 16.13 1st Qu.: 0.009852   
## Median : 145.93 Median : 0.021840   
## Mean : -17.17 Mean : 0.022180   
## 3rd Qu.: 504.74 3rd Qu.: 0.035305   
## Max. : 1019.62 Max. : 0.064395   
## NA's :1

En la transformación de variables mediante la función mutate(), se pueden utilizar **funciones integradas en otros paquetes** de R. Por ejemplo, si queremos calcular la variable ACTIVOS\_ACUM como la variable que recoge los activos acumulados de las empresas, comenzando por la empresa con menor activo, podríamos utilizar la función cumsum() del paquete {base}, y hacer:

eolica\_20 <- arrange(eolica\_20, ACTIVO)  
eolica\_20 <- mutate (eolica\_20, ACTIVOS\_ACUM = cumsum(ACTIVO))  
select(eolica\_20, ACTIVO, ACTIVOS\_ACUM)

## ACTIVO ACTIVOS\_ACUM  
## WPD Wind Investment SL. 109023.8 109023.8  
## Parque Eolico Santa Catalina SL 147742.5 256766.3  
## Eolica La Janda SL 153429.4 410195.8  
## Esquilvent SL 157630.6 567826.4  
## Biovent Energia SA 183899.0 751725.4  
## Acciona Eolica Del Levante SL 188354.0 940079.4  
## Guzman Energia SL 190287.0 1130366.4  
## Innogy Spain SA. 230338.5 1360704.9  
## Al-Andalus Wind Power SL 249853.8 1610558.7  
## Viesgo Renovables SL. 269730.0 1880288.7  
## Naturgy Wind, S.L. 273542.0 2153830.7  
## Parque Eolico La Boga SL. 303904.4 2457735.1  
## Olivento SL 381207.0 2838942.0  
## Elawan Energy SL. 443467.0 3282409.0  
## Saeta Yield SA. 796886.4 4079295.4  
## Corporacion Acciona Eolica SL 864606.0 4943901.4  
## EDP Renovables España SLU 1275939.0 6219840.4  
## Naturgy Renovables SLU 1956869.0 8176709.4  
## Global Power Generation SA. 2002458.0 10179167.4  
## Holding De Negocios De GAS SL. 13492812.0 23671979.4

Podemos verificar cómo se ha integrado en el *data frame* la variable ACTIVOS\_ACUM.

Un último ejemplo de adición de una variable que es transformación de otras. En este caso, crearemos la variable TAM (tamaño), que es **categórica** (los datos son conjuntos de carcteres). Esta variable toma valor “G” para las empresas con un valor de la variable ACTIVO mayor que 1.000.000, y “P” para las que tengan un valor en la variable ACTIVO menor o igual a 1.000.000. Para calcular automáticamente esta nueva variable categórica, utilizaremos la función de {base} cut(). De este modo, haremos:

eolica\_20 <- mutate(eolica\_20, TAM = cut(ACTIVO,  
 breaks = c(-Inf, 1000000, Inf), labels = c("P", "G")))  
select(eolica\_20, ACTIVO, TAM)

## ACTIVO TAM  
## WPD Wind Investment SL. 109023.8 P  
## Parque Eolico Santa Catalina SL 147742.5 P  
## Eolica La Janda SL 153429.4 P  
## Esquilvent SL 157630.6 P  
## Biovent Energia SA 183899.0 P  
## Acciona Eolica Del Levante SL 188354.0 P  
## Guzman Energia SL 190287.0 P  
## Innogy Spain SA. 230338.5 P  
## Al-Andalus Wind Power SL 249853.8 P  
## Viesgo Renovables SL. 269730.0 P  
## Naturgy Wind, S.L. 273542.0 P  
## Parque Eolico La Boga SL. 303904.4 P  
## Olivento SL 381207.0 P  
## Elawan Energy SL. 443467.0 P  
## Saeta Yield SA. 796886.4 P  
## Corporacion Acciona Eolica SL 864606.0 P  
## EDP Renovables España SLU 1275939.0 G  
## Naturgy Renovables SLU 1956869.0 G  
## Global Power Generation SA. 2002458.0 G  
## Holding De Negocios De GAS SL. 13492812.0 G

Podemos advertir cómo la función cut(), que incluimos dentro de nuestra función de {dplyr} mutate(), tiene, a su vez, varios argumentos: la variable numérica de referencia (ACTIVO); el argumento **“breaks”**, en el que decimos los intervalos en que quedarán divididos los casos (uno, de menos infinito a 1.000.000; y otro de 1.000.000 a más infinito), y **“labels”**, que es el valor que tomará la variable creada (TAM) según el intervalo en el que se sitúe cada caso de la muestra.

Cabe destacar que podíamos haber escrito el código para crear la variable TAM de un modo más elegante y cómodo, utilizando el operador **“*pipe*”** (**%>%**). Este operador permite concatenar una serie de instrucciones:

eolica\_20 <- eolica\_20 %>% mutate(TAM = cut(ACTIVO,  
 breaks = c(-Inf, 1000000, Inf), labels = c("P", "G")))  
select(eolica\_20, ACTIVO, TAM)

## ACTIVO TAM  
## WPD Wind Investment SL. 109023.8 P  
## Parque Eolico Santa Catalina SL 147742.5 P  
## Eolica La Janda SL 153429.4 P  
## Esquilvent SL 157630.6 P  
## Biovent Energia SA 183899.0 P  
## Acciona Eolica Del Levante SL 188354.0 P  
## Guzman Energia SL 190287.0 P  
## Innogy Spain SA. 230338.5 P  
## Al-Andalus Wind Power SL 249853.8 P  
## Viesgo Renovables SL. 269730.0 P  
## Naturgy Wind, S.L. 273542.0 P  
## Parque Eolico La Boga SL. 303904.4 P  
## Olivento SL 381207.0 P  
## Elawan Energy SL. 443467.0 P  
## Saeta Yield SA. 796886.4 P  
## Corporacion Acciona Eolica SL 864606.0 P  
## EDP Renovables España SLU 1275939.0 G  
## Naturgy Renovables SLU 1956869.0 G  
## Global Power Generation SA. 2002458.0 G  
## Holding De Negocios De GAS SL. 13492812.0 G

Podríamos interpretar la línea de código así: “asigna al *data frame*”eolica\_20” sus propios datos, después (%>%) crea la variable TAM con la función cut() y añádela a “eolica\_20”.

### 2.3.7 Extrayendo y sintetizando información de las variables de un *data frame*.

Otra posibilidad que permite {dplyr} es extraer y sintetizar la información de las variables contenidas en un *data frame*. Para ello, nos ayudaremos de la función summarise(). Como ejemplo, calculemos la *rentabilidad financiera* media de las 20 empresas:

#Extrayendo información de las variables de un data frame  
summarise(eolica\_20, RENFIN\_media = mean(RENFIN))

## RENFIN\_media  
## 1 -3.45005

A veces, es de gran utilidad combinar summarise() con group\_by(), que extrae la información por grupos definidos por una de las variables. Para ilustrarlo, vamos a utilizar la variable recién creada TAM, para hacer dos grupos de empresas: las de menor (“P”) y las de mayor (“G”) volumen de activo; tras lo cual calcularemos la media de las rentabilidades para cada grupo:

eolica\_20 %>% group\_by(TAM) %>% summarise(RENFIN\_media = mean(RENFIN))

## # A tibble: 2 × 2  
## TAM RENFIN\_media  
## <fct> <dbl>  
## 1 P -6.52  
## 2 G 8.82

Hemos utilizado el operador *pipe* (**%>%)** para concatenar diferentes instrucciones de {dplyr}: primero agrupar casos, y luego calcular las medias de cada grupo. Es decir, en este caso se podría “traducir” la línea de código como: “Toma el *data frame*”eolica\_9”, divide los casos en grupos según el valor de la variable TAM, y para cada grupo calcula la media de la variable RENFIN”.

## 2.4 Exportando datos.

Antes de concluir el capítulo, vamos a tratar brevemente el aspecto de la exportación de datos.

R cuenta con un **formato propio de datos**, que se traduce en archivos de extensión “RData”, y que puede incluir cualquier objeto de R. Como ejemplo, en el siguiente *script*, llamado “explora\_exporta.R” (obtener aquí), vamos a importar los datos del archivo de Microsoft (R) Excel (R) “eolica\_20.xlsx”, y el *data frame* donde almacenemos los datos vamos a exportarlo como el archivo de datos de R “eolica\_20.RData”. Posteriormente, borraremos el *data frame* del *Environment* y recuperaremos los datos cargando ese archivo “eolica\_20.RData”. Por supuesto, seguimos trabajando, como en todo el capítulo, en el proyecto “explora”.

Tras abrir el *script* “explora\_exporta.R”, las primeras líneas de código que veremos serán las que ya hemos estudiado para borrar el contenido del *Environment*, importar los datos de la hoja “Top 20” del archivo “eolica\_20.xlsx” (situado en nuestra carpeta de proyecto), y tratar la variable “NOMBRE” para transformarla en el conjunto de nombres de las filas:

# Exportando datos de empresas eolicas (disculpad la falta de tildes)  
  
rm(list = ls())  
  
# DATOS  
  
library(readxl)  
eolica\_20 <- read\_excel("eolica\_20.xlsx", sheet = "Top 20")  
eolica\_20 <- data.frame(eolica\_20, row.names = 1)  
summary(eolica\_20)

## RES ACTIVO FPIOS RENECO   
## Min. : -5662 Min. : 109024 Min. : -77533 Min. :-2.8130   
## 1st Qu.: 2865 1st Qu.: 187240 1st Qu.: 27615 1st Qu.: 0.8765   
## Median : 7388 Median : 271636 Median : 77740 Median : 3.6150   
## Mean : 50754 Mean : 1183599 Mean : 563678 Mean : 2.9399   
## 3rd Qu.: 21206 3rd Qu.: 813816 3rd Qu.: 219345 3rd Qu.: 4.7735   
## Max. :727548 Max. :13492812 Max. :6904824 Max. : 8.5860   
## NA's :1 NA's :1   
##   
## RENFIN LIQUIDEZ MARGEN SOLVENCIA   
## Min. :-359.773 Min. :0.0780 Min. :-302.03 Min. :-40.74   
## 1st Qu.: 1.664 1st Qu.:0.7342 1st Qu.: 12.39 1st Qu.: 11.26   
## Median : 10.812 Median :1.2345 Median : 21.42 Median : 23.68   
## Mean : -3.450 Mean :1.4200 Mean : 16.40 Mean : 32.68   
## 3rd Qu.: 25.312 3rd Qu.:1.5615 3rd Qu.: 38.56 3rd Qu.: 52.62   
## Max. : 52.261 Max. :5.3300 Max. : 208.36 Max. : 99.08   
##   
## APALANCA MATRIZ   
## Min. :-6265.50 Length:20   
## 1st Qu.: 16.13 Class :character   
## Median : 145.93 Mode :character   
## Mean : -17.17   
## 3rd Qu.: 504.74   
## Max. : 1019.62

Posteriormente, se exportará el data frame “eolica\_20” al archivo de formato R, “eolica\_20.RData”, mediante la función save:

# Exportando data frame a formato R (.RData)  
  
save(eolica\_20, file = "eolica\_20.RData")

Puede comprobarse cómo se ha generado el archivo correspondiente en la carpeta de proyecto. Para comprobar que la exportación es correcta, vamos a borrar del *Environment* el *data frame* “eolica\_20”. Después, cargaremos el archivo “eolica\_20.RData”. Como resultado, podremos comprobar que tenemos un nuevo *data frame* “eolica\_20” que es exactamente igual al que teníamos al principio:

# Borrando el data frame eolica\_20  
  
rm(eolica\_20)  
  
# Importando el archivo .RData con los mismos datos  
  
load("eolica\_20.RData")  
summary (eolica\_20)

## RES ACTIVO FPIOS RENECO   
## Min. : -5662 Min. : 109024 Min. : -77533 Min. :-2.8130   
## 1st Qu.: 2865 1st Qu.: 187240 1st Qu.: 27615 1st Qu.: 0.8765   
## Median : 7388 Median : 271636 Median : 77740 Median : 3.6150   
## Mean : 50754 Mean : 1183599 Mean : 563678 Mean : 2.9399   
## 3rd Qu.: 21206 3rd Qu.: 813816 3rd Qu.: 219345 3rd Qu.: 4.7735   
## Max. :727548 Max. :13492812 Max. :6904824 Max. : 8.5860   
## NA's :1 NA's :1   
##   
## RENFIN LIQUIDEZ MARGEN SOLVENCIA   
## Min. :-359.773 Min. :0.0780 Min. :-302.03 Min. :-40.74   
## 1st Qu.: 1.664 1st Qu.:0.7342 1st Qu.: 12.39 1st Qu.: 11.26   
## Median : 10.812 Median :1.2345 Median : 21.42 Median : 23.68   
## Mean : -3.450 Mean :1.4200 Mean : 16.40 Mean : 32.68   
## 3rd Qu.: 25.312 3rd Qu.:1.5615 3rd Qu.: 38.56 3rd Qu.: 52.62   
## Max. : 52.261 Max. :5.3300 Max. : 208.36 Max. : 99.08   
##   
## APALANCA MATRIZ   
## Min. :-6265.50 Length:20   
## 1st Qu.: 16.13 Class :character   
## Median : 145.93 Mode :character   
## Mean : -17.17   
## 3rd Qu.: 504.74   
## Max. : 1019.62

Por supuesto, hay más formatos en los que se pueden exportar datos desde R. Por ejemplo, **a un archivo de Microsoft (R) Excel (R)**. Un modo de hacerlo es haciendo uso de la función write\_xlsx() del paquete {writexl}. Para que en la hoja de cálculo resultante se incluyan los nombres de las filas (empresas eólicas), hemos tenido previamente que crear un vector con el nombre de estas (vector “NOMBRE”), mediante la función row.names(), y unir ese vector al *data frame* “eolica\_20”, a modo de primera columna, creando un nuevo finalmente un *data frame* llamado “eolica\_20n”, para lo que se ha utilizado la función cbind(), que permite **pegar columnas de datos** que tengan un mismo número de filas.

Como resultado de todo el código, se ha obtenido el archivo de Microsoft (R) Excel (R) “eolica\_20\_new.xlsx”:

# Exportando el data frame eolica\_20 a Microsoft Excel  
  
library(writexl)  
NOMBRE <- row.names(eolica\_20)  
eolica\_20n <- cbind(NOMBRE, eolica\_20)  
write\_xlsx(eolica\_20n, path = "eolica\_20\_new.xlsx")  
  
#Fin del script

## 2.5 Materiales para realizar las prácticas del capítulo.

En esta sección se muestran los links de acceso a los diferentes materiales (*scripts*, datos…) necesarios para llevar a cabo los contenidos prácticos del capítulo.

**Datos (en formato Microsoft (R) Excel (R)):**

* eolica\_20.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1YDFCd0Z7udeIi6WdNYwthYHGmJSrEfG1/edit?usp=drive_link&ouid=115375878280465826079&rtpof=true&sd=true))

**Scripts:**

* explora\_eolica.R ([obtener aquí](https://drive.google.com/file/d/1_rt1TtlYGU0VWtnvkWLpxM_MDGOrSFYZ/view?usp=drive_link))
* explora\_dplyr.R ([obtener aquí](https://drive.google.com/file/d/1Is4XOd6EmFswB6w2eZfC8_pp2mtu7aQK/view?usp=drive_link))
* explora\_exporta.R ([obtener aquí](https://drive.google.com/file/d/1NVkpJ6toD3pWJsE0w628Sf4e_c5dcffR/view?usp=sharing))

# 3 Gráficos.

## 3.1 Tidyverse para gráficos: ggplot2.

R, en su instalación básica, cuenta con funciones destinadas a crear gráficos y, de este modo, visualizar nuestros datos a fin de generar información y extraer conclusiones de un modo sencillo. No obstante, estas funciones, a veces, se quedan “cortas”, o requieren de un complejo y/o extenso código. Esta es la razón por la que en el *Tidyverse* se incluyó un paquete específico destinado a la construcción de gráficos de un modo flexible y amigable. Recordemos que el *Tidyverse* es un conjunto de paquetes con una filosofía común, como es el uso de ciertas estructuras gramaticales, que facilitan muchas de las tareas y análisis que podrían hacerse con el lenguaje R estándar.

Este paquete destinado a la producción de gráficos es {ggplot2}, que proporciona unas herramientas muy flexibles para visualizar conjuntos de datos. A continuación, se expondrán los fundamentos de la sintaxis de {ggplot2} y se indicará cómo construir algunos de los gráficos más habituales.

Para ilustrar la creación de gráficos, vamos a suponer que trabajamos dentro de un proyecto que hemos creado previamente, de nombre “explora”. Dentro de la carpeta del proyecto guardaremos el *script* llamado “[explora\_ggplot2.R](https://drive.google.com/file/d/1R0qYnrjcel4SOH9lVI1_qkccKYXBLLDC/view?usp=sharing)” y el archivo de *Microsoft® Excel®* llamado “[eolica\_100.xlsx](https://docs.google.com/spreadsheets/d/1Up5ZWR57rgIkmYJcr2Pek6kUdG_yd4vu/edit?usp=drive_link&ouid=115375878280465826079&rtpof=true&sd=true)”.

Si abrimos este último archivo, comprobaremos que se compone de tres hojas. La primera muestra un aviso sobre el uso de los datos; la segunda recoge la descripción de las variables consideradas; y la tercera (hoja “Datos”) guarda los datos que debemos importar desde R-Studio. Estos datos se corresponden con diferentes variables económico-financieras de las 100 empresas productoras de electricidad mediante generación eólica con mayor volumen de activo total.

Tras abrir el *script* “explora\_ggplot2.R” e el editor de R-Studio, observaremos que la primera línea / instrucción es:

rm(list = ls())

La instrucción tiene como objeto limpiar el *Environment* (memoria) de objetos de anteriores sesiones de trabajo.

Para importar los datos que hay en la hoja “Datos” del archivo de Microsoft® Excel® llamado “eolica\_100.xlsx”, ejecutaremos el código:

library(readxl)  
eolica\_100 <- read\_excel("eolica\_100.xlsx", sheet = "Datos")  
summary (eolica\_100)

## NOMBRE RES ACTIVO FPIOS   
## Length:100 Min. : -5661.5 Min. : 24944 Min. : -77533   
## Class :character 1st Qu.: 670.2 1st Qu.: 34437 1st Qu.: 2305   
## Mode :character Median : 2114.7 Median : 46896 Median : 11936   
## Mean : 11477.3 Mean : 274756 Mean : 123743   
## 3rd Qu.: 3951.2 3rd Qu.: 85542 3rd Qu.: 28292   
## Max. :727548.0 Max. :13492812 Max. :6904824   
##   
## RENECO RENFIN LIQUIDEZ MARGEN   
## Min. :-3.446 Min. :-359.773 Min. : 0.0140 Min. :-2248.157   
## 1st Qu.: 1.421 1st Qu.: 2.556 1st Qu.: 0.6567 1st Qu.: 12.126   
## Median : 4.144 Median : 15.326 Median : 1.0650 Median : 26.618   
## Mean : 5.294 Mean : 17.243 Mean : 2.7214 Mean : 3.583   
## 3rd Qu.: 7.904 3rd Qu.: 31.307 3rd Qu.: 1.6078 3rd Qu.: 39.580   
## Max. :35.262 Max. : 588.190 Max. :128.4330 Max. : 400.899   
##   
## SOLVENCIA APALANCA MATRIZ DIMENSION   
## Min. :-40.74 Min. :-8254.11 Length:100 Length:100   
## 1st Qu.: 4.71 1st Qu.: 16.13 Class :character Class :character   
## Median : 16.65 Median : 161.97 Mode :character Mode :character   
## Mean : 27.57 Mean : 345.03   
## 3rd Qu.: 45.59 3rd Qu.: 623.13   
## Max. : 99.08 Max. :12244.35

Podemos observar cómo, en el *Environment,* ya aparece un *data frame* que se llama “eolica\_100”, y contiene 12 columnas. R ha considerado la primera columna como una variable de tipo cualitativo o atributo. En realidad, esa columna no es una variable, sino que está formada por los nombres de los diferentes casos u observaciones (filas). Para evitar que R tome la columna de los nombres de los casos como una variable más, podemos redefinir nuestro *data frame* diciéndole que considere esa primera columna como el conjunto de los *nombres de los individuos o casos*:

eolica\_100 <- data.frame(eolica\_100, row.names = 1)

En la línea anterior, hemos asignado al *data frame* “eolica\_100” los propios datos de “eolica\_100”; pero indicando que la primera columna no es una variable; sino que contiene el nombre de los casos. Si hacemos ahora un summary():

summary (eolica\_100)

## RES ACTIVO FPIOS RENECO   
## Min. : -5661.5 Min. : 24944 Min. : -77533 Min. :-3.446   
## 1st Qu.: 670.2 1st Qu.: 34437 1st Qu.: 2305 1st Qu.: 1.421   
## Median : 2114.7 Median : 46896 Median : 11936 Median : 4.144   
## Mean : 11477.3 Mean : 274756 Mean : 123743 Mean : 5.294   
## 3rd Qu.: 3951.2 3rd Qu.: 85542 3rd Qu.: 28292 3rd Qu.: 7.904   
## Max. :727548.0 Max. :13492812 Max. :6904824 Max. :35.262   
##   
## RENFIN LIQUIDEZ MARGEN SOLVENCIA   
## Min. :-359.773 Min. : 0.0140 Min. :-2248.157 Min. :-40.74   
## 1st Qu.: 2.556 1st Qu.: 0.6567 1st Qu.: 12.126 1st Qu.: 4.71   
## Median : 15.326 Median : 1.0650 Median : 26.618 Median : 16.65   
## Mean : 17.243 Mean : 2.7214 Mean : 3.583 Mean : 27.57   
## 3rd Qu.: 31.307 3rd Qu.: 1.6078 3rd Qu.: 39.580 3rd Qu.: 45.59   
## Max. : 588.190 Max. :128.4330 Max. : 400.899 Max. : 99.08   
##   
## APALANCA MATRIZ DIMENSION   
## Min. :-8254.11 Length:100 Length:100   
## 1st Qu.: 16.13 Class :character Class :character   
## Median : 161.97 Mode :character Mode :character   
## Mean : 345.03   
## 3rd Qu.: 623.13   
## Max. :12244.35

Comprobamos que ya no aparece NOMBRE como variable y que, en el *Environment*, se recoge el *data frame* “eolica\_100” con 100 casos y con 11 variables.

## 3.2 Gráficos de una variable: histogramas, gráficos de densidad, gráficos de caja o *boxplots*.

A continuación, cargaremos el paquete {ggplot2}. Si nunca antes se ha utilizado, cuando lo intentemos activar con la función library() nos dará un error, advirtiendo que previamente hay que instalarlo. En ese caso, iremos a la ventana inferior-derecha de R-Studio y pulsaremos en la pestaña Packages, luego en Install, y emergerá una ventana donde dejaremos el “repositorio” que viene por defecto y, en el campo Packages, escribiremos el nombre del “paquete” (en nuestro caso *ggplot2*). Una vez descargado el “paquete”, podremos ejecutar el código sin problemas:

# Cargando ggplot2  
library (ggplot2)

La primera instrucción para crear un gráfico con el paquete {ggplot2} es ggplot(). A continuación, entre paréntesis, se deberán aportar una serie de argumentos o informaciones. Estas informaciones irán definiendo el gráfico en mayor o menor detalle.

En realidad, lo que se hace es definir el conjunto de datos a representar (que suelen estar contenidos en un *data frame*, o en varios), y a partir de ellos se van añadiendo capas gráficas o “***geoms****”,* que son caracterizadas con ciertos atributos estéticos *(”*aesthetics”*, o* “aes”).

### 3.2.1 Histograma.

Uno de los gráficos indispensables para tener una idea de la distribución de frecuencias que siguen los casos (en nuestro ejemplo, las empresas eólicas) con relación a una variable métrica es el **histograma**. Vamos a construir un histograma para la variable de rentabilidad económica, RENECO. El código será:

# Histograma  
ggplot(data = eolica\_100, map = aes(x = RENECO)) +  
 geom\_histogram()

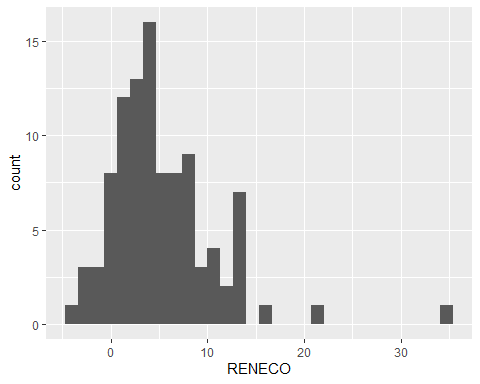
Como acabamos de decir, en primer lugar viene el comando ggplot(), seguido de unos paréntesis que recogen ciertas informaciones:

* **“data =”**, seguido de la fuente que almacena los datos a graficar (en nuestro caso, el *data frame* “eolica\_100”).
* **“map =”**, o “mapeo”, que define los aspectos del gráfico que dependen del valor de alguna o algunas variables. Siempre que alguna característica del gráfico no sea “fija”, sino que dependa de los valores que toma una variable, tal variable deberá ir indicada dentro de un *elemento estético* (*aes*). En el código de ejemplo, el elemento *aes* sirve para indicar que las coordenadas del eje x que toman los casos a representar, dependen de los valores de la variable RENECO.

Para indicar que las siguientes líneas continúan con el código del gráfico, se añade al final de esta línea el símbolo “**+**”.

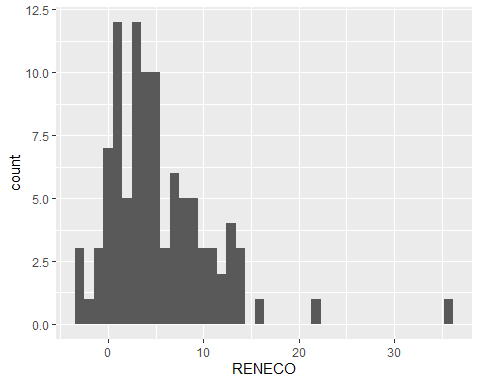
En la segunda línea, se establece el tipo de gráfico que se va a realizar, mediante la inclusión de un elemento ***geom***. Para decir que lo que queremos construir es un histograma, el elemento *geom* será geom\_histogram().

El resultado del código anterior es el siguiente gráfico:



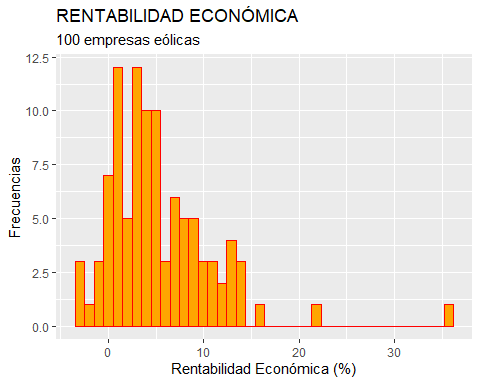
Por supuesto, {ggplot2} permite personalizar y refinar la apariencia del gráfico. Uno de los aspectos que nos puede interesar modificar es el número de intervalos en los que queda dividido el rango de valores que puede tomar la variable (“grosor” de las barras), o *bins*. Por defecto, el número es 30. Para incrementar este número de barras a 40, por ejemplo, añadiremos en la línea del *geom* el argumento **“bins =”**:

ggplot(data = eolica\_100, map = aes(x = RENECO)) +  
 geom\_histogram(bins = 40)



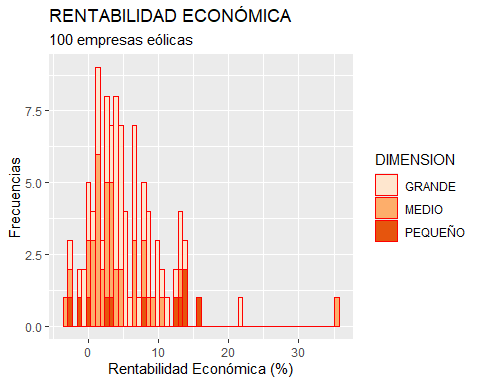
A continuación, vamos a modificar el color de las barras. Para el borde de estas, se utiliza el argumento **“colour =”**; y, para el relleno, **“fill =”**. Además, vamos a mejorar la presentación del gráfico añadiéndole un título y un subtítulo, y unas etiquetas en los ejes. Hay que prestar atención a los signos **“+”** incluidos para que R entienda que el código de la siguiente línea pertenece al mismo gráfico que estamos diseñando:

ggplot(data = eolica\_100, map = aes(x = RENECO)) +  
 geom\_histogram(bins = 40, colour = "red", fill = "orange") +  
 ggtitle("RENTABILIDAD ECONÓMICA", subtitle = "100 empresas eólicas")+  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Frecuencias")



Puede ser que nos interese diferenciar los casos según grupos preestablecidos. Por ejemplo, entre las variables de nuestro *data frame* “eolica\_100”, existe una variable categórica, atributo o factor, denominada DIMENSION, que clasifica a las 100 empresas en “GRANDE”, “MEDIO” o “PEQUEÑO” atendiendo al tamaño del grupo empresarial al que pertenecen (medido en número de empresas). Lo que vamos a hacer es crear, en el mismo gráfico, un histograma de la rentabilidad económica, pero para cada categoría de DIMENSION. Para ello, habrá que incluir esta variable categórica en el “mapeo”, en concreto mediante el argumento “fill =”. **Es** **necesario hacerlo dentro del elemento *aes***, ya que el resultado (color de grupo de empresas) depende del valor que toma la variable DIMENSION para cada caso o empresa:

ggplot(data = eolica\_100, map = aes(x = RENECO, fill = DIMENSION)) +  
 geom\_histogram(bins = 60, colour = "red") +  
 scale\_fill\_brewer(palette = "Oranges") +  
 ggtitle("RENTABILIDAD ECONÓMICA", subtitle = "100 empresas eólicas")+  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Frecuencias")

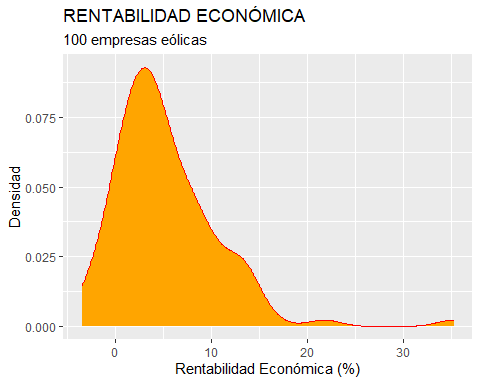


Como puede comprobarse, se superponen los tres histogramas, con tres colores diferentes, dependiendo de la dimensión considerada. Además, aparece, al lado derecho del gráfico, una leyenda que detalla qué color se asocia a cada uno de los grupos de empresas. La función scale\_fill\_brewer() nos permite personalizar la paleta de colores a utilizar (para ver las paletas disponibles, podemos consultar [esta sección](https://ggplot2-book.org/scales-colour) de ([Wickham 2021](#ref-ggplot22021)).

### 3.2.2 Gráfico de densidad.

Un gráfico parecido al histograma es el de **densidad**. Un gráfico de densidad estima la función de densidad de probabilidad empírica de la variable representada. En realidad, podemos considerarlo como un histograma “suavizado”. Probemos a ejecutar este código:

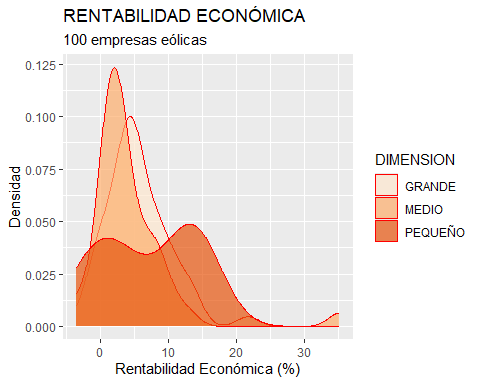
ggplot(data = eolica\_100, map = aes(x = RENECO)) +  
 geom\_density(colour = "red", fill = "orange") +  
 ggtitle("RENTABILIDAD ECONÓMICA", subtitle = "100 empresas eólicas")+  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Densidad")



En el código se observa la utilización del tipo de gráfico geom\_density(). Además, desaparece el número de intervalos o *bins*, y se puede dotar a la función de densidad estimada de un color en su borde (colour=), y de un color de relleno (fill=).

Como en casos anteriores, se puede crear una función de densidad estimada para cada grupo de empresas, eliminando las características, colour= y fill= del bloque del *geom*, y añadiéndolas en el “mapeo”, dentro del aes():

ggplot(data = eolica\_100, map = aes(x = RENECO, fill = DIMENSION)) +  
 geom\_density(colour = "red", alpha = 0.70, ) +  
 scale\_fill\_brewer(palette = "Oranges") +  
 ggtitle("RENTABILIDAD ECONÓMICA", subtitle = "100 empresas eólicas")+  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Densidad")



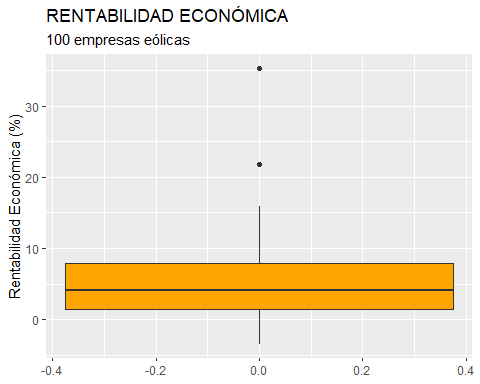
En efecto, el argumento fill= ha pasado a integrarse, en el “mapeo”, dentro de un elemento aes, ya que el color de relleno va a variar dependiendo del grupo de pertenencia de la empresa (variable DIMENSION). Por otro lado, en el geom se ha añadido el argumento alpha=. Esta información consiste en un número de 0 a 1 que gradúa el grado de transparencia / opacidad de los rellenos de las figuras (en este caso las funciones de densidad estimadas) incluidas en los gráficos.

### 3.2.3 Gráfico de caja o *Box-Plot*.

Un tipo muy interesante de gráfico es el de “caja” (***box-plot***), que informa de la dispersión de una variable. Fijémonos en el siguiente código:

ggplot(data = eolica\_100, map = (aes(y = RENECO))) +  
 geom\_boxplot(fill= "orange") +  
 ggtitle("RENTABILIDAD ECONÓMICA", subtitle = "100 empresas eólicas") +  
 ylab("Rentabilidad Económica (%)")

Puede observarse cómo en el “mapeo” se fija la variable que va a determinar las coordenadas del eje “y”. Como es una variable, hay que incluirla en el “mapeo” mediante una característica aes. El geom o tipo de gráfico es geom\_boxplot(), y en este caso no le hemos añadido ninguna característica específica. Las últimas líneas configuran los títulos del gráfico y del eje “y”. El resultado de ejecutar el código es el siguiente gráfico:



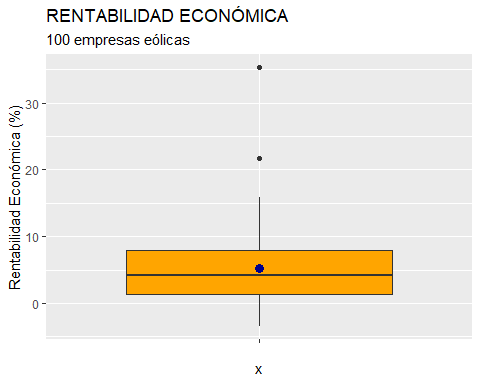
El gráfico se caracteriza por una “caja” (rectángulo) central. Esta caja está limitada por el primer y tercer cuartil, luego recoge el 50% de los casos con una rentabilidad económica superior al 25% de los casos con menor rentabilidad, y por debajo del 25% de los casos con la rentabilidad más alta. Así, la altura de la caja es la diferencia entre los cuartiles tercero y primero, que es lo que se denomina “rango intercuartílico” (*IQR* por las siglas en inglés). La caja, a su vez, está dividida en dos zonas por una línea horizontal, que es la mediana de la distribución: la rentabilidad económica que divide a los casos en dos grupos con el mismo número de casos, uno con los casos de mayor rentabilidad, y otro con los casos de menor rentabilidad.

Por encima y por debajo de la caja se disponen dos segmentos (llamados “bigotes”). Estos “bigotes” recogen los casos con valores en la variable inferiores al primer cuartil (comenzando por la base de la caja, hacia abajo), o superiores al tercer cuartil (comenzando por el techo de la caja, hacia arriba); y que están a menos de **1.5 veces** la altura de la caja. Los casos con valores de rentabilidad inferiores al primer cuartil (por abajo) y superiores al tercero (por arriba), que están alejados de la caja en más de 1.5 veces la altura de esta, se indican con puntos, y se corresponden con los casos conocidos como **casos atípicos** o ***outliers***. La identificación de los *outliers* es una fase muy importante a la hora de aplicar algunas técnicas estadísticas.

En esta práctica, comprobamos cómo, en el caso de la rentabilidad económica (RENECO), existen dos *outliers*, es decir, dos casos que presentan sendas rentabilidades anormalmente elevadas (más de un 20%).

{ggplot2} permite **integrar en el gráfico medidas estadísticas** y otros cálculos. Por ejemplo, en el box-plot se representa el valor de la mediana; pero no el de la media. Si queremos incluir el valor de la media (u otro estadístico), podemos calcularlo e integrarlo con la función stat\_summary(), algo parecido al summarise() de {dplyr}. El argumento fun = "mean" indica que la medida a calcular y representar es la media aritmética, el argumento geom = "point" el tipo de gráfico para representar esa medida (un punto). También hay otros argumentos opcionales. Para que funcione correctamente el código del gráfico, en el “mapeo” de la función ggplot() hay que añadir, dentro del aes(), x = “” (si no se hace, la ejecución dará un error en el que advierte de que falta el “*aesthetics: x*”):

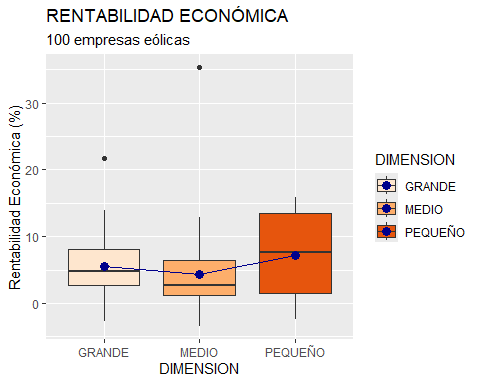
ggplot(data = eolica\_100, map = (aes(x = "", y = RENECO))) +  
 geom\_boxplot(fill = "orange") +  
 stat\_summary(fun = "mean",  
 geom = "point",  
 size = 3,  
 col = "darkblue") +  
 ggtitle("RENTABILIDAD ECONÓMICA", subtitle = "100 empresas eólicas") +  
 ylab("Rentabilidad Económica (%)")



Se aprecia cómo el valor de la *rentabilidad económica media* se ha insertado como un punto azul oscuro grueso dentro del gráfico de caja (tiene un valor algo superior a la *mediana*).

Vamos a refinar el *box-plot* anterior. Por ejemplo, quizá nos pueda interesar crear un *box-plot* para cada grupo de empresas, según el tamaño del grupo empresarial de pertenencia (atributo DIMENSION). Esto lo conseguiremos con el código:

ggplot(data = eolica\_100, map = (aes(x = DIMENSION, y = RENECO, fill = DIMENSION))) +  
 geom\_boxplot() +  
 stat\_summary(fun = "mean",  
 geom = "point",  
 size = 3,  
 col = "darkblue") +  
 stat\_summary(fun = "mean",  
 geom = "line",  
 col = "darkblue",  
 map = (aes(group = TRUE))) +  
 scale\_fill\_brewer(palette = "Oranges") +  
 ggtitle("RENTABILIDAD ECONÓMICA", subtitle = "100 empresas eólicas") +  
 ylab("Rentabilidad Económica (%)")

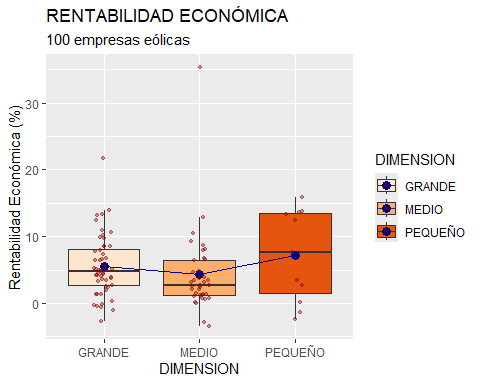


Para construir una caja por categoría de la variable cualitativa o atributo DIMENSION, se ha incluido, en el “mapeo” de la primera línea, el eje x con la variable tal variable DIMENSION. Como, además, queremos que cada caja sea de un color diferente, hemos hecho que los colores de estas dependan de la variable DIMENSION; añadiendo en el aes() del “mapeo” la característica fill= (que se refiere al color de relleno de las cajas). También se ha incluido una línea con el scale\_fill\_brewer() para que los colores de las cajas consistan en diferentes tonalidades de naranjas.

En el ejemplo, el primer bloque de stat\_summary() consigue puntear, para cada grupo de empresas, la media de RENECO en dicho grupo, en color azul oscuro. Para comparar mejor estas medias, se ha procedido a unir los puntos con unos segmentos o líneas de color azul oscuro, lo que se consigue con el segundo bloque de stat\_summary(). La última línea de ese bloque, map = (aes(group = TRUE))), obliga a que las líneas vayan de una media a otra de los grupos (de punto azul oscuro a punto azul oscuro).

Como última extensión, se ha considerado que, a veces, es conveniente tener en cuenta la posición de cada caso individual dentro del gráfico. Una opción es utilizar una capa o bloque geom\_jitter(). Con este geom se dispondrán, para cada grupo, los valores individuales de la variable RENECO; y para que estos, en su caso, no se solapen, se situarán un poco más a la izquierda o a la derecha, de modo aleatorio. Como los *outliers* son ya casos individuales, para que no se dupliquen con los provenientes del “jitter”, se indicará en el geom\_boxplot() que, en ese bloque gráfico, no se señalen los *outliers*. Esto se conseguirá con el argumento outlier.shape = NA. El código, en definitiva, será:

ggplot(data = eolica\_100, map = (aes(x = DIMENSION, y = RENECO, fill = DIMENSION))) +  
 geom\_boxplot(outlier.shape = NA) +  
 stat\_summary(fun = "mean",  
 geom = "point",  
 size = 3,  
 col = "darkblue") +  
 stat\_summary(fun = "mean",  
 geom = "line",  
 col = "darkblue",  
 map = (aes(group = TRUE))) +  
 geom\_jitter(width = 0.1,  
 size = 1,  
 col = "darkred",  
 alpha = 0.40) +  
 scale\_fill\_brewer(palette = "Oranges") +  
 ggtitle("RENTABILIDAD ECONÓMICA", subtitle = "100 empresas eólicas") +  
 ylab("Rentabilidad Económica (%)")



Como puede observarse, el geom\_jitter() proporciona, en cada caja, la nube de casos (empresas) individuales, en cuanto a la rentabilidad económica (incluidos los *outliers*). Las características de estos puntos (amplitud del desplazamiento lateral “aleatorio”, tamaño, color, opacidad) se controlan con diversos argumentos (width=, size=, col=, alpha=).

## 3.3 Gráficos de dos variables.

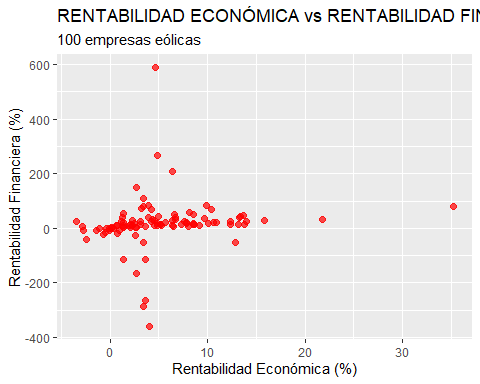
### 3.3.1 Gráfico de dispersión o *scatterplot*.

Pasamos ahora a comentar un tipo de gráfico muy común cuando trabajamos con dos variables métricas: los **gráficos de dispersión** (o ***scatterplots***). En este tipo de gráficos, cada variable ocupa un eje (x o y), y los puntos internos al gráfico representan los diversos casos u observaciones.

Como ejemplo, vamos a crear un gráfico de dispersión que represente las empresas eólicas en función de su rentabilidad económica (RENECO) y de su rentabilidad financiera (RENFIN). El código es el siguiente:

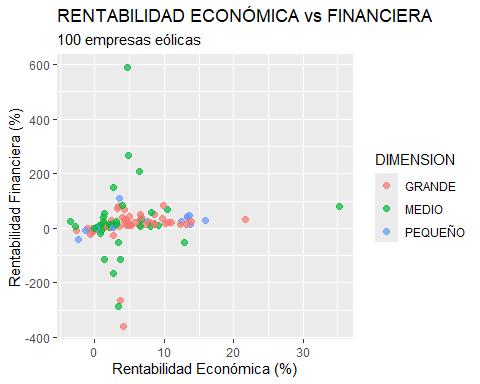
ggplot(data = eolica\_100, map = (aes(x = RENECO, y = RENFIN))) +  
 geom\_point(color = "red", size = 2, alpha = 0.7) +  
 ggtitle("RENTABILIDAD ECONÓMICA vs RENTABILIDAD FINANCIERA", subtitle = "100 empresas eólicas") +  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Rentabilidad Financiera (%)")

El resultado es el siguiente gráfico:



Vamos a refinar el gráfico algo más. En primer lugar, puede ser interesante **distinguir entre los tipos de empresas**, según el tamaño del grupo empresarial al que pertenecen ( variable DIMENSION). Para ello, podemos poner el color de los puntos en el “mapeo”, en función de la variable DIMENSION:

ggplot(data = eolica\_100, map = (aes(x = RENECO,  
 y = RENFIN,  
 col = DIMENSION))) +  
 geom\_point(size = 2, alpha = 0.7) +  
 ggtitle("RENTABILIDAD ECONÓMICA vs FINANCIERA",  
 subtitle = "100 empresas eólicas") +  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Rentabilidad Financiera (%)")

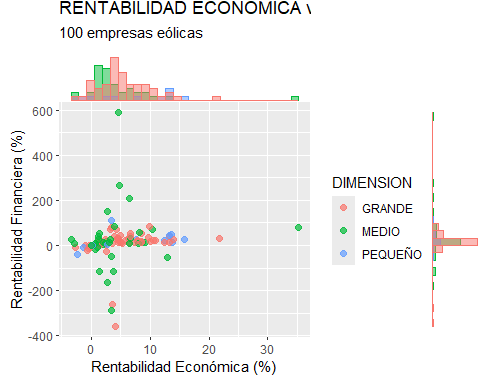


En los dos gráficos anteriores pueden observarse puntos (casos) candidatos a ser *outliers* para cada una de las dos variables analizadas. En el caso de RENECO, ya se pudo advertir esta circunstancia al construir los *boxplots*.

Por otro lado, podría ser interesante complementar el gráfico con información sobre las dos variables por separado, es decir, con **información sobre las distribuciones marginales**. Existe un paquete complementario a {ggplot2}, llamado {ggExtra}, que puede ayudar fácilmente a este cometido. Para ello, hemos de activar dicho paquete con library() (si no ha sido previamente instalado, habrá que hacerlo con anterioridad). El segundo paso consistirá en **asignar** nuestro *scatterplot*, diseñado con la función ggplot(), a un objeto con el nombre que queramos, por ejemplo, “scatter\_plus”. Luego, ese objeto, que contiene nuestro gráfico, entrará como argumento en la función de {ggExtra} llamada ggMarginal(), como se muestra en el siguiente código:

library ("ggExtra")  
scatter\_plus <- ggplot(data = eolica\_100, map = (aes(x = RENECO,  
 y = RENFIN,  
 col = DIMENSION))) +  
 geom\_point(size = 2, alpha = 0.7) +  
 ggtitle("RENTABILIDAD ECONÓMICA vs FINANCIERA",  
 subtitle = "100 empresas eólicas") +  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Rentabilidad Financiera (%)")  
ggMarginal(scatter\_plus, type = "histogram", groupColour = T,  
 groupFill = T, position = "identity", alpha = 0.5)

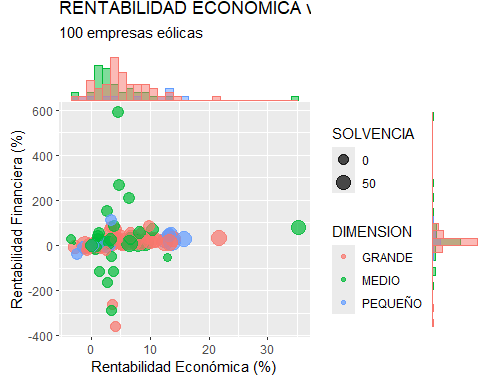
Con el código anterior, apreciamos cómo se añaden los histogramas de cada variable, RENECO y RENFIN, en los márgenes del gráfico:



Conviene apuntar que el argumento position = “identity” hace que las barras del histograma estén perfectamente alineadas con los datos del gráfico de dispersión, sin ningún tipo de desplazamiento.

Adicionalmente, los diámetros de los puntos de los diversos casos podrían contener también información, haciéndolos proporcionales a una tercera variable. Por ejemplo, podrían ser proporcionales al nivel de solvencia (variable SOLVENCIA). Para ello, ejecutaríamos el código:

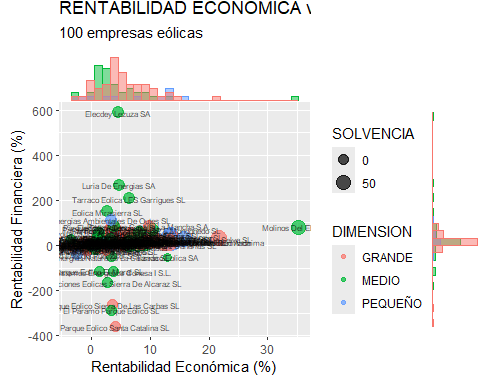
scatter\_plus <- ggplot(data = eolica\_100, map = (aes(x = RENECO,  
 y = RENFIN,  
 col = DIMENSION,  
 size = SOLVENCIA))) +  
 geom\_point(alpha = 0.7) +  
 ggtitle("RENTABILIDAD ECONÓMICA vs FINANCIERA",  
 subtitle = "100 empresas eólicas") +  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Rentabilidad Financiera (%)")  
ggMarginal(scatter\_plus, type = "histogram", groupColour = T,  
 groupFill = T, position = "identity", alpha = 0.5)



En el código anterior, puede comprobarse que la característica size = sube del bloque de geom al “mapeo” (incluido en el aes), debido a que el diámetro de cada punto ya no va a ser un parámetro fijo, sino que va a depender de la magnitud de la variable SOLVENCIA.

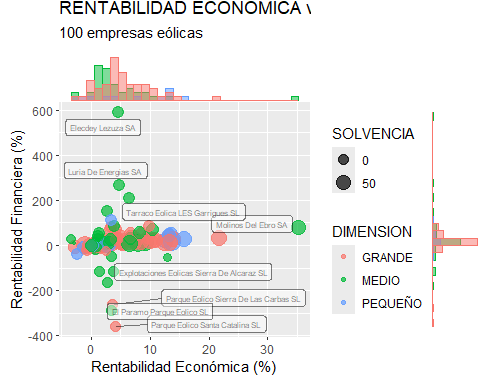
Finalmente, podría ser útil, en algunos gráficos, añadir una etiqueta (*label*) a cada punto, para **identificar el caso concreto** al que representa. Si bien en esta práctica, el elevado número de casos y el extenso nombre de las empresas hacen poco claro el uso de estas etiquetas, vamos a añadirlas por motivos pedagógicos. Para ello, se añadirá un bloque geom llamado geom\_text(), con una información label = que se hace depender de valores que cambian (en concreto, el nombre de los casos, es decir, de las filas del *data frame*), por lo que tendrá que integrarse en una característica aes:

scatter\_plus <- ggplot(data = eolica\_100, map = (aes(x = RENECO,  
 y = RENFIN,  
 col = DIMENSION,  
 size = SOLVENCIA))) +  
 geom\_point(alpha = 0.7) +  
 geom\_text(aes(label=row.names(eolica\_100)), size=2, color="black", alpha = 0.7) +  
 ggtitle("RENTABILIDAD ECONÓMICA vs FINANCIERA",  
 subtitle = "100 empresas eólicas") +  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Rentabilidad Financiera (%)")  
ggMarginal(scatter\_plus, type = "histogram", groupColour = T,  
 groupFill = T, position = "identity", alpha = 0.5)



Las etiquetas de los casos pueden refinarse algo más mediante la función geom\_label\_repel(), disponible al cargar el paquete {ggrepel}:

library(ggrepel)  
scatter\_plus <- ggplot(data = eolica\_100, map = (aes(x = RENECO,  
 y = RENFIN,  
 col = DIMENSION,  
 size = SOLVENCIA))) +  
 geom\_point(alpha = 0.7) +  
 geom\_label\_repel(aes(label = row.names(eolica\_100)),  
 size = 2,  
 color = "black",  
 alpha = 0.5) +  
 ggtitle("RENTABILIDAD ECONÓMICA vs FINANCIERA",  
 subtitle = "100 empresas eólicas") +  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Rentabilidad Financiera (%)")  
ggMarginal(scatter\_plus, type = "histogram", groupColour = T,  
 groupFill = T,  
 position = "identity", alpha = 0.5)



La ventaja de este gráfico, como se puede apreciar, es que se omiten las etiquetas superpuestas, si bien existe el riesgo de que se omitan una gran cantidad de estas.

## 3.4 Materiales para realizar las prácticas del capítulo.

En esta sección se muestran los links de acceso a los diferentes materiales (scripts, datos…) necesarios para llevar a cabo los contenidos prácticos del capítulo.

**Datos (en formato Microsoft (R) Excel (R)):**

* eolica\_100.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1Up5ZWR57rgIkmYJcr2Pek6kUdG_yd4vu/edit?usp=drive_link&ouid=115375878280465826079&rtpof=true&sd=true))

**Scripts:**

* explora\_ggplot2.R ([obtener aquí](https://drive.google.com/file/d/1R0qYnrjcel4SOH9lVI1_qkccKYXBLLDC/view?usp=sharing))

# 4 Estadística descriptiva.

La Estadística Descriptiva es la parte de la Ciencia Estadística que se ocupa de la recopilación de datos, su depuración, y la caracterización mediante dichos datos de un conjunto de casos o individuos.

Los datos se organizan en variables y/o atributos.

Las **variables** son características de los casos o individuos en estudio que se plasman en valores que están expresados en **escala métrica**. Los **atributos** son características de los casos o individuos en estudio que se concretan en diversas categorías (si el atributo tiene **escala nominal**) o niveles (si el atributo tiene **escala ordinal**). Los atributos se denominan también variables cualitativas o factores.

Centrándonos en las variables (características que afectan a un grupo de casos o individuos, y que se concretan en valores que poseen una escala métrica), podemos plantearnos el estudio de una única variable sin tener en cuenta la existencia de otras variables que caracterizan al mismo grupo de casos o individuos. En tal caso estaremos planteando un **análisis estadístico univariante**. Si nuestro análisis se centra en cómo dos variables caracterizan al mismo conjunto de individuos o casos, y la posible relación entre ambas, estaremos planteando un análisis bivariante. Generalizando, si estudiamos cómo un grupo de variables caracterizan de modo conjunto a un mismo grupo de casos o individuos, estaremos planteando un **análisis estadístico multivariante**.

## 4.1 Análisis univariante.

En el análisis estadístico univariante, estudiamos cómo una única característica (nos centraremos en una variable, aunque también puede tratarse de un atributo) afecta a un grupo de casos, individuos o elementos. Por ejemplo, la variable podría ser el salario percibido por un grupo de individuos que podría ser el conjunto de trabajadores en nómina en una empresa. Otro ejemplo podría ser el de la (variable) rentabilidad económica obtenida por un grupo de empresas pertenecientes a un determinado sector económico.

El conjunto de pares formado por cada valor que puede tomar la variable en estudio (o categoría o nivel, en el caso de un atributo) y el número de casos que toman tal valor se denomina **distribución de frecuencias** de la variable.

¿Cómo podemos estudiar el modo en que afecta una variable, de modo global, a un grupo de casos? Mediante el cálculo de una serie de **medidas**. Las medidas son instrumentos matemáticos que extraen y sintetizan la información contenida en una distribución de frecuencias.

Hay diferentes tipos de medidas, principalmente las de **posición**, **dispersión** y **forma**.

Antes de profundizar en las principales medidas, su significado y su obtención; vamos a indicar cómo se pueden presentar en R, mediante la creación de tablas, los datos referentes a un grupo de individuos y las variables o atributos que los caracterizan, y las distribuciones de frecuencias univariantes.

## 4.2 Representando datos y distribuciones de frecuencias en tablas con R.

Para aprender a representar los datos referentes a las variables y atributos que caracterizan a un grupo de casos o individuos, y las distribuciones de frecuencias univariantes, vamos a suponer que trabajamos dentro de un proyecto que hemos creado previamente, de nombre “explora”. Dentro de la carpeta del proyecto guardaremos el *script* llamado “[explora\_trabajadores.R](https://drive.google.com/file/d/1pfyGPGQRmgOyo9wrZ7fr_KAFj49ecXID/view?usp=sharing)” y el archivo de *Microsoft® Excel®* llamado “[trabajadores.xlsx](https://docs.google.com/spreadsheets/d/1e_h8P_0sBuD5Nvw6nkeWtXXeiqItFfdI/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true)”.

Las primeras líneas del script se refieren, como ya hemos visto en otras secciones del libro, a la limpieza de la memoria o environment, eliminando objetos que se hayan podido crear con anterioridad, y en la importación de los datos que hay en la hoja “Datos” de “trabajadores.xlsx”. Estos datos se almacenan en el *data frame* “datos”, y consisten en el registro del salario (variable SALARIO, expresada en cientos de euros), el nivel de estudios (atributo NESTUDIOS) y departamento al que se pertenece (atributo DEP), correspondientes a los 49 trabajadores de una determinada empresa:

# Script para la construcción de tablas de datos   
# y trabajo con distribucionesde frecuencias univariantes.  
#  
  
rm(list = ls())  
  
## DATOS  
  
# Importando  
  
library(readxl)  
datos <- read\_excel("trabajadores.xlsx", sheet = "Datos")

Como sabemos, R interpreta el nombre de los trabajadores como una variable más, en lugar de como la identificador de cada “fila” o caso. Para corregir esto, y hacer saber a R que la primera columna no es una variable, sino el nombre de cada fila (caso, en este caso trabajador), añadimos la línea:

datos <- data.frame(datos, row.names = 1)

Posteriormente, podremos comprobar que “datos” contiene el valor del salario, el nivel de estudios y la categoría de departamento para cada uno de los 49 trabajadores de la empresa:

summary (datos)

## SALARIO NESTUDIOS DEP   
## Min. : 8.00 Length:49 Length:49   
## 1st Qu.:12.00 Class :character Class :character   
## Median :15.00 Mode :character Mode :character   
## Mean :16.04   
## 3rd Qu.:20.00   
## Max. :30.00

Sabemos que, simplemente escribiendo el nombre del *data frame*, aparecerán en la consola los datos almacenados en él. No obstante, esta presentación no es muy elegante para presentar los datos. Vamos a presentar tales datos de un modo más amigable, mediante la **confección de una “tabla”**.

Un paquete de R muy popular para generar tablas de datos es {knitr}. Este paquete contiene la función kable(), que permite generar tablas en varios formatos y con diversas características que pueden ser personalizadas (como el título de la tabla). Si queremos personalizar más aún la apariencia de nuestras tablas, podemos usar las facilidades del paquete {kableExtra}, que complementa las posibilidades que ofrece la función kable() de {knitr}.

Para hacer una tabla con nuestros casos y variables, es decir, para escribir nuestro *data frame* “datos” de un modo más elegante, primero activaremos los paquetes anteriores, y añadiremos una línea donde diremos el formato de la tabla a generar (en nuestro ejemplo, formato .html), todo con el siguiente código:

# Tabla de datos  
  
library (knitr)  
library (kableExtra)

Después, generaremos nuestra tabla con los datos contenidos en el *data frame* “datos”. El código para generar la tabla, y el resultado, es el siguiente:

knitr.table.format = "html"  
 datos %>%  
 kable(caption = "Trabajadores asalariados de la empresa",  
 col.names = c("Trabajador", "Salario", "Nivel de estudios",  
 "Departamento")) %>%  
 kable\_styling(full\_width = F, bootstrap\_options = "striped", "bordered",  
 "condensed", position = "center", font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:(nrow(datos)), bold= F, align = "c")

(#tab:unnamed-chunk-113)(#tab:unnamed-chunk-113)Trabajadores asalariados de la empresa

Trabajador

Salario

Nivel de estudios

Departamento

Andrés

8

Básicos

Almacén

Ángela

8

Medios

Almacén

Miguel

9

Básicos

Almacén

Luis

9

Básicos

Almacén

María

9

Medios

Gestión Interna

Lourdes

10

Medios

Gestión Interna

Carlos

10

Medios

Gestión Interna

Adriana

10

Medios

Gestión Interna

Ricardo

10

Medios

Gestión Proveedores

Juan José

10

Medios

Gestión Proveedores

Daniel

10

Medios

Gestión Proveedores

Pedro

12

Básicos

Almacén

Ana

12

Básicos

Almacén

Isabel

12

Básicos

Almacén

Manuel

12

Medios

Gestión Proveedores

Isidro

12

Medios

Gestión Proveedores

Carla

12

Medios

Gestión Proveedores

Fernando

12

Universitarios

Gestión Clientes

Belén

12

Universitarios

Marketing

Margarita

15

Básicos

Almacén

Andrea

15

Básicos

Almacén

Pablo

15

Medios

Gestión Interna

Celia

15

Medios

Gestión Interna

Alba

15

Medios

Gestión Interna

Nicolás

15

Medios

Gestión Proveedores

David

15

Medios

Gestión Proveedores

Elena

15

Medios

Gestión Proveedores

Victoria

15

Medios

Gestión Proveedores

Antonio

15

Universitarios

Gestión Clientes

Tomás

15

Universitarios

Gestión Clientes

Bartolomé

20

Universitarios

Almacén

Irene

20

Universitarios

Gestión Proveedores

Guadalupe

20

Universitarios

Gestión Proveedores

Ignacio

20

Universitarios

Gestión Proveedores

Ernesto

20

Universitarios

Marketing

Abel

20

Universitarios

Gestión Clientes

Nieves

20

Universitarios

Gestión Clientes

Carolina

20

Universitarios

Gestión Clientes

Luisa

22

Universitarios

Gestión Clientes

Alberto

22

Universitarios

Marketing

Paula

22

Universitarios

Almacén

Sergio

22

Universitarios

Gestión Proveedores

Estrella

22

Universitarios

Gestión Interna

Alicia

22

Universitarios

Coordinación

Marta

25

Universitarios

Dirección

Alfonso

25

Universitarios

Dirección

Mar

25

Universitarios

Dirección

Martín

25

Universitarios

Dirección

Blanca

30

Universitarios

Dirección

Primero llamamos al *data frame* a partir de cuyos datos vamos a generar la tabla, “datos”. Con el operador *pipe* %>%, ligamos los datos del *data frame* al diseño la tabla realizado con la función kable() de {knitr}. kable() tiene diversos argumentos, entre los que destacan:

* caption =: Este argumento informa del título de la tabla.
* col.names =: Este argumento, opcional, fija el nombre para las columnas de la tabla, si no queremos que aparezcan los nombres “por defecto”, que son los nombres de cada columna en el propio *data frame*.

Luego, con el operador *pipe* %>% informamos de que vamos a completar o personalizar el diseño de esta tabla con otras funciones complementarias del paquete {kableExtra}. En primer lugar, utilizamos la función kable\_styling(), que aporta algunas características adicionales a la tabla, según sus argumentos:

* full\_width = : este argumento ha de tener un valor lógico, y se refiere a si deseamos que la tabla ocupe todo el ancho del documento (TRUE) o solo lo necesario (FALSE).
* bootstrap\_options = : este argumento es de tipo alfanumérico, y sirve para fijar ciertas características estéticas complementarias. “striped” se refiere a que las filas aparezcan sombreadas de modo alternativo, “bordered” se refiere a que cada fila quede delimitada por unas finas líneas en la parte superior y en la inferior, “condensed” significa que la tabla tendrá un aspecto más compacto.
* position = : este argumento se utiliza para situar la tabla centrada, a la izquierda del párrafo, o a la derecha.
* font\_size = : este argumento numérico se refiere al tamaño de los caracteres, lo cuál es importante a la hora de que una tabla “quepa” en un documento de deteminada anchura.

Por último, hacemos uso dos veces de la función row\_spec() del paquete {kableExtra}. Esta función sirve para personalizar algo más las filas concretas de la tabla que consideremos. El encabezado se identifica como la fila “0”. En el ejemplo, se ha utilizado esta función dos veces: una para el encabezado (el primer argumento de la función nos informa de las filas a las que se refiere, en esta ocasión la fila 0), y otra para el resto de filas (desde la fila 1 hasta la que contiene al último caso individuo, la fila con posición nrow(datos)). Los otros argumentos definen si se quiere que los caracteres aparezcan en negrita (bold = ) y cómo deben estar alineados los elementos, dentro de las columnas ( align = ).

A veces, puede ocurrir que solo nos interese estudiar una variable (columna del data frame). Además, esm posible que el conjunto de casos sea muy numeroso, y que, adicionalmente, algunos de los valores de la variable que queremos estudiar estén repetidos para varios casos. Cuando esto ocurre, una opción interesante es, en lugar de representar en una tabla todos nuestros datos, **representar la distribución de frecuencias** de la variable (o atributo) que nos interesa. Es lo que vamos a hacer a continuación, tomando como variable a analizar la variable SALARIO.

Lo primero a tener en cuenta es que, en las distribuciones de frecuencias, **los valores de la variable suelen disponerse de menor a mayor**. Para ello, previamente vamos a ordenar las filas del *data frame* “datos” según el valor que toma, en el caso correspondiente, la variable SALARIO y, si existen casos con el mismo valor de SALARIO, los ordenaremos por orden alfabético del nombre del caso (nombre del trabajador, o de la fila del *data frame*). Para realizar este reordenamiento de casos (filas) del *data frame* de un modo sencillo, vamos a utilizar la función arrange() del paquete {deplyr}:

# Colocar los datos  
  
library(dplyr)  
datos <- datos %>% arrange(SALARIO, row.names(datos))

Una vez que los casos están ordenados en el data frame de menor a mayor valor de SALARIO, calcularemos, para cada valor del SALARIO, el número de casos que lo poseen, es decir, la frecuencia absoluta de cada valor de la variable SALARIO. Para ello, vamos a crear un objeto denominado “conteo” que va a ser de clase “tabla” de la variable SALARIO. Todo ello lo realizamos mediante la función table(), como se muestra a continuación:

conteo <- table(datos$SALARIO)  
conteo

##   
## 8 9 10 12 15 20 22 25 30   
## 2 3 6 8 11 8 6 4 1

Al mostrar en la consola el objeto “tabla” conteo, vemos cómo se compone de dos filas de datos. La primera se corresponde con los valores que toma la variable SALARIO en los distintos casos, y la segunda es el número de casos (**frecuencia absoluta**) que toma cada valor. Es decir, el objeto “tabla” es la distribución de frecuencias de la variable SALARIO.

Vamos a convertir este objeto “tabla” en un *data frame*, llamado “conteo\_df”, con el objeto de poder representar de un modo más elegante la distribución de frecuencias. Para ello, ejecutaremos el código:

# Convertir el resultado a un data frame para una mejor visualización  
  
conteo\_df <- as.data.frame(conteo)  
conteo\_df

## Var1 Freq  
## 1 8 2  
## 2 9 3  
## 3 10 6  
## 4 12 8  
## 5 15 11  
## 6 20 8  
## 7 22 6  
## 8 25 4  
## 9 30 1

Al mostrar en la consola el data frame “conteo\_df”, observamos que consta de dos columnas o variables. “Var1” recoge los valores que toma la variable SALARIO en el grupo de casos, y “Freq” es el conjunto de frecuencias absolutas de los diferentes valores. Para que se entienda mejor qué es cada columna, las renombraremos:

# Renombrar las columnas para mayor claridad  
  
colnames(conteo\_df) <- c("Valor", "Frecuencia")

A continuación, vamos a calcular el resto de frecuencias que suelen calcularse para una variable. La **frecuencia total**, N, que es la suma de todas las frecuencias absolutas, es decir, el número total de casos, se puede calcular fácilmente como:

# Calcular y guardar la frecuencia total  
  
N <- sum(conteo\_df$Frecuencia)

La serie de frecuencias absolutas acumuladas se calcularán del siguiente modo:

# Calcular frecuencias absolutas acumuladas  
  
conteo\_df$Frecuencia\_acum <- cumsum(conteo\_df$Frecuencia)

Como sabemos, la última frecuencia absoluta acumulada debe coincidir con la frecuencia total. Por último, calcularemos las frecuencias relativas, que son las frecuencias absolutas divididas por la frecuencia total, y recogen la proporción de casos correspondientes al valor de la variable:

# Calcular frecuencias relativas  
  
conteo\_df$Frecuencia\_R <- conteo\_df$Frecuencia / N  
  
# Calcular frecuencias relativas acumuladas  
  
conteo\_df$Frecuencia\_R\_acum <- cumsum(conteo\_df$Frecuencia\_R)

La suma de las frecuencias relativas es siempre 1 (el 100% de los casos). Además, la última frecuencia relativa acumulada siempre es, igualmente, 1.

Ahora vamos a construir una tabla que recoja la distribución de frecuencias de la variable SALARIO (con los diversos tipos de frecuencias). Para ello, simplemente hemos de aplicar al data frame “conteo\_df” las funciones kable() del paquete {knitr}, y el resto de funciones auxiliares del paquete {kableExtra}:

conteo\_df %>%  
 kable(caption = "Distribución de frecuencias de los salarios de la empresa",  
 col.names = c("x(i) = Salario", "Frecuencia absoluta n(i)",  
 "Frecuencia absoluta acum. N(i)", "Frecuencia relativa f(i)",  
 "Frecuencia relativa acum. F(i)"),  
 format.args = list(decimal.mark = ".", digits = 2)) %>%  
 kable\_styling(full\_width = F, bootstrap\_options = "striped",  
 "bordered", "condensed",  
 position = "center", font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:(nrow(conteo\_df)), bold= F, align = "c")

(#tab:unnamed-chunk-122)(#tab:unnamed-chunk-122)Distribución de frecuencias de los salarios de la empresa

x(i) = Salario

Frecuencia absoluta n(i)

Frecuencia absoluta acum. N(i)

Frecuencia relativa f(i)

Frecuencia relativa acum. F(i)

8

2

2

0.04

0.04

9

3

5

0.06

0.10

10

6

11

0.12

0.22

12

8

19

0.16

0.39

15

11

30

0.22

0.61

20

8

38

0.16

0.78

22

6

44

0.12

0.90

25

4

48

0.08

0.98

30

1

49

0.02

1.00

Hemos de advertir que en la función kable() se ha insertado un nuevo argumento, format.args = , que es una “lista” que controla aspectos de formato como si los decimales se indican con un punto o una coma, o el número de decimales a mostrar en la tabla.

Hay ocasiones en las que la cantidad de valores diferentes que toma la variable analizada para los diferentes casos es muy elevado. Esto puede deberse, por ejemplo, a que el número de casos es muy elevado, o a que la variable es de naturaleza continua, y puede tomar una gran variedad de posibles valores (incluso infinitos). En estos casos, un modo de representar la distribución de frecuencias de la variable en una tabla de dimensión reducida es **agrupando los valores en intervalos**. Esto es lo que vamos a hacer ahora con la variable SALARIO.

La primera tarea a realizar será formar los intervalos. Para ello podemos usar la función cut(), que permite decir el número de intervalos (de la misma amplitud) en que queremos dividir el intervalo que va desde el menor valor de la distribución (menor salario) al mayor valor (mayor salario). Por ejemplo, si deseamos agrupar los valores en 4 intervalos, el código será:

# Distribución de frecuencias agrupadas en intervalos  
# del salario de los trabajadores de la empresa.  
  
# Crear los intervalos  
datos$intervalos <- cut(datos$SALARIO, breaks = 4, include.lowest = TRUE)

El resultado del código anterior es una nueva columna en el *data frame* “datos”, llamada “intervalos”, que informa, para cada caso, cuál de los 4 intervalos calculados lo contiene. El argumento lógico include.lowest = se especifica para indicar que el intervalo inferior es cerrados por la izquierda. Lo usual es que, salvo este, el resto sean abiertos, es decir, que los casos que toman como valor de la variable un extremo de intervalo se contabilicen dentro del intervalo donde ese valor es el extremo superior.

La columna “intervalos” es de la clase “factor”. Precisamente, los posibles “niveles” de ese factor son los 4 intervalos que se han creado con cut():

# Obtener los niveles de los intervalos  
  
levels(datos$intervalos)

## [1] "[7.98,13.5]" "(13.5,19]" "(19,24.5]" "(24.5,30]"

Es preciso advertir que el vector “limites” contiene elementos de clase caracter (aunque contengan cifras, ya que también contienen corchetes, paréntesis y comas).

Las siguientes líneas de código son similares a las que vimos en el caso de distribuciones de frecuencias no agrupadas: se creará un objeto “tabla” para contabilizar el número de casos que pertenecen a cada intervalo (frecuencias absolutas), se transformará este objeto en un *data frame* para poder trabajar de un modo más fácil, y se cambiarán el nombre de las dos columnas para que se entienda mejor:

# Contar las frecuencias de cada intervalo  
  
conteo\_intervalos <- table(datos$intervalos)  
  
# Convertir el resultado a un data frame para una mejor visualización  
  
conteo\_intervalos\_df <- as.data.frame(conteo\_intervalos)  
  
# Renombrar las columnas para mayor claridad  
  
colnames(conteo\_intervalos\_df) <- c("Intervalo", "Frecuencia")

Con todo lo anterior, se obtiene un *data frame* denominado “conteo\_intervalos\_df”, que contiene dos columnas: la columna “Intervalo”, con los 4 intervalos calculados, y la columna “Frecuencia”, con el número de casos que tienen un salario incluido dentro de cada intervalo salarial.

Antes de proceder a diseñar la tabla de presentación de la distribución de frecuencia con kable(), vamos a obtener, para incluir en la tabla, otras informaciones que suelen ser presentadas junto a las frecuencias absolutas de cada intervalo.

Una de estas informaciones es lo que denominamos “**marca de clase**” de un intervalo. La marca de clase de un intervalo de valores es simplemente el punto medio de dicho intervalo.

La obtención en nuestro ejemplo de las marcas de clase puede resultar algo compleja, ya que hemos de recordar que los intervalos, tal y como están almacenados, son los niveles de una variable de clase “factor”:

marca\_clase <- sapply(strsplit(as.character(conteo\_intervalos\_df$Intervalo), ",|\\[|\\(|\\]"), function(x) {  
 mean(as.numeric(x[2:3]))  
})

Explicaremos detenidamente el código anterior:

1. **conteo\_intervalos\_df$Intervalo**: Aquí se está accediendo a la columna “Intervalo” del *data frame* “conteo\_intervalos\_df”.
2. **as.character(conteo\_intervalos\_df$Intervalo)**: convierte los valores de la columna “Intervalo” a caracteres (*strings*). Esto es necesario porque la función strsplit() trabaja con cadenas de texto.
3. **strsplit(as.character(conteo\_intervalos\_df$Intervalo), ",|\\[|\\(|\\]")**: strsplit() divide cada cadena de texto en partes usando los delimitadores especificados. En este caso, se están utilizando como delimitadores las comas “,”, los corchetes “[” y ”]”, y el paréntesis de apertura “(”. El resultado es una lista de vectores de caracteres, donde cada vector contiene las partes de la cadena original que estaban separadas por los delimitadores.
4. **sapply(..., function(x) { ... })**: sapply() aplica una función a cada elemento de una lista y simplifica el resultado a un vector o matriz. Por otro lado, la función anónima function(x) { ... } se aplica a cada vector resultante de strsplit().
5. **function(x) { mean(as.numeric(x[2:3])) }**: Esta es la función anónima que se aplica a cada vector “x”. Después, **x[2:3]** selecciona el segundo y tercer elemento del vector “x”. Estos elementos corresponden con los límites del intervalo. **as.numeric(x[2:3])** convierte estos elementos a números. **mean(as.numeric(x[2:3]))** calcula la media de estos dos números, que representa el punto medio del intervalo.
6. **marca\_clase <- ...**: Finalmente, el resultado de sapply() se asigna al vector”marca\_clase”, que contendrá los puntos medios de los 4 intervalos.

El resto de código integra el vector “marca\_clase” en el *data frame* “conteo\_intervalo\_df” como una variable más, reodena con la función select() del paquete {dplyr} el orden de las columnas del *data frame*, calcula el resto de frecuencias (absoluta acumulada, relativa, relativa acumulada), y diseña la tabla de presentación de la distribución de frecuencias de los salarios de los trabajadores de la empresa; pero agrupada en 4 intervalos de valores.

# Agregar la columna "marca\_clase" al data frame  
  
conteo\_intervalos\_df$marca\_clase <- marca\_clase  
  
#Cambiar el orden de las columnas en el data frame con dplyr  
  
conteo\_intervalos\_df <- conteo\_intervalos\_df %>% select(Intervalo, marca\_clase, Frecuencia)  
  
# Calcular y guardar la frecuencia total  
  
N\_agre <- sum(conteo\_intervalos\_df$Frecuencia)  
  
# Calcular frecuencias absolutas acumuladas  
  
conteo\_intervalos\_df$Frecuencia\_acum <- cumsum(conteo\_intervalos\_df$Frecuencia)  
  
# Calcular frecuencias relativas  
  
conteo\_intervalos\_df$Frecuencia\_R <- conteo\_intervalos\_df$Frecuencia / N\_agre  
  
# Calcular frecuencias relativas acumuladas  
  
conteo\_intervalos\_df$Frecuencia\_R\_acum <- cumsum(conteo\_intervalos\_df$Frecuencia\_R)  
  
# Mostrar el resultado  
  
conteo\_intervalos\_df %>%  
 kable(caption = "Distribución de frecuencias agrupadas en intervalos de los salarios de la empresa",  
 col.names = c("Intervalo salarial", "Marca de clase x(i)", "Frecuencia absoluta n(i)",  
 "Frecuencia absoluta acum. N(i)", "Frecuencia relativa f(i)",  
 "Frecuencia relativa acum. F(i)"),  
 format.args = list(decimal.mark = ".", digits = 2)) %>%  
 kable\_styling(full\_width = F, bootstrap\_options = "striped",  
 "bordered", "condensed",  
 position = "center", font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:(nrow(conteo\_intervalos\_df)), bold= F, align = "c")

(#tab:unnamed-chunk-129)(#tab:unnamed-chunk-129)Distribución de frecuencias agrupadas en intervalos de los salarios de la empresa

Intervalo salarial

Marca de clase x(i)

Frecuencia absoluta n(i)

Frecuencia absoluta acum. N(i)

Frecuencia relativa f(i)

Frecuencia relativa acum. F(i)

[7.98,13.5]

11

19

19

0.39

0.39

(13.5,19]

16

11

30

0.22

0.61

(19,24.5]

22

14

44

0.29

0.90

(24.5,30]

27

5

49

0.10

1.00

## 4.3 Medidas de posición.

Las medidas de posición son instrumentos matemáticos que pretenden, mediante un único valor o muy pocos valores, **caracterizar de modo global** la distribución de frecuencias de una variable determinada.

Las medidas de posición se pueden clasificar en medidas de posición central, y en medidas de posición no central (principalmente, los llamados cuantiles).

Las principales **medidas de posición central** son: la media, la mediana y la moda. Dentro de la media, podemos distinguir la media aritmética, la geométrica y la armónica. De ellas, nos centraremos en la más común: la media aritmética.

La **media aritmética** de la distribución de frecuencias de una variable X se calcula como:

Hemos de tener en cuenta en la fórmula anterior que N es la frecuencia total, y h es el número de valores diferentes que toma la variable.

Si las frecuencias de todos los valores de la variable son 1 (distribución de frecuencias unitarias), lógicamente la media pasará a ser:

En R, la función para obtener la media de una variable es mean(). Así, para obtener el salario medio de la variable SALARIO de los trabajadoes de la empresa, ejecutaremos el código:

## MEDIDAS  
  
# Media aritmética.  
  
media <- mean(datos$SALARIO)  
media

## [1] 16.04082

Como podemos observar, el salario medio de los trabajadores de la empresa, recogido en el valor “media”, es de 16.04 cientos de euros, es decir, 1604 euros.

¿Qué significado tiene la media aritmética? La media aritmética es el “centro de gravedad” de la distribución, el punto de equilibrio, en el sentido de que, si todos los trabajadores ganaran el salario medio, no habría diferencias salariales aun cuando la “masa” salarial invertida por la empresa permanecería invariable. Es decir, la media aritmética supone un reparto igualitario de la masa total de la variable.

Entre sus ventajas destaca el que, para variables (escala métrica) es siempre calculable y única. Como inconvenientes, que pierde su representatividad ante la existencia de casos atípicos o *outliers*, y que no se puede calcular en el caso de trabajar con atributos, variables cualitativas o factores (escalas nominal u ordinal).

La **mediana** es el valor que se corresponde con el caso o casos que dividen a la distribución en dos grupos con el mismo número de casos (frecuencias), siempre teniendo en cuenta que, previamente, la distribución ha sido ordenada según los valores de la variable en estudio, de menor a mayor. Si la distribución tiene frecuencia total par, los casos “frontera” entre los dos grupos en que queda dividida la distribución son dos, por lo que, si estos casos asumen valores diferentes en la variable estudiada, podría ocurrir que hubiera dos medianas diferentes. En tal caso, se suele tomar, como convenio, el promedio de de ambos valores para tener una única mediana.

En R, la función para obtener la mediana de una variable es median(). De este modo, para obtener el salario mediano de la variable SALARIO de los trabajadoes de la empresa, ejecutaremos el código:

# Mediana  
  
mediana <- median(datos$SALARIO)  
mediana

## [1] 15

En el ejemplo de la variable SALARIO, la mediana es 15. Es decir, 15 es el salario percibido por el caso 25, que es el trabajador que divide la distribución de frecuencias en dos grupos de 24 trabajadores: 24 que ganan un salario menor o igual que el caso 25 (menos o igual que 15 cientos de euros), y otros 24 trabajadores que ganan más o lo mismo que el caso en la posición 25 (o sea, más o iagual que 15 cientos de euros).

Como ventaja de la mediana contamos con que no es sensible a la existencia de casos atípicos o outliers, y que se puede calcular en el caso de atributos o factores en escala ordinal. Como desventajas tenemos que no tiene por qué ser única, y que no tiene en cuenta la totalidad de los valores de la distribución.

Con **la moda** hacemos referencia al valor (o valores) que posee (o poseen) una mayorv frecuencia absoluta.

En R, la moda se calcula mediante la función Mode() del paquete {DescTools}, que habremos de activar con library() (e instalar previamente, si aún no tenemos instalado este paquete):

# Moda  
  
library(DescTools)  
moda <- Mode(datos$SALARIO)  
moda

## [1] 15  
## attr(,"freq")  
## [1] 11

Como podemos apreciar, la moda de la distribución es 15 (un salario de 1500 euros), que aparece en la distribución en 11 ocasiones (la frecuencia absoluta de ese salario es 11).

La moda puede ser calculada en atributos o factores en escala nominal. Como inconveniente principal tenemos que no tiene por qué ser un valor único (existen distribuciones multimodales).

Existen otras medidas que son de posición no central, principalmente lo que llamamos **cuantiles**. La naturaleza de loc cuantiles es fácil de comprender si los consideramos una generalización de la mediana. Ya sabemos que, ordenados los valores (y por tanto, los casos que toman dichos valores) de una distribución de frecuencias de una variable de menor a mayor, la mediana es el valor (o valores, porque pueden existir dos medianas, aunque vamos a suponer que solo hay una)) de la variable correspondiente al caso que divide a la distribución en dos grupos con el mismo número de frecuencias. Pues bien, si en lugar de dividir a la distribución de frecuencias en dos grupos con el mismo número de elementos, la dividimos en 4 grupos, estaremos hablando de tres valores correspondientes a los casos que delimitan a esos cuatro grupos. Estos valores serán los **cuartiles** de la distribución.

Si queremos dividir la distribución de 9 valores de la variable que toman los casos “frontera” que separan a estos 10 grupos. Esos valores serán los **deciles**. Y si queremos dividir la distribución de frecuencias en 100 grupos con el mismo número de casos o individuos, estaríamos hablando de 99 valores de la variable que toman los casos “frontera” que separan a estos 100 grupos. Esos valores serán los **percentiles**.

En R, la función para calcular los diferentes cuantiles es quantile(). Para calcular, por ejemplo, los cuartiles de la variable SALARIO, procederemos así:

# Calcular los cuartiles  
cuartiles <- quantile(datos$SALARIO, probs = c(0.25, 0.5, 0.75))  
cuartiles

## 25% 50% 75%   
## 12 15 20

El argumento probs = informa de la proporción de los casos que han de quedar por detrás (con valores menores o iguales) de cada uno de los casos que hacen de “frontera” entre los grupos. En el caso de los cuartiles, estos son 0.25, 0.5 (este cuartil es, a su vez, la mediana de la distribución) y 0.75. Vemos cómo los cuartiles son 12, 15 y 20.

## 4.4 Medidas de dispersión o variabilidad.

Las medidas de dispersión cuantifican **lo cerca o lejos que, en general, los valores asumidos por los casos de una distribución de frecuencias se hallan respecto a una medida de posición central**. Si la medida de dispersión toma un valor muy elevado, querrá decir que la medida de posición central no representa bien a la distribución de frecuencias, ya que, en general, los casos toman valores alejados de dicha medida.

La medida de posición central a la que suelen hacer referencia las medidas de dispersión es la media aritmética.

Existen múltiples medidas de dispersión, que principalmente se dividen en **medidas absolutas** (que se expresan en ciertas unidades, como por ejemplo euros, o euros al cuadrado) y **medidas relativas** (que carecen de unidades y siven, por tanto, para comparar la dispersión entre distribuciones de frecuencias expresadas en distintas unidades).

La medida de dispersión absoluta más utilizada es la **varianza**, cuya fórmula es:

Hemos de tener en cuenta en la fórmula anterior que N es la frecuencia total, y h es el número de valores diferentes que toma la variable.

Si las frecuencias de todos los valores de la variable son 1 (distribución de frecuencias unitarias), lógicamente la varianza pasará a ser:

En realidad, la varianza es el promedio de las diferencias que existen entre los valores que toma la variable y la media aritmética de esta, diferencias que son elevadas al cuadrado para evitar la compensación entre diferencias por los signos.

Una limitación de la varianza viene referida a que, debido al exponente del paréntesis, puede tomar valores muy elevados. Para evitar el inconveniente, una medida alternativa es la **desviación típica**, que queda definida como la raíz cuadrada positiva de la varianza:

Otra media de dispersión muy utilizada, sobre todo en *Econometría*, es la *varianza insesgada* o **cuasivarianza**, cuya fórmula es:

Como siempre, si las frecuencias de todos los valores de la variable son 1 (distribución de frecuencias unitarias), lógicamente la cuasivarianza pasará a ser:

En cuanto a una medida de dispersión relativa, cabe nombrar al **coeficiente de variación de Pearson**, definido como el cociente entre la desviación típica y la media aritmética (en valor absoluto):

El coeficiente de variación informa del número de medias aritméticas que “caben” en la desviación típica de una distribución de frecuencias. A mayor coeficiente, mayor dispersión y menor representatividad de la media aritmética con respecto a la distribución. Además, pueden compararse coeficientes de distribuciones expresadas en unidades diferentes (medida relativa).

A continuación, vamos a calcular varianza, desviación típica, cuasivarianza, y coeficiente de variación en R. Para ello, hemos de tener en cuenta que la función var() de R, en realidad, calcula la cuasivarianza. Para obtener la varianza, pues, hemos de realizar una corrección (en realidad, para un número de casos muy grande, ambas medidas prácticamente coinciden):

# Varianza  
  
varianza <- var(datos$SALARIO)\*(N-1)/N # recordar que la frecuencia total N ya fue calculada  
varianza

## [1] 30.48813

# Desviación típica  
  
desv <- varianza ^ (1/2)  
desv

## [1] 5.521606

# Cuasivarianza  
  
cuasivarianza <- var (datos$SALARIO)  
cuasivarianza

## [1] 31.1233

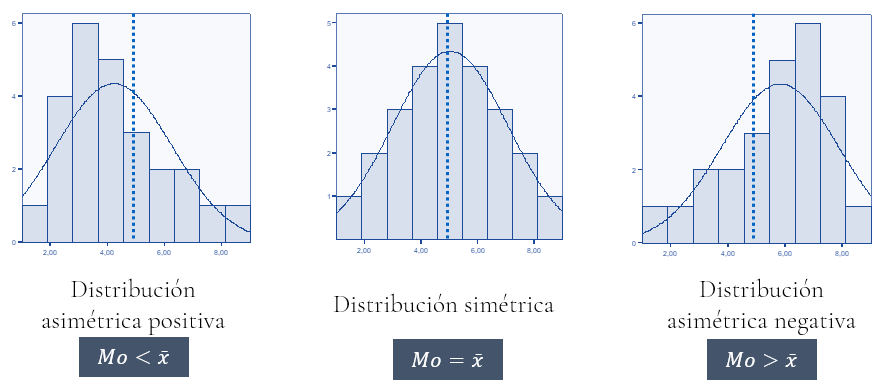
# Coeficiente de variación  
  
cvariacion <- desv / abs(media)  
cvariacion

## [1] 0.3442222

## 4.5 Medidas de forma.

Las medidas de forma cuantifican el grado de deformación vertical y horizontal de la representación gráfica de una distribución de frecuencias. Son de dos tipos: medidas de asimetría y medidas de apuntamiento o curtosis.

Las **medidas de asimetría** miden el grado de deformación vertical con respecto a un “eje de simetría”, que es aquel que pasa por el valor medio de la distribución. Si suponemos que la distribución es unimodal y campaniforme, tendremos los casos que se muestran en la figura:



Tipos de asimetría

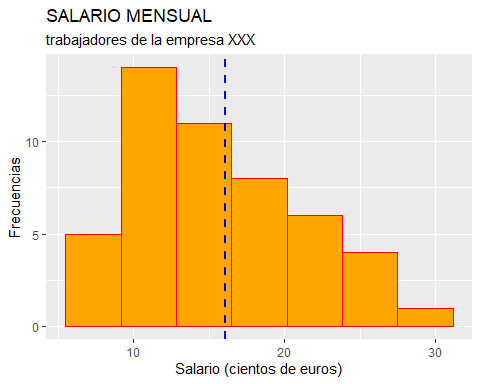
El tipo y grado de asimetría se puede obtener mediante el **coeficiente de asimetría de Fisher**. Este coeficiente toma valor negativo si la distribución es *asimétrica negativa* (mayores frecuencias a la derecha de la media), valor positivo si la distribución es *asimétrica positiva* (mayores frecuencias a la izquierda de la media), y se acerca a 0 en caso de que la distribución seaaproximadamente *simétrica*, aunque pueden darse casos de distribuciones no simétricas con coeficiente 0. En R, se puede obtener el coeficiente de asimetría mediante la función skewness() del paquete {moments}:

# Coeficiente de asimetría de Fisher  
  
library(moments)  
asimetria <- skewness(datos$SALARIO)  
asimetria

## [1] 0.413764

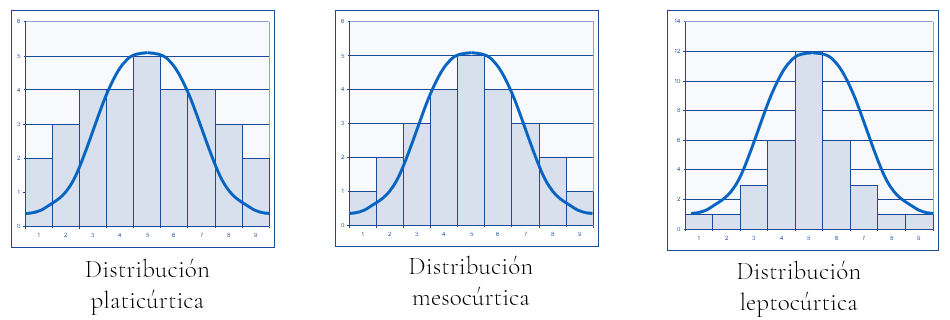
El valor obtenido para la variable SALARIO de nuestra distribución de frecuencias de los trabajadores de la empresa es positivo, lo que indica asimetría positiva: las mayores frecuencias se localizan a la derecha de la media. Esto se puede comprobar fácilmente construyendo el histograma de la variable SALARIO y trazando una línea vertical que pase por la media salarial. Para ello usamos el paquete {ggplot2}:

library (ggplot2)  
ggplot(data = datos, map = aes(x = SALARIO)) +  
 geom\_histogram(bins = 7, colour = "red", fill = "orange") +  
 geom\_vline(aes(xintercept = mean(SALARIO)), colour = "blue", linetype = "dashed", size = 1) +  
 ggtitle("SALARIO MENSUAL", subtitle = "trabajadores de la empresa XXX")+  
 xlab("Salario (cientos de euros)") +  
 ylab("Frecuencias")



Como comprobamos, la distribución es claramente asimétrica positiva.

En cuanto a las **medidas de curtosis**, miden el grado de deformación horizontal con respecto a una distribución “tipo”, la distribución normal. Suponemos previamente que la distribución de frecuencias estudiada es campaniforme, unimodal y simétrica (o con ligera asimetría). Pueden darse los casos que se muestran en la figura:



Tipos de distribución según el apuntamiento o curtosis

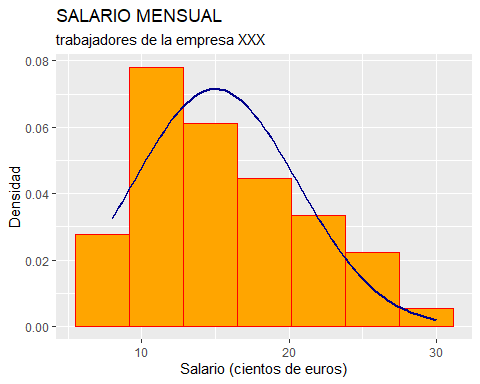
El tipo y grado de apuntamiento o curtosis se puede obtener mediante el **coeficiente de apuntamiento de Fisher**. Este coeficiente toma valor negativo si la distribución es *platicúrtica* (más aplastada que la distribución normal), valor positivo si la distribución es *leptocúrtica* (más apuntada que la distribución normal), y se acerca a 0 en caso de que la distribución sea aproximadamente igual de apuntada que la distribución normal. En R, se puede obtener el coeficiente de asimetría mediante la función kurtosis() del paquete {moments}. Hay que tener en cuenta que para que esta versión coincida con lo dicho anteriormente, al valor calculado hay que restarle el valor “3”:

# Coeficiente de apuntamiento o curtosis de Fisher  
  
curtosis <- kurtosis(datos$SALARIO) - 3  
curtosis

## [1] -0.8232965

Se aprecia como el coeficiente (corregido) es menor que 0, por lo que la distribución de frecuencias de la variable SALARIO es platicúrtica (más “aplastada” que la distribución de frecuencias normal. También hay que tener en cuenta que debemos tener precaución en la aplicación del coeficiente, ya que vimos con anterioridad que la distribución no es aproximadamente simétrica. Gráficamente, podemos comprobar lo anterior representando el histograma y una curva normal que posea la misma moda (que ya calculamos anteriormente). Para que sean comparables, debemos transformar el eje “y” del gráfico, pasando de “frecuencias” a “densidad”, para lo cuál se incluye en el geom\_histogram() el argumento aes(y = ..density..):

ggplot(data = datos, map = aes(x = SALARIO)) +  
 geom\_histogram(bins = 7, colour = "red", fill = "orange", aes(y = ..density..)) +  
 stat\_function(fun = dnorm, args = list(mean = moda, sd = sd(datos$SALARIO)), colour = "darkblue", size = 1) +  
 ggtitle("SALARIO MENSUAL", subtitle = "trabajadores de la empresa XXX")+  
 xlab("Salario (cientos de euros)") +  
 ylab("Densidad")



## 4.6 Materiales para realizar las prácticas del capítulo.

En esta sección se muestran los links de acceso a los diferentes materiales (*scripts*, datos…) necesarios para llevar a cabo los contenidos prácticos del capítulo.

**Datos (en formato Microsoft (R) Excel (R)):**

* trabajadores.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1e_h8P_0sBuD5Nvw6nkeWtXXeiqItFfdI/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true))

**Scripts:**

* explora\_trabajadores.R ([obtener aquí](https://drive.google.com/file/d/1pfyGPGQRmgOyo9wrZ7fr_KAFj49ecXID/view?usp=sharing))

# 5 Análisis previo de datos.

## 5.1 Introducción.

Antes de la aplicación de técnicas complejas que permitan extraer de los datos conclusiones relevantes, es necesario realizar unas **tareas previas** destinadas a conseguir dos objetivos:

* **Preparar nuestros datos** para que puedan ser procesados correctamente sin provocar distorsiones en los resultados.
* Obtener una **visión inicial de la información** que esconden los datos, fundamentalmente en cuanto a las medidas básicas que caracterizan la distribución de frecuencias de las variables en las que se estructuran estos, así como, en el caso de contar con más de una variable, de las relaciones estadísticas que existen entre ellas.

Además, es preciso tener en cuenta que, usualmente, es conveniente que estos rasgos iniciales que caracterizan a nuestra muestra o población sean plasmados de un modo **visualmente amigable**, claro y conciso.

En esta práctica, por medio de un ejemplo basado en información económico-financiera de una muestra constituida por 100 empresas dedicadas a la producción de electricidad mediante tecnología eólica, se mostrarán una serie de buenas prácticas y análisis básicos útiles a la hora de preparar y analizar inicialmente nuestro conjunto de datos.

Vamos a suponer que trabajamos dentro de un **proyecto** que hemos creado previamente, de nombre “explora”. Dentro de la carpeta del proyecto guardaremos el *script* llamado “[explora\_describe.R](https://drive.google.com/file/d/1VqZX81h15JZhF7KXXj3alLd08o5ywkX2/view?usp=sharing)”, y el archivo de Microsoft® Excel® llamado “[eolica\_100\_mv.xlsx](https://docs.google.com/spreadsheets/d/1f9KV5epNpBRM6v9sKPtgFDkzMlOo75Y2/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true)”. Si abrimos este último archivo, comprobaremos que se compone de tres hojas. La primera muestra un mensaje sobre el uso de los datos, la segunda recoge la descripción de las variables consideradas, y la tercera (hoja “Datos”) almacena los datos que debemos importar. Estos datos se corresponden con diferentes variables económico-financieras de las 100 empresas productoras de electricidad mediante generación eólica con mayor volumen de activo.

Es muy importante observar que existen variables con datos faltantes (*missing values*). En concreto, podemos identificar estas faltas de dato por la existencia de celdas en blanco; pero también por la existencia de celdas con el texto “n.d.” (no dato) o “s.d.” (sin dato). Es muy importante identificar el modo en que quedan recogidos los datos faltantes en la hoja de cálculo, ya que **tendremos que aplicar código adicional** en el comando de importación de R **para que estos casos queden correctamente recogidos como *NAs*** (*not available*).

Cerraremos el archivo de Microsoft® Excel®, “eolica\_100\_mv.xlsx” y volveremos a RStudio. Después, abriremos nuestro *script* “explora\_describe.R” con File → Open File… Este *script* contiene el programa que vamos a ir ejecutando en la práctica.

La primera línea / instrucción en el *script* es:

rm(list = ls())

La instrucción tiene como objeto limpiar el *Environment* (memoria) de objetos de anteriores sesiones de trabajo. Para importar los datos que hay en la hoja “Datos” del archivo de Microsoft® Excel® llamado “eolica\_100\_mv.xlsx”, ejecutaremos el código:

library(readxl)  
eolica\_100 <- read\_excel("eolica\_100\_mv.xlsx", sheet = "Datos",  
 na = c("n.d.", "s.d."))  
summary (eolica\_100)

## NOMBRE RES ACTIVO FPIOS   
## Length:100 Min. : -5661.5 Min. : 24944 Min. : -77533   
## Class :character 1st Qu.: 669.5 1st Qu.: 34547 1st Qu.: 2305   
## Mode :character Median : 2084.5 Median : 46950 Median : 11936   
## Mean : 11529.8 Mean : 277270 Mean : 123743   
## 3rd Qu.: 3806.7 3rd Qu.: 85610 3rd Qu.: 28292   
## Max. :727548.0 Max. :13492812 Max. :6904824   
## NA's :1 NA's :1   
##   
## RENECO RENFIN LIQUIDEZ ENDEUDA   
## Min. :-2.813 Min. :-359.773 Min. : 0.0140 Min. : 0.917   
## 1st Qu.: 1.558 1st Qu.: 2.556 1st Qu.: 0.6567 1st Qu.: 50.852   
## Median : 4.236 Median : 15.326 Median : 1.0650 Median : 83.346   
## Mean : 5.416 Mean : 17.243 Mean : 2.7214 Mean : 72.227   
## 3rd Qu.: 7.970 3rd Qu.: 31.307 3rd Qu.: 1.6078 3rd Qu.: 95.388   
## Max. :35.262 Max. : 588.190 Max. :128.4330 Max. :140.745   
## NA's :2 NA's :2   
##   
## MARGEN SOLVENCIA APALANCA MATRIZ   
## Min. :-2248.157 Min. :-40.74 Min. :-8254.11 Length:100   
## 1st Qu.: 12.316 1st Qu.: 4.71 1st Qu.: 16.13 Class :character   
## Median : 26.618 Median : 16.65 Median : 161.97 Mode :character   
## Mean : 3.228 Mean : 27.57 Mean : 345.03   
## 3rd Qu.: 39.590 3rd Qu.: 45.59 3rd Qu.: 623.13   
## Max. : 400.899 Max. : 99.08 Max. :12244.35   
## NA's :2   
##   
## DIMENSION   
## Length:100   
## Class :character   
## Mode :character

Por defecto, R considera las celdas en blanco de la hoja de cálculo como NAs; pero hemos de advertirle del resto de posibilidades que existen en la hoja para comunicar que falta un dato determinado, como ya se ha comentado. Para ello, hemos añadido en la función read\_excel() el argumento na =, que recoge los contenidos de celda de la hoja de cálculo que indican que falta el dato en cuestión.

Por otro lado, R ha considerado la primera columna como una variable de tipo cualitativo, atributo, o factor. En realidad, esta columna no es una variable, sino que está formada por los nombres de los diferentes casos u observaciones. Para evitar que R tome la columna de los nombres de los casos como una variable más, podemos redefinir nuestro *data frame* diciéndole que tome esa primera columna como el conjunto de los *nombres* de los casos:

eolica\_100 <- data.frame(eolica\_100, row.names = 1)  
summary (eolica\_100)

## RES ACTIVO FPIOS RENECO   
## Min. : -5661.5 Min. : 24944 Min. : -77533 Min. :-2.813   
## 1st Qu.: 669.5 1st Qu.: 34547 1st Qu.: 2305 1st Qu.: 1.558   
## Median : 2084.5 Median : 46950 Median : 11936 Median : 4.236   
## Mean : 11529.8 Mean : 277270 Mean : 123743 Mean : 5.416   
## 3rd Qu.: 3806.7 3rd Qu.: 85610 3rd Qu.: 28292 3rd Qu.: 7.970   
## Max. :727548.0 Max. :13492812 Max. :6904824 Max. :35.262   
## NA's :1 NA's :1 NA's :2   
##   
## RENFIN LIQUIDEZ ENDEUDA MARGEN   
## Min. :-359.773 Min. : 0.0140 Min. : 0.917 Min. :-2248.157   
## 1st Qu.: 2.556 1st Qu.: 0.6567 1st Qu.: 50.852 1st Qu.: 12.316   
## Median : 15.326 Median : 1.0650 Median : 83.346 Median : 26.618   
## Mean : 17.243 Mean : 2.7214 Mean : 72.227 Mean : 3.228   
## 3rd Qu.: 31.307 3rd Qu.: 1.6078 3rd Qu.: 95.388 3rd Qu.: 39.590   
## Max. : 588.190 Max. :128.4330 Max. :140.745 Max. : 400.899   
## NA's :2 NA's :2   
##   
## SOLVENCIA APALANCA MATRIZ DIMENSION   
## Min. :-40.74 Min. :-8254.11 Length:100 Length:100   
## 1st Qu.: 4.71 1st Qu.: 16.13 Class :character Class :character   
## Median : 16.65 Median : 161.97 Mode :character Mode :character   
## Mean : 27.57 Mean : 345.03   
## 3rd Qu.: 45.59 3rd Qu.: 623.13   
## Max. : 99.08 Max. :12244.35

Observaremos que ya no aparece NOMBRE, puesto que la columna correspondiente ya no es considerada como una variable.

## 5.2 Análisis de una variable.

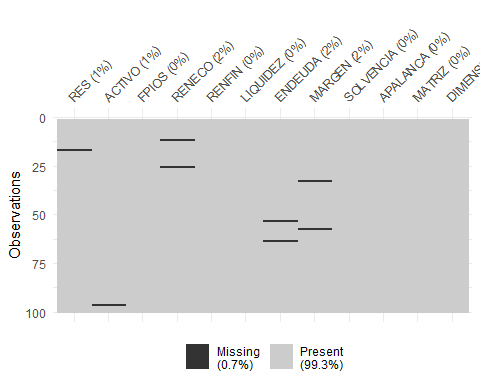
### 5.2.1 Buscando *missing values* y *outliers*.

Vamos a suponer que la variable que queremos estudiar es la variable *Rentabilidad Económica* (RENECO).

La primera acción que debe realizarse es comprobar que todos los casos (empresas) tienen su correspondiente dato o valor para la variable (RENECO), es decir, que no existen **valores perdidos o *missing values***.

Para tener una idea general, se puede utilizar la función vis\_miss() del paquete {visdat}, que nos localizará gráficamente los *missing values* de las diferentes variables, y calculará el porcentaje de casos que supone, con respecto al total de observaciones:

library(visdat)  
vis\_miss(eolica\_100)



Puede observarse cómo, en el caso concreto de la variable RENECO, un 2% de los casos no tienen dato (es decir, 2 casos de los 100). Para localizar los casos concretos, puede recurrirse a utilizar las herramientas de manejo de *data frames* del paquete {dplyr}. En concreto, realizaremos una copia del *data frame* original, “eolica\_100”, a la que llamaremos “muestra”, que es con la que trabajaremos (para mantener la integridad del *data frame* original); y filtraremos los casos para detectar aquellos que carecen de valor en la variable RENECO:

library (dplyr)  
muestra<- select(eolica\_100, everything())  
muestra %>% filter(is.na(RENECO)) %>% select(RENECO)

La función is.na() comprueba si, en la posición correspondiente a una fila o caso, para la variable escrita en el argumento; hay o no un dato o valor. Como resultado se obtienen dos empresas, para las que se puede comprobar que no hay valor para la variable RENECO:

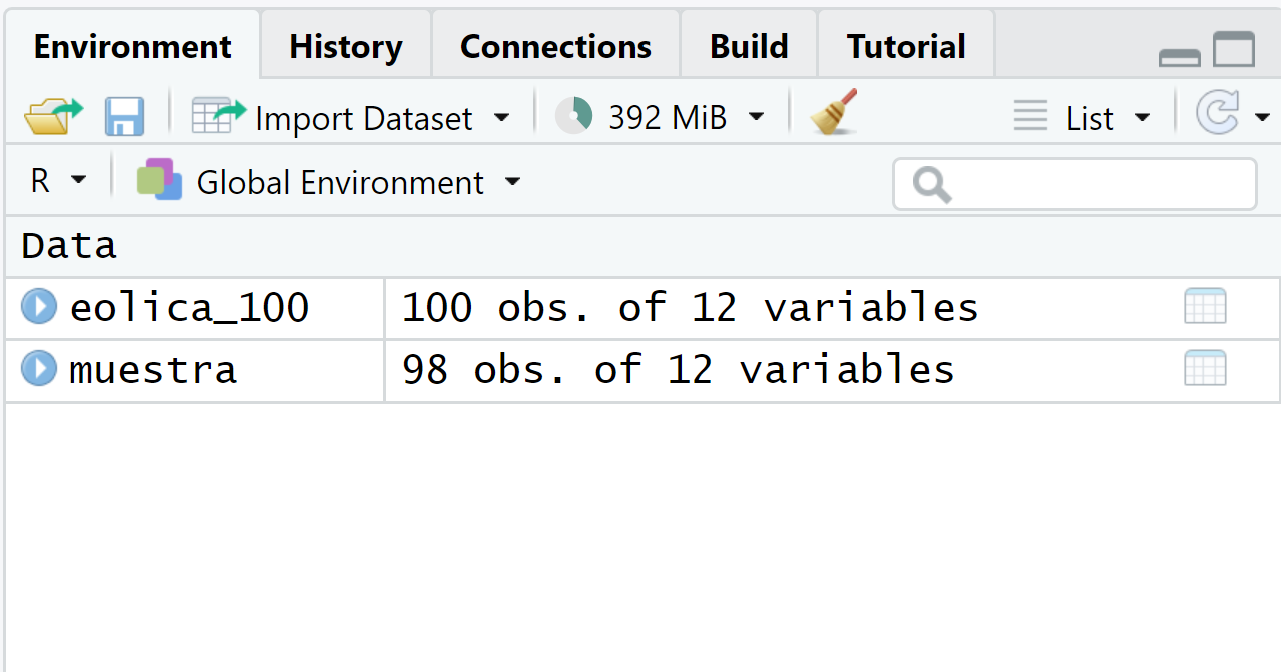
## RENECO  
## Viesgo Renovables SL. NA  
## Sargon Energias SLU NA

Ante la existencia de *missing values*, se puede actuar de varios modos. Por ejemplo, se puede intentar obtener por otro canal de información el conjunto de valores de RENECO que no están disponibles, o recurrir a alguna estimación para los mismos y asignarlos. En caso de que esto sea difícil, se puede optar, simplemente, por eliminar estos casos, en especial cuando representan un porcentaje muy reducido respecto al total de casos. En nuestro ejemplo, vamos a suponer que hemos optado por esta última vía, al no conseguir unos valores más o menos verosímiles de RENECO para las empresas de las que se carece de dato. Esta **eliminación de casos** se podrá realizar mediante el código:

muestra <- muestra %>% filter(! is.na(RENECO))

El operador **!** significa “no”.

Podemos comprobar cómo en el *Global Environment* aparece el *data frame* “muestra” con dos casos menos (98):

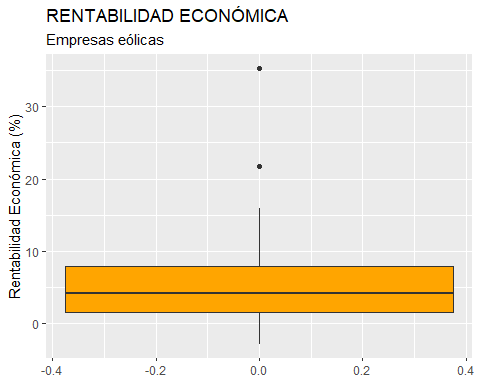


Global Environment.

Una vez tratados los casos con valores perdidos o *missing values*, **conviene detectar la posible presencia en la muestra de *outliers*** o casos atípicos, que pudieran desvirtuar los resultados derivados de ciertos análisis. Al trabajar con una sola variable métrica (la rentabilidad económica, RENECO), podemos intentar realizar esta tarea **representando gráficamente** la variable mediante un ***boxplot*** o *gráfico de caja*. Aplicaremos, por ejemplo, el código siguiente, que utiliza la gramática del paquete {ggplot2}:

library (ggplot2)  
ggplot(data = muestra, map = (aes(y = RENECO))) +  
 geom\_boxplot(fill = "orange") +  
 ggtitle("RENTABILIDAD ECONÓMICA", subtitle = "Empresas eólicas") +  
 ylab("Rentabilidad Económica (%)")

Obteniéndose el gráfico:



La “caja” contiene el 50% de los valores de la variable que toman los casos centrales (los que van del primer cuartil al tercero, cuya diferencia se llama *rango intercuartílico*), y contiene una línea horizontal que es la mediana (segundo cuartil). Por arriba sobresale un segmento que llega al mayor valor de la variable que toma algún caso y que no llega a ser atípico; y por debajo de la caja otro segmento que llega al menor valor de la variable que toma algún caso y que no llega a ser atípico. Los casos atípicos o outliers son aquellos que toman valores que se alejan más de 1.5 veces del rango intercuartílico (altura de la caja) del tercer cuartil, por arriba; o del primer cuartil, por abajo. Se registran mediante puntos.

En nuestro caso, el *boxplot* ratifica la existencia de dos casos atípicos. Para identificar esos dos casos concretos, podemos recurrir al paquete {dplyr}, y **establecer un filtro** con el siguiente código:

Q1 <- quantile (muestra$RENECO, c(0.25))  
Q3 <- quantile (muestra$RENECO, c(0.75))  
muestra %>%   
 filter(RENECO > Q3 + 1.5\*IQR(RENECO) | RENECO < Q1 - 1.5\*IQR(RENECO)) %>%  
 select(RENECO)

En el código anterior, las dos primeras filas calculan los cuartiles primero (Q1) y tercero (Q3) mediante la función quantile(). En esta función, es preciso poner como segundo argumento la proporción de casos que van a quedar por debajo del “cuantil” en cuestión (por ejemplo, el primer cuartil se calcula poniendo 0.25, dado que deja por debajo al 25% de casos con menor valor en la variable). Luego se filtran los *outliers* mediante la función filter() de {dplyr} , calculados como aquellos casos con valores de RENECO mayores que Q3 más 1,5 veces el rango intercuartílico de la variable; o menores que Q1 menos 1,5 veces dicho rango intercuartílico. Para calcular el rango intercuartílico se recurre a la función IQR(). Finalmente, con select(), se muestran los casos en la consola de R-Studio:

## RENECO  
## Molinos Del Ebro SA 35.262  
## Sierra De Selva SL 21.761

Como ocurría con los *missing values*, el tratamiento de los *outliers* depende de la información que se tenga, existiendo varias alternativas (corrección del dato, estimación, etc.) Si no se tiene información fiable, y los *outliers* no representan una gran proporción respecto al total de casos, puede optarse por su eliminación de la muestra. En este ejemplo, efectivamente, **eliminaremos estas dos empresas con comportamiento atípico** en la rentabilidad económica (RENECO), a fin de que su presencia en la muestra **no** **distorsione los resultados en la aplicación posterior de ciertas técnicas** (por ejemplo, un ANOVA o un análisis de regresión). Podemos hacerlo creando un nuevo *data frame* a partir de “muestra”; pero sin esos dos casos. Ese nuevo *data frame* se llamará, por ejemplo, “muestra\_so”:

muestra\_so <- muestra %>%  
 filter(RENECO <= Q3 + 1.5\*IQR(RENECO) & RENECO >= Q1 - 1.5\*IQR(RENECO))

Es importante observar que, en el código de la función filter(), las desigualdades deben cambiar, así como el operador “**|**” por el operador “**&**”. En el *Global Environment* podemos comprobar cómo el *data frame* “muestra\_so” posee el mismo número de variables que el *data frame* “muestra”; pero con dos observaciones o casos menos (96).

### 5.2.2 Descripción de una variable.

Una vez que se tiene preparada la base de datos, con un tratamiento adecuado de los *missing values* y de los *outliers,* y **antes** de proceder a la aplicación de una técnica adecuada según los objetivos perseguidos en el estudio; suelen presentarse una serie de **gráficos** básicos y **medidas** descriptivas que proporcionan una **idea inicial de la estructura** del sector para la variable o variables analizadas. Nos referimos a medidas y/o gráficos de posición, dispersión y forma (asimetría y curtosis).

Antes de ello, se puede presentar una tabla donde se recoja la distribución de frecuencias de la variable. Si son muchos los casos (en el ejemplo, 96), la distribución podría presentarse agrupada en intervalos, como ya se vio en el capítulo 4 de este libro. De este modo, el código para generar la tabla podría ser:

# Tabla de datos (distribución de frecuencias agrupadas en intervalos)  
  
library (knitr)  
library (kableExtra)  
muestra\_so <- muestra\_so %>% arrange(RENECO, row.names(muestra\_so))  
  
# Crear los intervalos  
muestra\_so$intervalos <- cut(muestra\_so$RENECO, breaks = 5, include.lowest = TRUE)  
  
# Contar las frecuencias de cada intervalo  
conteo\_intervalos <- table(muestra\_so$intervalos)  
  
# Convertir el resultado a un data frame para una mejor visualización  
conteo\_intervalos\_df <- as.data.frame(conteo\_intervalos)  
  
# Renombrar las columnas para mayor claridad  
colnames(conteo\_intervalos\_df) <- c("Intervalo", "Frecuencia")  
  
# Calcular y guardar la frecuencia total  
N\_agre <- sum(conteo\_intervalos\_df$Frecuencia)  
  
# Calcular frecuencias absolutas acumuladas  
conteo\_intervalos\_df$Frecuencia\_acum <- cumsum(conteo\_intervalos\_df$Frecuencia)  
  
# Calcular frecuencias relativas  
conteo\_intervalos\_df$Frecuencia\_R <- conteo\_intervalos\_df$Frecuencia / N\_agre  
  
# Calcular frecuencias relativas acumuladas  
conteo\_intervalos\_df$Frecuencia\_R\_acum <- cumsum(conteo\_intervalos\_df$Frecuencia\_R)  
  
# Mostrar el resultado  
conteo\_intervalos\_df %>%  
 kable(caption = "Distribución de frecuencias agrupadas en intervalos de la Rentabilidad Económica",  
 col.names = c("Intervalo rentabilidad", "Frecuencia absoluta n(i)",  
 "Frecuencia absoluta acum. N(i)", "Frecuencia relativa f(i)",  
 "Frecuencia relativa acum. F(i)"),  
 format.args = list(decimal.mark = ".", digits = 2)) %>%  
 kable\_styling(full\_width = F, bootstrap\_options = "striped",  
 "bordered", "condensed",  
 position = "center", font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:(nrow(conteo\_intervalos\_df)), bold= F, align = "c")

(#tab:unnamed-chunk-156)(#tab:unnamed-chunk-156)Distribución de frecuencias agrupadas en intervalos de la Rentabilidad Económica

Intervalo rentabilidad

Frecuencia absoluta n(i)

Frecuencia absoluta acum. N(i)

Frecuencia relativa f(i)

Frecuencia relativa acum. F(i)

[-2.83,0.926]

16

16

0.17

0.17

(0.926,4.66]

38

54

0.40

0.56

(4.66,8.4]

21

75

0.22

0.78

(8.4,12.1]

11

86

0.11

0.90

(12.1,15.9]

10

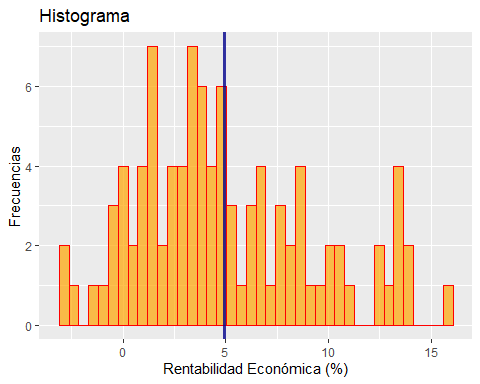
96

0.10

1.00

El **análisis gráfico** suele dar una idea atractiva e intuitiva de la estructura de la distribución de frecuencias de nuestro conjunto de casos en relación con la variable a analizar. Un gráfico fundamental es el **histograma** de la variable estudiada. Para ello, utilizaremos la gramática del paquete {ggplot2}:

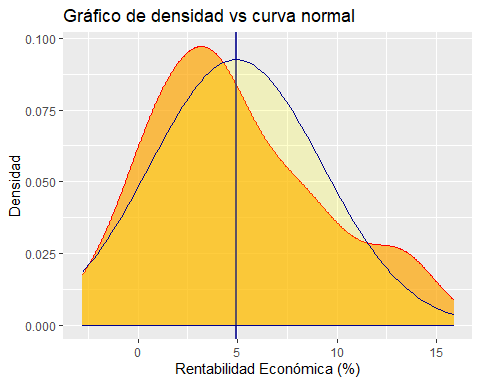
## Descriptivos básicos  
  
# Gráficos básicos  
  
g1 <-  
ggplot(data = muestra\_so, map = aes(x = RENECO)) +  
 geom\_histogram(bins = 40,  
 colour = "red",  
 fill = "orange",  
 alpha = 0.7) +  
 geom\_vline(xintercept = mean(muestra\_so$RENECO),  
 color = "dark blue",  
 size = 1.2,  
 alpha = 0.8) +  
 ggtitle("Histograma")+  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Frecuencias")  
  
g1



En el gráfico vemos de un modo bastante nítido la distribución de frecuencias en cuanto a la rentabilidad económica (RENECO). Se ha incorporado una línea vertical azul (mediante geom\_vline()) para localizar la rentabilidad media. Entre otras cosas, se puede apreciar que la distribución de frecuencias es acampanada y *asimétrica positiva*.

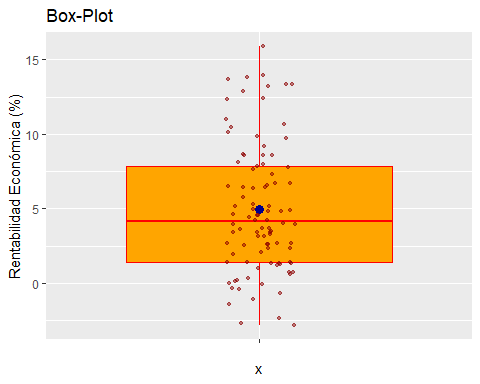
Como complemento al histograma, podemos realizar un gráfico de densidad de RENECO, al que añadiremos una curva normal con la misma media y desviación típica que nuestra distribución de frecuencias, y que se añade mediante stat\_function() y el argumento fun = dnorm. Este gráfico representa la distribución de probabilidad empírica de la muestra, es una especie de histograma “suavizado”. De este modo, se podrán verificar de un modo fácil algunas de las características avanzadas con la observación del histograma, como la asimetría positiva. El código es:

g2 <-  
ggplot(data = muestra\_so, map = aes(x = RENECO)) +  
 geom\_density(colour = "red",  
 fill = "orange",  
 alpha = 0.7) +  
 geom\_vline(xintercept = mean(muestra\_so$RENECO),  
 color = "dark blue",  
 size = 0.8,  
 alpha = 0.8) +  
 stat\_function(fun = dnorm, args = list(mean = mean(muestra\_so$RENECO),  
 sd = sd(muestra\_so$RENECO)),  
 geom = "area", color = "darkblue",   
 fill = "yellow", alpha = 0.2) +  
 ggtitle("Gráfico de densidad vs curva normal")+  
 xlab("Rentabilidad Económica (%)") +  
 ylab("Densidad")  
  
g2



Un tercer gráfico útil es el box-plot una vez se eliminaron los casos outliers, con la incorporación de los valores que toman los casos que componen la muestra, para lo cuál se utiliza el geom\_jitter:

g3 <-  
ggplot(data = muestra\_so, map = (aes(x = "", y = RENECO))) +  
 geom\_boxplot(color = "red",  
 fill = "orange",  
 outlier.shape = NA) +  
 stat\_summary(fun = "mean",  
 geom = "point",  
 size = 3,  
 col = "darkblue") +  
 geom\_jitter(width = 0.1,  
 size = 1,  
 col = "darkred",  
 alpha = 0.50) +  
 ggtitle("Box-Plot") +  
 ylab("Rentabilidad Económica (%)")  
  
g3



Por otro lado, conviene tener conocimiento del valor de las principales **medidas descriptivas** (de posición, dispersión, forma) que caracterizan a la distribución de la variable a analizar. Para ello, vamos a crear un *data frame* llamado, por ejemplo, “estadisticos”, que recogerá las diferentes medidas, calculadas al aplicar a la variable RENECO del *data frame* “muestra\_so” la función de {deplyr} llamada summarise(). Se calcularán la media, desviación típica, valor mínimo, mediana, valor máximo, el coeficiente de asimetría de Fisher, y el coeficiente de apuntamiento o curtosis de Fisher. Precisamente, para poder calcular esta última medida, es preciso activar el paquete {moments}, que contiene la función kurtosis(). La versión del coeficiente de apuntamiento de esta función dispone como distribución perfectamente mesocúrtica el valor de 3, por lo que se le restará 3 en la versión que manejaremos para que la distribución mesocúrtica se sitúe en un coeficiente de 0. El código es:

# Calcular estadísticos  
  
library (moments) # paquete necesario para calcular la curtosis.  
  
estadisticos <- muestra\_so %>% summarise( Media = mean(RENECO),  
 DT = sd(RENECO),  
 Mínimo = min(RENECO),  
 Mediana = median(RENECO),  
 Maximo = max(RENECO),  
 Asimetria = skewness(RENECO),  
 Curtosis = kurtosis(RENECO) - 3)

La ventaja de volcar las medidas y estadísticos en el data frame (de un solo caso) “estadisticos” es que se pueden mostrar los valores en una tabla elegante generada a partir de el mismo mediante las funciones knitr() y kableExtra(), como ya sabemos:

# Mostrar estadisticos  
  
estadisticos %>%  
 kable(caption = "Principales Estadísticos de la Rentabilidad Económica",  
 col.names = c("Media", "Mediana",  
 "Desviación Típica", "Valor mínimo",  
 "Valor Máximo", "C. Asimetría Fisher",  
 "C. Curtosis Fisher"),  
 format.args = list(decimal.mark = ".", digits = 2)) %>%  
 kable\_styling(full\_width = F, bootstrap\_options = "striped",  
 "bordered", "condensed",  
 position = "center", font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:(nrow(estadisticos)), bold= F, align = "c")

(#tab:unnamed-chunk-162)(#tab:unnamed-chunk-162)Principales Estadísticos de la Rentabilidad Económica

Media

Mediana

Desviación Típica

Valor mínimo

Valor Máximo

C. Asimetría Fisher

C. Curtosis Fisher

4.9

4.3

-2.8

4.1

16

0.56

-0.37

La interpretación de estas medidas fueron comentadas en el capítulo 4.

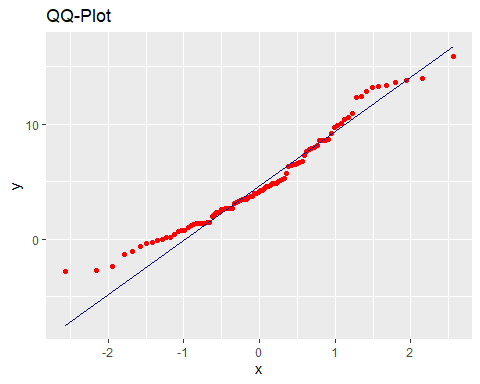
### 5.2.3 Normalidad.

En muchas técnicas multivariantes basadas en métodos inferenciales (por ejemplo, análisis de la varianza, o en la regresión lineal), se requiere que las variables sigan una **distribución normal**. Para comprobarlo, se puede a recurrir a análisis gráficos o a análisis formales, estos últimos basados en contrastar la hipótesis nula de normalidad.

Vamos a mostrar un **método gráfico** muy extendido. Comprobaremos la normalidad de la variable RENECO mediante un ***gráfico qq*** (cuantil-cuantil), que compara los cuantiles de nuestra muestra con los de una distribución normal teórica (con la misma media y desviación típica). Si los puntos se sitúan cercanos a la diagonal, entonces se asumirá un comportamiento (aproximadamente) normal. El código para realizar el gráfico con las herramientas del paquete {ggplot2} es:

## Normalidad  
  
# Grafico QQ  
  
g4 <-  
ggplot(data = muestra\_so, aes(sample = RENECO)) +  
 stat\_qq(colour = "red") +   
 stat\_qq\_line(colour = "dark blue") +  
 ggtitle("QQ-Plot")  
  
g4

Y el resultado:



A veces, es difícil obtener una conclusión sólida con el gráfico *qq*; aunque en el ejemplo se aprecia, sobre todo en los primeros puntos, una separación notable de estos con respecto a la línea, lo que induce a pensar en que podría **no** seguirse una distribución normal.

Si queremos ser más precisos, en lugar de un análisis gráfico se puede recurrir a realizar un análisis formal, basado en la realización de **contrastes de hipótesis**. Una prueba muy usual es la **prueba de normalidad de *Shapiro y Wilk***, que tiene un buen comportamiento en muestras relativamente reducidas. En esta prueba, la hipótesis nula equivale al supuesto de normalidad. Para un 5% de significación estadística, un p-valor superior a 0.05 implicará el no-rechazo de la hipótesis de normalidad. Para realizar la prueba, se ejecutará el código:

# Prueba de Shapiro-Wilk  
  
shapiro.test(x = muestra\_so$RENECO)

El resultado obtenido en la consola:

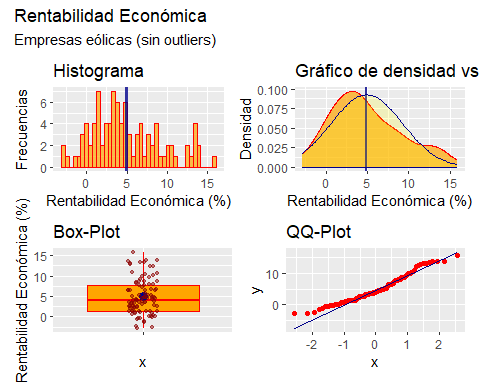
##   
## Shapiro-Wilk normality test  
##   
## data: muestra\_so$RENECO  
## W = 0.9605, p-value = 0.005523

Como el p-valor es (muy) inferior a 0.05, **se rechaza la hipótesis nula de normalidad en la distribución**, lo que implica que, para una significación estadística del 5%, admitimos que RENECO **no** sigue, para nuestra muestra, un comportamiento normal, como ya se anticipó con el gráfico *qq*.

### 5.2.4 Resumen: los 4 gráficos básicos en la descripción de una variable.

En definitiva, para describir de un modo inicial una distribución de frecuencias de una variable (en escala métrica), podrían analizarse los 4 gráficos que se han comentado anteriormente. Estos gráficos se pueden presentar conjuntamente, para ahorrar espacio en un informe, utilizando el paquete {patchwork}, que permite combinar e integrar en una sola imagen varios gráficos generados con {ggplot2}. En nuestro ejemplo, vamos a generar una figura que integra los 4 gráficos anteriores (que hemos denominado “g1”, “g2”, “g3” y “g4”). Para ello creamos el objeto “resumen”, que integra los gráficos, mediante una asignación con la sintaxis del paquete {patchwork}: El operador / indica que los gráficos siguientes se dispondrán inmediatamente debajo; mientras que | indica que el gráfico siguiente se dispone al lado del anterior. Luego, se le añade también un título y un subtítulo:

## Resumen gráfico  
  
library (patchwork)  
resumen <- (g1 | g2)/(g3 | g4)  
resumen <- resumen +   
 plot\_annotation(  
 title = "Rentabilidad Económica",  
 subtitle = "Empresas eólicas (sin outliers)")  
resumen



## 5.3 Análisis de múltiples variables.

Son muchas las técnicas aplicadas al análisis de datos económicos basadas en una distribución de frecuencias multivariante. En este apartado nos centraremos en el caso de **variables métricas**, ya que al caso de atributos, variables categóricas o factores; le dedicaremos un capítulo en exclusiva. Algunas técnicas multivariantes son el análisis de componentes principales, el análisis de regresión, el análisis clúster…

Todas estas metodologías requieren, de nuevo, de una fase inicial que ponga a punto la base de datos y ofrezca una fotografía de cómo es la situación en cuanto a las variables en estudio. En este sentido, es conveniente aplicar, para cada variable por separado, algunos de los análisis gráficos básicos vistos anteriormente.

A estos análisis básicos hay que añadir, principalmente, algún análisis previo adicional, destinado fundamentalmente a comprobar el **grado de intensidad en la relación estadística** entre las variables implicadas. Antes de abordar esta cuestión, hemos de pararnos en una casuística específica que se presenta cuando trabajamos con numerosas variables: la detección de casos atípicos o *outliers*.

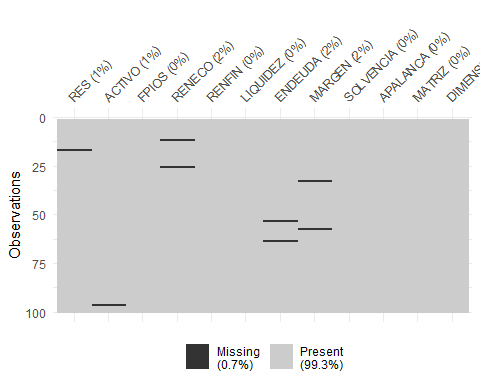
### 5.3.1 Localización de *missing values* y *outliers*.

Para trabajar con múltiples variables, en primer lugar es preciso localizar los casos con **valores perdidos o *missing values***, para decidir cómo procesarlos (eliminación del caso, estimación del valor faltante, etc.)

Vamos a imaginar que queremos realizar un análisis en el que tendremos en cuenta las variables RENECO (rentabilidad económica), ACTIVO (volumen de activos de la empresa), MARGEN (margen de beneficio) y RES (resultado del ejercicio).

Ya vimos cómo el siguiente código nos aporta gráficamente una idea de la posible existencia de valores faltantes:

## Trabajando con multiples variables.  
  
# Localizando y descartando casos con missing values.  
  
vis\_miss(eolica\_100)



Podemos apreciar cómo existen varios casos con *missing values* en las variables objeto de estudio.

Para localizar los *missing values*, podemos recurrir al siguiente código. En él, hacemos una copia del *data frame* original, “eolica\_100”, para preservar su integridad. A esa copia la hemos llamado “muestra2”. Luego, hemos sometido a “muestra2” a un *filtro* para detectar los casos en los que no hay valor para alguna (o varias) de las variables analizadas. El operador **|** significa “o”. Posteriormente, hemos decidido eliminar esos casos. Para ello asignamos a “muestra2” el resultado de pasar un filtro en el que se eligen los casos que no tienen valores faltantes en ninguna de las variables. El operador **&** significa “y”:

## Trabajando con multiples variables.  
  
# Localizando y descartando casos con missing values.  
muestra2<- select(eolica\_100, everything())  
muestra2 %>% filter(is.na(RENECO) | is.na(ACTIVO) | is.na(MARGEN) | is.na(RES))%>%  
 select(RENECO, ACTIVO, MARGEN, RES)

## RENECO ACTIVO MARGEN RES  
## Viesgo Renovables SL. NA 269730.00 11.818 4609.000  
## Biovent Energia SA 4.551 183899.00 22.792 NA  
## Sargon Energias SLU NA 85745.00 -615.625 -2216.000  
## Parc Eolic Sant Antoni SL 1.361 69654.00 NA 668.000  
## Eolica La Brujula SA 7.295 42146.98 NA 2306.062  
## La Caldera Energia Burgos SL 2.643 NA 14.448 511.304

muestra2 <- muestra2 %>%  
 filter(! is.na(RENECO) & ! is.na(ACTIVO) & ! is.na(MARGEN) & ! is.na(RES))

El *data frame* “muestra2” contiene los mismos datos que “eolica\_100”, salvo los 6 casos con *missing values* (94).

Para la **detección de *outliers***, si las variables que entran en el análisis son numerosas, podría ser poco operativo estudiar las variables una a una. Una alternativa consiste en calcular la ***distancia de Mahalanobis*** de las variables del estudio, como **“resumen” del comportamiento de cada caso** en todas las variables del análisis, consideradas conjuntamente.

Así, primero vamos a calcular una columna más en el *data frame* “muestra2” con los valores de la *distancia de* *Mahalanobis* del conjunto de las 4 variables en cada uno de los casos (empresas eólicas). Esta columna o variable la denominaremos, por ejemplo, MAHALANOBIS*.* Para ello, se utiliza la función mutate() del paquete {deplyr}, y como argumento de esta la función mahalanobis(), en la que hay que especificar:

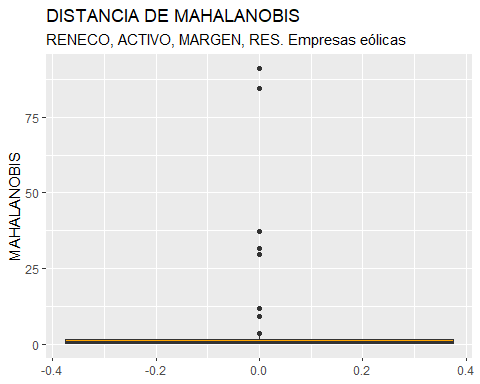
* Las columnas de “muestra2” para las que se van a calcular las distancias, unidas en una matriz interna con la función cbind().
* El vector de medias de las variables para las que se calcula la distancia (argumento center =). Las variables, aquí, se recogen con la función select() de {deplyr}, en donde el punto significa que las columnas pertenecen al *data frame* al que se llamó con el operador *pipe* %>%:
* La matriz de varianzas y covarianzas de las variables para las que se calcula la distancia (argumento cov =).

En definitiva, el código es:

# Identificando y descartando outliers con distancia de Mahalanobis.  
  
muestra2 <- muestra2 %>%  
 mutate (MAHALANOBIS = mahalanobis(cbind(RENECO, ACTIVO, MARGEN, RES),  
 center = colMeans(select(., RENECO, ACTIVO, MARGEN, RES)),  
 cov = cov(select(., RENECO, ACTIVO, MARGEN, RES))))

Posteriormente, se puede construir el diagrama de caja de la variable MAHALANOBIS, como cualquier otra variable:

ggplot(data = muestra2, map = (aes(y = MAHALANOBIS))) +  
 geom\_boxplot(fill = "orange") +  
 ggtitle("DISTANCIA DE MAHALANOBIS",  
 subtitle = "RENECO, ACTIVO, MARGEN, RES. Empresas eólicas ") +  
 ylab("MAHALANOBIS")



Se observa cómo existen varios casos *outliers*. Para saber de qué casos concretos se trata, se podrá ejecutar el código:

Q1M <- quantile (muestra2$MAHALANOBIS, c(0.25))  
Q3M <- quantile (muestra2$MAHALANOBIS, c(0.75))  
  
muestra2 %>%  
 filter(MAHALANOBIS > Q3M + 1.5\*IQR(MAHALANOBIS) |  
 MAHALANOBIS < Q1M - 1.5\*IQR(MAHALANOBIS))%>%  
 select(MAHALANOBIS, RENECO, ACTIVO, MARGEN, RES)

En la consola se obtendrá el listado:

## MAHALANOBIS RENECO ACTIVO  
## Holding De Negocios De GAS SL. 91.041690 5.264 13492812.00  
## Global Power Generation SA. 37.255573 1.393 2002458.00  
## Naturgy Renovables SLU 31.675561 1.959 1956869.00  
## Saeta Yield SA. 11.891027 0.360 796886.38  
## Molinos Del Ebro SA 29.589696 35.262 62114.37  
## Tarraco Eolica SA 3.600426 12.868 38102.00  
## WPD Parque Eolico Navillas SL. 84.589929 -0.416 35511.45  
## Brulles Eolica SL 3.599069 15.882 29722.58  
## Sierra De Selva SL 9.055155 21.761 27728.00  
##   
## MARGEN RES  
## Holding De Negocios De GAS SL. 91.152 727548.0000  
## Global Power Generation SA. 22.403 39995.0000  
## Naturgy Renovables SLU 20.442 42737.0000  
## Saeta Yield SA. 16.258 2084.4760  
## Molinos Del Ebro SA 41.821 17026.2569  
## Tarraco Eolica SA 400.899 4953.0000  
## WPD Parque Eolico Navillas SL. -2248.157 -110.9293  
## Brulles Eolica SL 47.227 3540.5693  
## Sierra De Selva SL 47.045 4525.0000

Si se opta por eliminar estos casos cara al análisis a aplicar posteriormente, se podrá crear un nuevo *data frame*, por ejemplo “muestra2\_so”, con el código siguiente:

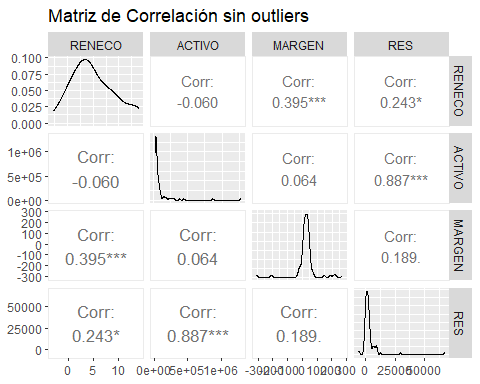
muestra2\_so <- muestra2 %>%  
 filter(MAHALANOBIS <= Q3M + 1.5\*IQR(MAHALANOBIS) &  
 MAHALANOBIS >= Q1M - 1.5\*IQR(MAHALANOBIS))

El *data frame* “muestra2\_so” será una réplica de “muestra2”, aunque sin incluir los casos detectados como atípicos o *outliers* (85 casos).

### 5.3.2 Correlación entre variables.

Cuando trabajamos con más de una variable, una característica muy importante viene dada por la intensidad con la que tales variables están relacionadas estadísticamente entre sí, es decir, el estudio de las correlaciones. Un modo atractivo y rápido de visualizar la **matriz de correlaciones** de las variables es a través de la función ggpairs() del paquete {GGally}. Para aplicar la función, hemos creado el *data frame* “temporal” con las variables (métricas) del estudio, que borramos tras general el gráfico:

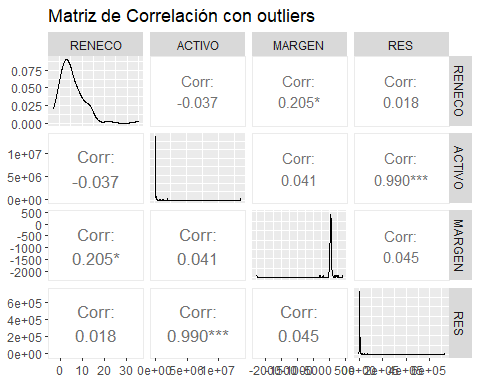
## Correlaciones entre variables.  
library (GGally)  
  
temporal <- muestra2\_so %>% select(RENECO, ACTIVO, MARGEN, RES)  
corr\_plot\_so <- ggpairs(temporal,   
 lower = list(continuous = wrap("cor",  
 size = 4.5,  
 method = "pearson",  
 stars = TRUE)),  
 title = "Matriz de Correlación sin outliers")  
rm(temporal)  
corr\_plot\_so



Un coeficiente de correlación puede tomar un valor entre -1 (fuerte relación, en sentido opuesto) a +1 (fuerte relación, en el mismo sentido). Como puede apreciarse en el gráfico, las variables ACTIVO y RES mantienen una relación muy intensa y en sentido positivo. Entre MARGEN y RENECO existe también una relación de intensidad destacable. En cambio, ACTIVO y MARGEN; y RENECO y ACTIVO apenas están estadísticamente relacionadas.

Antes de terminar la práctica, vamos a comparar las correlaciones anteriores con las que se dan si se incluyen los casos outliers en la muestra. Estas correlaciones se generarán con el mismo código; pero sustituyendo el *data frame* “muestra2\_so” por “muestra”:

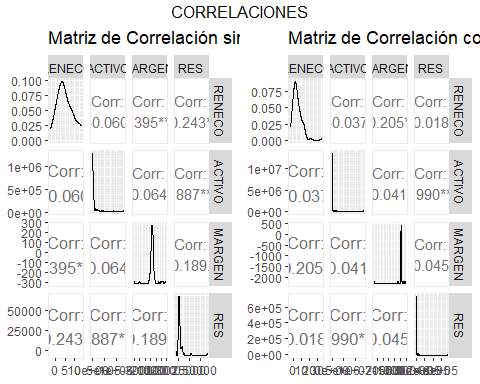
temporal <- muestra2 %>% select(RENECO, ACTIVO, MARGEN, RES)  
corr\_plot\_co <- ggpairs(temporal,   
 lower = list(continuous = wrap("cor",  
 size = 4.5,  
 method = "pearson",  
 stars = TRUE)),  
 title = "Matriz de Correlación con outliers")  
rm(temporal)  
corr\_plot\_co



Puede observarse cómo la presencia de outliers puede variar la relación entre las variables. Salvo el caso de la correlación entre ACTIVO y RES, que se fortalece; las correlaciones entre RENECO y MARGEN, y RENECO y RES se debilitan (en este último caso, pasa a no ser significativa ni tan siquiera para una significación de 0.1).

Un modo visual para poder comparar ambas figuras consiste en utilizar la función ggmatrix\_gtable() del paquete {GGally} para realizar “copias” en formato compatible con la función ggplot() del paquete {ggplo2}. Estas “copias” pueden integrarse en una sola figura utilizando la función grid.arrange() del paquete {gridExtra}. Nota: este paquete funciona de modo parecido a {patchwork}, y lo utilizamos porque este último paquete no funciona correctamente al intentar combinar las copias creadas en formato compatible con la función ggplot(). El código y resultado serán:

corr\_plot\_so\_gg <- ggmatrix\_gtable(corr\_plot\_so)  
corr\_plot\_co\_gg <- ggmatrix\_gtable(corr\_plot\_co)  
  
library (gridExtra)  
  
grid.arrange(corr\_plot\_so\_gg, corr\_plot\_co\_gg, ncol = 2, top = "CORRELACIONES")



## 5.4 Materiales para realizar las prácticas del capítulo.

En esta sección se muestran los links de acceso a los diferentes materiales (scripts, datos…) necesarios para llevar a cabo los contenidos prácticos del capítulo.

**Datos (en formato Microsoft (R) Excel (R)):**

* eolica\_100\_mv.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1f9KV5epNpBRM6v9sKPtgFDkzMlOo75Y2/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true))

**Scripts:**

* explora\_describe.R ([obtener aquí](https://drive.google.com/file/d/1VqZX81h15JZhF7KXXj3alLd08o5ywkX2/view?usp=sharing))

# 6 Componentes principales.

## 6.1 Introducción.

A veces, menos es más. Esta es la filosofía que subyace a las **técnicas de reducción de la dimensión** de la información.

Imaginemos una serie de casos (por ejemplo, las empresas de un sector económico) caracterizados por múltiples variables. Puede ocurrir que, paradójicamente, el contar con tantas variables haga difícil la caracterización de los casos. Esto ocurre cuando algunas de las variables aportan una información muy parecida sobre los mismos. Por ejemplo, es más difícil hacerse una idea del comportamiento global en el ámbito económico o financiero de un grupo de empresas si tenemos que atender a los valores que toman en un conjunto de 10 variables, que si solo tenemos que atender a un par de indicadores. Las técnicas de reducción de la dimensión de la información tratan, precisamente, de disminuir el número de variables necesarias para caracterizar un grupo de casos aprovechando la posibilidad de que las (múltiples) variables originales compartan información sobre los mismos. Es decir, la idea subyacente es pasar de un planteamiento basado en manejar muchas variables con información compartida o redundante (variables que “dicen lo mismo” sobre el comportamiento de los casos) a un planteamiento en el que hay **menos variables**, pero que **no comparten información** (variables que “dicen” cosas diferentes sobre el comportamiento de los casos). En este proceso es importante que la pérdida de información sea mínima, y que solo se pierda la información redundante o repetida.

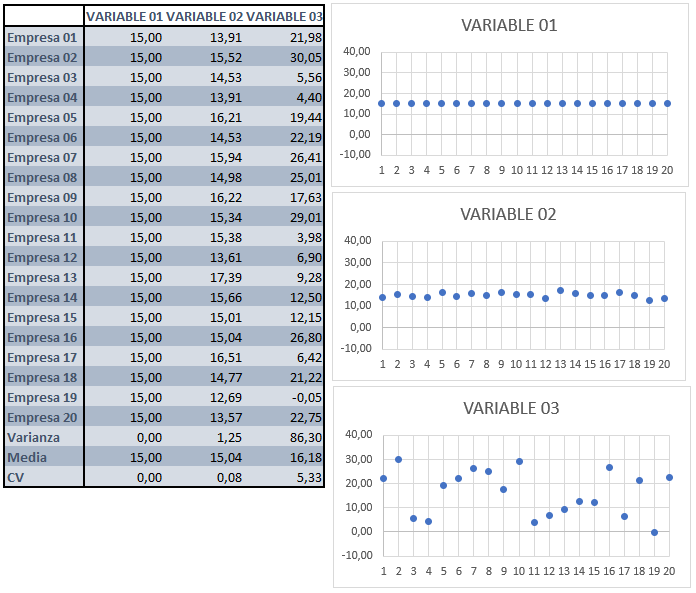
La principal técnica de reducción de la dimensión de la información es la de **componentes principales,** y es la que se expondrá y ejemplificará en el resto del capítulo. Pero antes, es preciso concretar la relación entre dos conceptos muy presentes en esta técnica: información y varianza.

## 6.2 Información y varianza.

En el apartado anterior hemos hablado de la posibilidad de que algunas variables compartan “información” sobre el comportamiento de los casos que constituyen nuestra muestra u objeto de estudio. Pero, ¿qué es, en este contexto, la “información”?

La ***información*** que una variable contiene sobre un conjunto de casos puede entenderse como su **capacidad para diferenciar a unos casos de otros**.

Observemos este ejemplo, en el que se representan los valores que toman un grupo de 20 empresas en 3 variables.



Información y Varianza

En la variable 1, todas las empresas toman el mismo valor. Por tanto, la capacidad que tiene la variable para distinguir a los casos (empresas), unos de otros, es nula. Eso es debido a que, en definitiva, esta variable no contiene información sobre el grupo de 20 empresas.

En la variable 2, existe cierta dispersión, aunque reducida, en los valores que adoptan los casos. Esto permite distinguir a unos de otros, aunque a veces con cierta dificultad. Por ejemplo, la empresa 17 se distingue del resto por ser la que tiene un valor (un poco) mayor. Aun así, como la dispersión es reducida, no se distinguen algunos casos de otros demasiado bien. En definitiva, la variable 2 contiene cierta cantidad de información sobre el conjunto de empresas de la muestra, aunque no demasiado grande.

Por último, la variable 3 muestra una dispersión considerablemente mayor que las otras dos variables. Existe un amplio abanico de valores que toman los diferentes casos (empresas). Esto hace que puedan diferenciarse con facilidad, en general, unos de otros. Esta variable posee, por tanto, una cantidad de información superior respecto a las empresas, ya que observando los valores que toman en la variable pueden diferenciarse con facilidad unas de otras.

Como conclusión, podemos establecer que cuanto mayor dispersión muestra una variable para un grupo de casos, mayor cantidad de información contiene sobre ellos, en el sentido de disfrutar de un mayor “poder” de diferenciación de unos casos respecto a otros.

Una medida de la dispersión de una variable usualmente utilizada es la **varianza**. Por tanto, en cierta manera, la varianza sirve para medir la cantidad de información que contiene la variable: a mayor varianza, mayor dispersión. Y a mayor dispersión, mayor cantidad de información.

En el ejemplo, puede observarse cómo la variable 3 es la que mayor varianza tiene, luego la variable 2, y la variable 1 tiene una varianza de 0 (y no posee información sobre las 20 empresas). Esta comparación de varianzas es válida siempre y cuando las tres variables estén expresadas en las mismas unidades, ya que la varianza es una medida de dispersión absoluta. Por ello, para poder comparar, hemos añadido también en el ejemplo una medida de dispersión relativa: el coeficiente de variación. Podemos comprobar cómo el mayor coeficiente de variación pertenece a la variable 3 (que es la que tiene una mayor cantidad de información), luego la variable 2 (que cuenta con menor cantidad de información), y por último la variable 1, con un coeficiente de 0 (no contiene información sobre las empresas).

## 6.3 Cálculo de componentes.

Vamos a considerar una serie de variables (en escala métrica) que caracterizan a la población de empresas de producción eléctrica mediante tecnología eólica en lo referente a la idea de “solidez del negocio”. Hemos seleccionado una muestra de 60 empresas. Nuestro objetivo es obtener una combinación lineal de estas variables (“componente principal”) que recoja la mayor parte de la suma de varianzas de las variables originales (“comunalidad”), de manera que pueda usarse como un indicador que resume o sintetiza el grado de solidez del negocio de cada empresa, con una pérdida mínima de información.

Las variables a partir de las cuales construiremos el indicador (“componente principal”) son:

* **RES:** Resultado del ejercicio.
* **FPIOS:** Fondos propios.
* **MARGEN:** Margen de beneficio.
* **SOLVENCIA:** Coeficiente de solvencia.

Vamos a suponer que trabajamos dentro de un **proyecto** que hemos creado previamente, de nombre “componentes”. Dentro de la carpeta del proyecto guardaremos el *script* llamado “componentes\_eolica.R”, y el archivo de Microsoft® Excel® llamado “eolica\_60.xlsx”. Si abrimos este último archivo, comprobaremos que se compone de tres hojas. La primera muestra un mensaje sobre el uso de los datos, la segunda recoge la descripción de las variables consideradas, y la tercera (hoja “Datos”) almacena los datos que debemos importar. Estos datos se corresponden con diferentes variables económico-financieras de 60 empresas productoras de electricidad mediante generación eólica.

Es muy importante observar que existen variables con datos faltantes (*missing values*). En concreto, podemos identificar estas faltas de dato por la existencia de celdas en blanco; pero también por la existencia de celdas con el texto “n.d.” (no dato). Así, tendremos que aplicar código adicional en el comando de importación de R para que estos casos queden correctamente recogidos como *NAs* (*not available*).

Cerraremos el archivo de Microsoft® Excel®, “[eolica\_60.xlsx](https://docs.google.com/spreadsheets/d/1G7yk22TShY1bYS1i9v4PfVi_HCkOG2ZO/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true)” y volveremos a RStudio. Después, abriremos nuestro *script* “[componentes\_eolica.R](https://drive.google.com/file/d/1d0ss6sh-jIAcrrlWSHHQI_CfRtiDqANX/view?usp=sharing)”. Este *script* contiene el programa que vamos a ejecutar en lel ejemplo.

La primera línea / instrucción en el *script* es:

rm(list = ls())

La instrucción tiene como objeto limpiar el *Environment* (memoria) de objetos de anteriores sesiones de trabajo. Para importar los datos almacenados en la hoja “Datos” del archivo “eolica\_60.xlsx”, ejecutaremos el código:

# DATOS  
  
library(readxl)  
datos <- read\_excel("eolica\_60.xlsx", sheet = "Datos", na = "n.d.")

Podemos observar cómo, en el *Environment,* ya aparece un objeto. Este objeto es una estructura de datos tipo *data frame*, se llama “datos” y contiene 12 columnas, una por cada una de las variables almacenadas en el archivo de Microsoft® Excel®. De estas variables, tres son de tipo cualitativo (atributos o factores), formadas por cadenas de caracteres: el nombre de la empresa (NOMBRE), el nombre de la sociedad matriz (grupo empresarial) a la que pertenece (MATRIZ), y el tamaño de dicho grupo de empresas (DIMENSION).

R ha considerado la primera columna de la hoja de Excel (NOMBRE) como una variable de tipo cualitativo o atributo. En realidad no es una variable; sino el nombre de los casos (empresas). Para evitar esto, podemos redefinir nuestro *data frame* diciéndole que esa primera columna contiene los nombres de los casos (filas):

# DATOS  
datos <- data.frame(datos, row.names = 1)

En la línea anterior hemos asignado al *data frame* “datos” los propios datos de “datos”; pero indicando que la primera columna no es una variable; sino el nombre de los casos o filas (empresas). Advertimos que ya no aparece NOMBRE como variable, y que en el *Environment* ya aparece el *data frame* “datos” con 60 observaciones, pero con 11 variables (una menos).

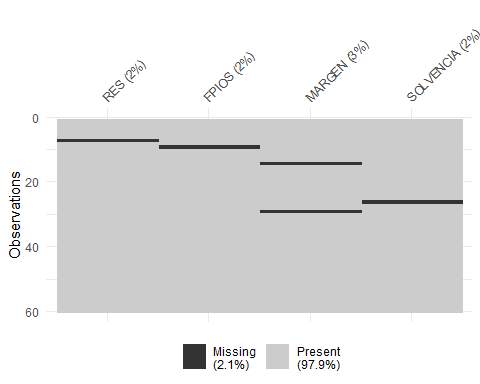
Como en el ejemplo se plantea construir un indicador basado en las 4 variables antes indicadas (RES, FPIOS, MARGEN y SOLVENCIA), crearemos un *data frame* con solo esas variables. Lo llamaremos, por ejemplo, “muestra”:

# Seleccionando variables metricas para el analisis.  
  
library(dplyr)  
muestra <- datos %>% select(RES, FPIOS, MARGEN, SOLVENCIA)  
summary (muestra)

## RES FPIOS MARGEN SOLVENCIA   
## Min. :-5661 Min. :-77533 Min. :-302.03 Min. :-40.74   
## 1st Qu.: 496 1st Qu.: 2323 1st Qu.: 11.70 1st Qu.: 5.94   
## Median : 1939 Median : 9727 Median : 27.64 Median : 16.88   
## Mean : 2699 Mean : 18801 Mean : 30.63 Mean : 28.12   
## 3rd Qu.: 3903 3rd Qu.: 26493 3rd Qu.: 39.59 3rd Qu.: 51.38   
## Max. :17026 Max. :177707 Max. : 400.90 Max. : 99.08   
## NA's :1 NA's :1 NA's :2 NA's :1

El siguiente paso será localizar los posibles ***missing values***, ya que para obtener componentes principales es necesario que todos los casos posean dato para todas las variables del análisis. Para tener una idea general, se puede utilizar la función vis\_miss() del paquete {visdat}, que localizará gráficamente los *missing values* de las diferentes variables, y calculará el porcentaje de casos que supone, con respecto al total de observaciones:

library(visdat)  
vis\_miss(muestra)



Del gráfico anterior se desprende que existen 5 *missing values* repartidos en las 4 variables del estudio. Para localizarlos, podemos filtrar nuestro *data frame* con las herramientas de {dplyr}:

muestra %>%  
 filter(is.na(RES) | is.na(FPIOS) | is.na(MARGEN) | is.na(SOLVENCIA)) %>%  
 select(RES, FPIOS, MARGEN, SOLVENCIA)

## RES FPIOS MARGEN SOLVENCIA  
## Biovent Energia SA NA 70033.0000 22.792 38.082  
## Eolica La Janda SL 9880.09100 NA 38.256 16.428  
## Parc Eolic Sant Antoni SL 668.00000 9727.0000 NA 13.964  
## WPD Parque Eolico El Poleo SL. -30.63754 520.6033 -11.121 NA  
## Eolica La Brujula SA 2306.06200 21694.7910 NA 51.474

Ante la existencia de *missing values*, se puede actuar de varios modos. Por ejemplo, **se puede intentar obtener por otro canal de información el conjunto de valores** que no están disponibles, **o recurrir a alguna estimación**. En caso de que esto sea difícil, se puede optar, simplemente, por **eliminar** estos casos, en especial cuando representan un porcentaje muy reducido respecto al total de casos. En nuestro ejemplo, supondremos que hemos optado por esta última vía, y eliminaremos estos casos con el código:

muestra <- muestra %>%  
 filter(! is.na(RES) & ! is.na(FPIOS) & ! is.na(MARGEN) & ! is.na(SOLVENCIA))

Verificamos en el *Environment* que el *data frame* “muestra” ha pasado a tener 53 casos.

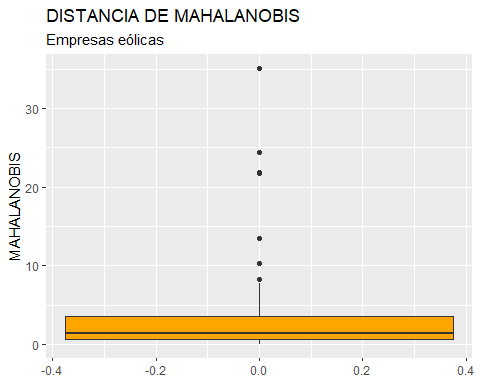
Por otro lado, la técnica de componentes principales **es muy sensible a la existencia de *outliers***. En consecuencia, deberán ser identificados y, en su caso, eliminados. Para realizar este proceso, y dado que en nuestro análisis contamos con 4 variables, primero “resumiremos” el valor que toman dichas variables para cada observación (empresa), mediante el cálculo de la *distancia de Mahalanobis*. De hecho, las distancias de los diferentes casos se almacenarán en una nueva columna o variable de nuestro *data frame,* a la que llamaremos MAHALANOBIS:

muestra <- muestra %>%  
 mutate (MAHALANOBIS = mahalanobis(cbind(RES, FPIOS, MARGEN, SOLVENCIA),  
 center = colMeans(select(., RES, FPIOS, MARGEN, SOLVENCIA)),  
 cov = cov(select(., RES, FPIOS, MARGEN, SOLVENCIA))))

Dentro de las funciones select() hay unos puntos. Recordemos que estos puntos deben ser añadidos cuando una función no es la primera del operador “*pipe*” (%>%), para indicar que las variables de los paréntesis hacen referencia al *data frame* “muestra” (o, en general, el objeto que fluye a través del “*pipe*”).

A continuación, construiremos un diagrama de caja de la variable MAHALANOBIS, como si fuera cualquier otra variable, a partir de la función ggplot() del paquete {ggplot2}:

library (ggplot2)  
ggplot(data = muestra, map = (aes(y = MAHALANOBIS))) +  
 geom\_boxplot(fill = "orange") +  
 ggtitle("DISTANCIA DE MAHALANOBIS", subtitle = "Empresas eólicas") +  
 ylab("MAHALANOBIS")



En el gráfico se observa que existen, por encima de la caja, varios *outliers*. Para identificarlos de modo concreto, hemos de calcular los cuartiles primero y tercero de la variable MAHALANOBIS y pasar el correspondiente filtro:

Q1M <- quantile (muestra$MAHALANOBIS, c(0.25))  
Q3M <- quantile (muestra$MAHALANOBIS, c(0.75))  
muestra %>%  
 filter(MAHALANOBIS > Q3M + 1.5\*IQR(MAHALANOBIS) |  
 MAHALANOBIS < Q1M - 1.5\*IQR(MAHALANOBIS))%>%  
 select(MAHALANOBIS)

## MAHALANOBIS  
## Viesgo Renovables SL. 35.05290  
## Guzman Energia SL 10.31024  
## WPD Wind Investment SL. 21.69461  
## Molinos Del Ebro SA 24.38816  
## Tarraco Eolica SA 21.91118  
## WPD Parque Eolico Las Panaderas SL. 8.21836  
## Eolica Navarra SL 13.41478

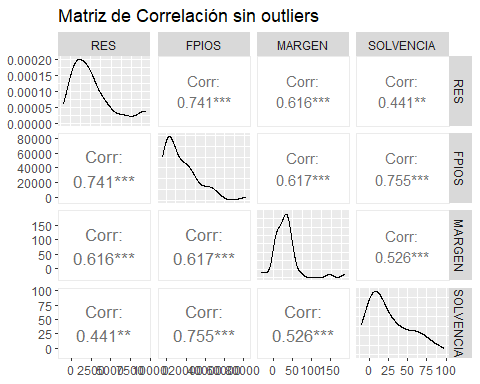
Si, tras el estudio de los valores que toman las variables originales en estos casos, se decide eliminarlos, el código será:

muestra\_so <- muestra %>%  
 filter(MAHALANOBIS <= Q3M + 1.5\*IQR(MAHALANOBIS) &  
 MAHALANOBIS >= Q1M - 1.5\*IQR(MAHALANOBIS))  
muestra\_so <- muestra\_so %>% select(-MAHALANOBIS)

Se ha creado un nuevo *data frame* llamado “muestra\_so” con los casos que **no** son *outliers* (y que no contienen *missing values*), y se ha eliminado la variable MAHALANOBIS, puesto que su única utilidad era la de localizar y filtrar los *outliers*. Con este *data frame* “muestra\_so” es con el que se procederá al cálculo de las componentes.

La condición previa para el cálculo de componentes es que las variables originales del análisis contengan información redundante, es decir, que en buena medida aporten una “misma información” sobre los casos (empresas). Esto se verifica con la existencia de altas correlaciones entre las variables (al menos, entre algunas de ellas). Por tanto, hemos de calcular la matriz de correlaciones correspondiente. Un modo gráfico visualmente efectivo es utilizar las posibilidades que nos ofrece el paquete {GGally} mediante la función ggpairs():

# Correlaciones.  
  
library (GGally)  
  
corr\_plot\_so <- ggpairs(muestra\_so,   
 lower = list(continuous = wrap("cor",  
 size = 4.5,  
 method = "pearson",  
 stars = TRUE)),  
 title = "Matriz de Correlación sin outliers")  
corr\_plot\_so



Puede apreciarse cómo existen altas correlaciones (en valor absoluto) entre todas las variables. Por tanto, tiene sentido hacer un análisis de componentes principales, ya que hay variables que parecen **compartir información**.

La obtención de las componentes se va a realizar mediante la función prcomp(). Es conveniente que activemos el argumento scale = con “T” (*true*) para que las variables originales sean consideradas en sus **versiones tipificadas**. Vamos a asignar los resultados a un objeto de nombre, por ejemplo, “componentes”. Por último, guardaremos el summary() o resumen de los resultados con un nombre provisional, por ejemplo, “temporal”. El código es el siguiente:

# Obtencion de componentes.  
  
componentes <- prcomp (muestra\_so, scale=T)  
temporal <- summary (componentes)  
temporal

## Importance of components:  
## PC1 PC2 PC3 PC4  
## Standard deviation 1.6903 0.7646 0.6487 0.3709  
## Proportion of Variance 0.7143 0.1462 0.1052 0.0344  
## Cumulative Proportion 0.7143 0.8604 0.9656 1.0000

La “*Standard deviation*” es la raíz cuadrada de los autovalores asociados a cada componente. “*Proportion of Variance*” nos dice la proporción de la suma de varianzas de las variables originales (*comunalidad*) recogida por cada componente, proporción que se acumula en “*Cumulative Proportion*”. Nótese que las componentes aparecen ordenadas de más a menos importantes en función de la cantidad de varianza que capturan. En este caso, la primera componente acumula más del 71% de la varianza (comportamiento) de las variables originales. Por tanto, esta primera componente resume bastante bien la información que las 4 variables originales contienen sobre los casos (empresas).

Si el elemento “importance” del summary() o resumen “temporal” lo convertimos en un *data frame*, por ejemplo “summary\_df”, podremos presentar los resultados por medio de una tabla estéticamente más atractiva, a partir de la función kable() del paquete {knitr}, y las funciones complementarias del paquete {kableExtra}:

# Convertir el resumen en un data frame  
  
summary\_df <- as.data.frame(temporal$importance)  
summary\_df <- t(summary\_df) # Transponer para mejor visualización  
rm (temporal)  
  
# Crear la tabla con kable y personalizarla con kableExtra  
  
library (knitr)  
library (kableExtra)  
knitr.table.format = "html"  
  
summary\_df %>%  
kable(caption = "Resumen de Componentes",  
 col.names = c("Desviación típica","Proporción de varianza (comunalidad)",  
 "Proporción de varianza (comunalidad) acumulada"),  
 format.args = list(decimal.mark = ".", digits = 4)) %>%  
 kable\_styling(bootstrap\_options = c("striped", "bordered", "condensed"),  
 full\_width = F,   
 position = "center") %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:(nrow(summary\_df)), bold= F, align = "c") %>%  
 column\_spec(1, bold = TRUE, extra\_css = "text-align: left;")

Table 6.1: (#tab:unnamed-chunk-192)Resumen de Componentes

Desviación típica

Proporción de varianza (comunalidad)

Proporción de varianza (comunalidad) acumulada

PC1

1.6903

0.7143

0.7143

PC2

0.7646

0.1462

0.8604

PC3

0.6487

0.1052

0.9656

PC4

0.3709

0.0344

1.0000

Los coeficientes o **cargas** de cada componente se obtienen pidiendo a nuestro objeto “componentes” el elemento “rotation”. Estas cargas las vamos a guardar en un nuevo objeto que llamaremos, por ejemplo, “cargas”, que presentaremos mediante una pequeña tabla diseñada con la función kable() del paquete {knitr} y otras funciones del paquete {kableExtra}:

# Cargas de cada componente.  
  
cargas <- componentes$rotation  
cargas %>%  
 kable(caption = "Cargas de las componentes obtenidas",  
 format.args = list(decimal.mark = ".", digits = 4)) %>%  
 kable\_styling(full\_width = F, bootstrap\_options = "striped",  
 "bordered", "condensed",  
 position = "center") %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:(nrow(cargas)), bold= F, align = "c") %>%  
 column\_spec(1, bold = TRUE, extra\_css = "text-align: left;")

(#tab:unnamed-chunk-194)(#tab:unnamed-chunk-194)Cargas de las componentes obtenidas

PC1

PC2

PC3

PC4

RES

0.4915

0.5578

0.48460

0.46094

FPIOS

0.5491

-0.1554

0.33515

-0.74971

MARGEN

0.4803

0.3406

-0.80447

-0.07849

SOLVENCIA

0.4758

-0.7407

-0.07528

0.46830

A partir de las cargas se pueden explicitar las ecuaciones correspondientes a cada componente. Por ejemplo, para la primera componente, la ecuación será:

Puede apreciarse que, en cuanto a la primera componente, que es la que especialmente nos interesa como “indicador” de la “solidez del negocio” en el caso de las empresas eólicas seleccionadas, las 4 cargas tienen signo positivo, lo que implica que, cuanto mayores sean los valores de una empresa en las varibles RES (resultado), FPIOS (fondos propios), MARGEN (margen de beneficio) y SOLVENCIA (coeficiente de solvencia), mayor será el valor ndel indicador y, por tanto, la solidez del negocio. Además, como las variables fueron tipificadas, el valor de las cargas son comparables. De este modo, vemos cómo, dentro de la primera componente, que es la que adoptamos como “indicador de solidez”, la mayor importancia la tiene el valor de los fondos propios de la empresa, seguido del resultado del ejercicio, el margen y la solvencia.

## 6.4 Retención de componentes principales.

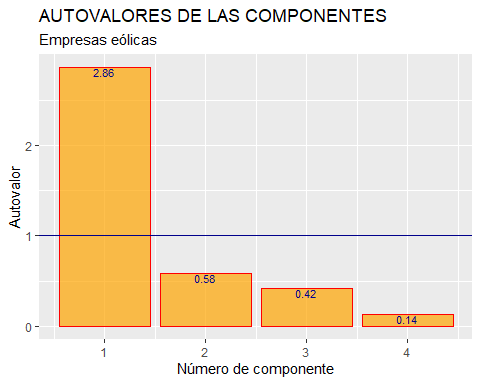
La etapa de retención de componentes principales consiste en decidir cuántas de las componentes generadas (recordemos que, en un principio, se calculan tantas componentes como variable originales) consideramos que resumen de un modo aceptable la información contenida en las variables originales. Estas **componentes “retenidas” se convertirán en las componentes principales**.

La primera componente siempre es retenida y, por tanto, es una “componente principal”. El resto, que van capturando proporciones cada vez más pequeñas de la varianza común de las variables originales (comunalidad), podrán o no retenerse; aunque, siempre, la retención de una componente implica que se han retenido todas las anteriores.

Hay varios procedimientos o criterios para tomar la decisión de cuántas componentes retener. Uno de ellos, comúnmente aplicado, es el de **retener aquellas componentes cuyo autovalor es mayor que 1** (suponiendo que se ha trabajado con las variables en sus versiones tipificadas).

Los autovalores, como ya vimos, son el cuadrado del elemento “Standard deviation” (sdev) del objeto “componentes” que hemos generado a partir de la función prcomp(). Hemos creado un *data frame* con estos autovalores calculados (y su orden, al que hemos llamado variable o columna “orden”, y que es un vector de números enteros consecutivos que va desde uno hasta número de variables originales o de componentes) y los hemos dispuesto en un gráfico de barras:

# Determinacion Componentes a retener.  
  
# Criterio del Autovalor mayor que 1.  
  
orden <- c(1:ncol(muestra\_so))  
autovalor <- componentes$sdev^2  
autovalores <- data.frame(orden, autovalor)  
  
autograph <- ggplot(data = autovalores,  
 map = (aes(x = orden, y = autovalor))) +  
 geom\_bar(stat = "identity",  
 colour = "red",  
 fill = "orange",  
 alpha = 0.7) +  
 scale\_x\_continuous(breaks=c(1:nrow(autovalores)))+  
 geom\_hline(yintercept = 1, colour = "dark blue") +  
 geom\_text(aes(label = round(autovalor,2)),  
 vjust = 1, colour = "dark blue", size = 3) +  
 ggtitle("AUTOVALORES DE LAS COMPONENTES",  
 subtitle = "Empresas eólicas") +  
 xlab ("Número de componente") +  
 ylab("Autovalor")  
  
autograph



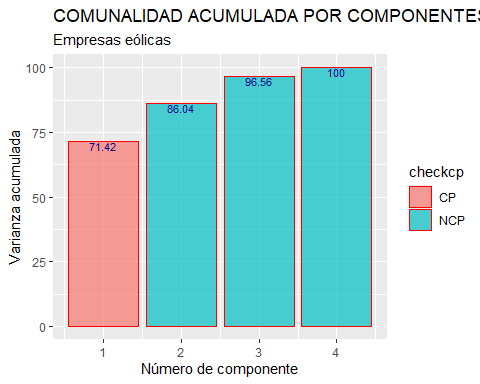
Respecto al gráfico, conviene recordar que, al ser un gráfico de barras, si no se quieren representar las frecuencias sino los valores que toma una variable (en este caso, “autovalor”) para cada valor de la otra variable (en este caso, “orden”); en el geom\_bar() habrá que añadir el argumento stat = con el valor “identity”. Además, se utiliza el elemento scale\_x\_continuous() para pesonalizar la escala del eje x, y que se divida dicho eje en tantos tramos como componentes hay.

En el gráfico obtenido se advierte que solo el primer autovalor es mayor que 1, por lo que solo se retendrá la primera componente, que será la única **componente principal.** Nuestro objetivo era, de todos modos, obtener un único indicador (de “solidez del negocio”) que resuma la información contenida en las cuatro variables RES, FPIOS, MARGEN y SOLVENCIA, por lo que, aunque hubiera habido más componentes con autovalor mayor que uno, solo hubiéramos seleccionado la primera. No obstante, este resultado es positivo para nuestros intereses, ya que reafirma la idea de que únicamente con la primera componente se recoge una gran proporción de la *comunalidad* o varianza conjunta (comportamiento) de las 4 variables, luego es un buen resumen global de las mismas. Esto ya se vio anteriormente al hacer summary (componentes), aunque se puede representar gráficamente con el siguiente código:

# Determinar si cada autovalor es mayor o igual a 1  
  
autovalores <- autovalores %>%  
 mutate(variacum = 100\*(cumsum((autovalor/nrow(autovalores)))))  
checkcp <- ifelse(autovalores$autovalor >= 1, "CP", "NCP")  
checkcp

## [1] "CP" "NCP" "NCP" "NCP"

vacumgraph <- ggplot(data = autovalores, map = (aes(x = orden,  
 y = variacum,  
 fill = checkcp))) +  
 geom\_bar(stat = "identity", colour = "red", alpha = 0.7) +  
 scale\_x\_continuous(breaks=c(1:nrow(autovalores)))+  
 geom\_text(aes(label = round(variacum,2)), vjust = 1,  
 colour = "dark blue", size = 3) +  
 ggtitle("COMUNALIDAD ACUMULADA POR COMPONENTES",  
 subtitle = "Empresas eólicas") +  
 xlab ("Número de componente") +  
 ylab("Varianza acumulada")  
vacumgraph

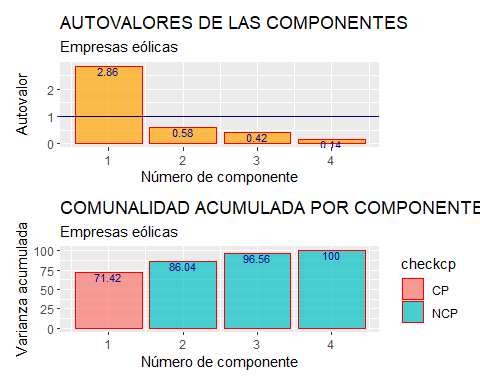


Para obtener el gráfico anterior, se comienza añadiendo al *data frame* “autovalores” una columna o variable que es la s*uma acumulada del porcentaje de* *comunalidad* recogido por las sucesivas componentes, que están ordenadas de mayor a menor autovalor. Para calcular el porcentaje, se usa la función cumsum(), y se tiene en cuenta que, como las variables fueron tipificadas para calcular las componentes, la *comunalidad*, que coincide con la suma de las varianzas de las componentes (autovalores), es igual al número de variables o componentes (valor que toma la función nrow()).

Después, se ha creado un vector que contiene tantos elementos como variables o componentes hay en el análisis (vector “checkcp”). Con la función condicional ifelse() se consigue que los elementos de “checkcp” sean “CP” o “NCP” según los correspondientes autovalores sean mayores o no que 1. Finalmente, según sea el valor de cada elemento de “checkcp”, las barras del gráfico se colorearán de uno u otro modo.

Posteriormente, mediante el paquete {patchwork}, se han unido los dos gráficos creados en esta fase, poniendo uno debajo del otro:

library (patchwork)  
autograph / vacumgraph



## 6.5 Puntuaciones de los casos (scores)

Para obtener las puntuaciones de cada caso (empresa) en el indicador de “solidez del negocio” (y que es nuestra componente principal, que a su vez coincide con la primera componente), simplemente debemos tener en cuenta que tales puntuaciones están guardadas en la matriz “x” del objeto prcomp() creado. Vamos a renombrar a las primera columna (componente) de esta matriz como “scores” y vamos a ver las puntuaciones de las empresas, que volcaremos en una tabla, junto al valor que toman en las variables originales, recolocando las filas (empresas) de mayor a menor valor de la puntuación (lo que se consigue mediante la función arrange() del paquete {dplyr}:

# Scores.  
  
scores <- componentes$x[,1] #tantas columnas como componentes retenidas  
scores\_df <- as.data.frame(scores)  
scores\_df <- cbind(scores\_df,muestra\_so)  
scores\_df %>%  
 arrange(desc(scores)) %>%  
 kable(caption = "Puntuaciones de las componentes obtenidas",  
 col.names = c("Empresa", "Puntuación", "Resultado", "F. Propios",  
 "Margen", "Solvencia"),  
 format.args = list(decimal.mark = ".", digits = 4)) %>%  
 kable\_styling(full\_width = F, bootstrap\_options = "striped",  
 "bordered", "condensed",  
 position = "center", font\_size = 12) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:(nrow(scores\_df)), bold= F, align = "c") %>%  
 column\_spec(1, bold = TRUE, extra\_css = "text-align: left;")

(#tab:unnamed-chunk-199)(#tab:unnamed-chunk-199)Puntuaciones de las componentes obtenidas

Empresa

Puntuación

Resultado

F. Propios

Margen

Solvencia

Eolia Gregal De Inversiones SA.

6.34608

9420.00

81778.0

186.208

79.196

Elecdey SL

3.55316

5539.03

41042.3

144.924

65.208

CYL Energia Eolica SL

2.87125

9401.38

51821.6

37.795

51.290

Esquilvent SL

2.38469

9010.21

48769.1

39.476

30.938

Disa Duna SL.

2.35060

2594.40

42283.1

55.953

98.303

Parques Eolicos San Lorenzo SL

2.11047

4311.98

37396.2

36.810

89.806

Insular De Aguas De Lanzarote SA

1.86856

4159.91

28249.3

57.138

76.913

Sierra De Selva SL

1.42059

4525.00

19555.0

47.045

70.524

Naturgy Wind, S.L.

1.31880

8500.00

28418.0

39.575

10.388

Parques Eolicos De Cerrato SL

1.26532

6541.00

22817.0

44.755

34.878

Eolica De Rubio SL

0.91962

2693.00

25486.0

34.607

60.688

Brulles Eolica SL

0.88594

3540.57

16432.0

47.227

55.284

Sistemas Energeticos Valle De Sedano SA

0.88563

897.00

45733.0

4.327

66.896

Acciona Eolica Del Levante SL

0.65681

6853.00

21769.0

27.520

11.557

BON Vent De Corbera S. L. EN Constitucion

0.65640

3261.47

23095.6

36.607

41.453

Molinos De La Rioja Sociedad Anonima

0.56574

2371.00

15787.0

42.017

54.789

Desarrollo De Energias Renovables De La Rioja Sociedad Anonima

0.48954

2145.00

17441.0

27.297

62.005

Eolica Campollano SA

0.48574

2592.20

27217.6

25.327

40.630

Bajoz Eolica SL

0.39866

2620.34

9641.3

70.476

29.644

Parc Eolic De Vilalba Dels Arcs SL

0.02524

2454.07

9438.7

48.378

28.085

Empordavent SL

-0.04223

2582.00

15176.0

32.635

25.295

Al-Andalus Wind Power SL

-0.07062

4403.21

21466.1

12.582

8.591

Eolica Fontesilva SL

-0.07618

1584.33

15524.1

27.756

37.786

Eolica 2000 SL

-0.56288

1713.00

5765.0

37.852

16.878

Fergo Galicia Vento -P E Mondoñedo SL

-0.64177

2288.50

4159.3

30.405

14.925

Renovables Castilla La Mancha S.A.

-0.68969

1847.03

2902.4

39.595

11.635

Comiolica SL

-0.73350

1938.69

4404.7

29.295

13.845

Drago Renovables SL

-0.78124

1000.92

7735.3

20.826

22.401

Parque Eolico La Boga SL.

-0.79622

11.94

29316.8

1.001

9.646

Parque Eolico Santa Catalina SL

-0.81797

3645.28

-1664.8

31.780

-1.126

Parque Eolico Valcaire SL

-0.84931

1152.72

2173.0

45.881

5.981

Energias Renovables De Peñanebina SL

-0.88259

983.00

5231.0

25.939

16.901

Expertise EN Energias Renovables Eolica Y Fotovoltaica De Canarias SL

-0.96396

52.30

4916.5

34.569

15.809

Parque Eolico Tesosanto SL

-1.10299

1706.13

2842.2

21.008

4.249

Eolica Del Guadiana SL

-1.20211

1571.57

4430.2

9.341

6.671

Suresa Retama S.L.

-1.24341

-78.00

12892.0

-29.295

39.249

Elecdey Palencia S.L.

-1.25359

987.27

1581.5

22.016

4.760

Eolica Mirasierra SL

-1.36573

1123.48

995.8

17.064

1.780

Helios Patrimonial 1 SL

-1.50762

237.85

3468.1

5.379

8.411

Sistemes Energetics Conesa I S.L.

-1.56230

878.00

-988.0

16.649

-3.193

Inversiones Fotovoltaicas Mallorquinas SL.

-1.57826

272.75

930.1

11.661

3.235

Helios Patrimonial 2 SL

-1.60027

190.12

2472.1

4.315

6.145

Fotovoltaica Los Navalmorales SL

-1.60453

143.00

966.0

12.228

2.581

Energias Naturales La Calzada SL

-1.69272

754.00

-1983.0

15.561

-6.834

Parque Eolico El Moral SL

-1.74511

267.00

-310.0

7.935

-1.240

Eolica De Medinaceli SL

-1.91245

324.00

-2854.0

4.302

-4.142

Wigep Andalucia SA

-2.05070

296.00

-3762.0

4.611

-10.634

Parque Eolico Energia Puertosol SL.

-2.12892

-945.38

2488.6

-17.969

5.899

Las puntuaciones de los casos en nuestro indicador (componente principal) pueden integrarse en un *data frame* y ser utilizadas como cualquier otra variable en un análisis multivariante, sabiendo que este indicador asume gran parte de la información que, como caracterización de las distintas empresas, contenían las 4 variables originales.

## 6.6 Materiales para realizar las prácticas del capítulo.

En esta sección se muestran los links de acceso a los diferentes materiales (*scripts*, datos…) necesarios para llevar a cabo los contenidos prácticos del capítulo.

**Datos (en formato Microsoft (R) Excel (R)):**

* eolica\_60.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1G7yk22TShY1bYS1i9v4PfVi_HCkOG2ZO/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true))

**Scripts:**

* componentes\_eolica.R ([obtener aquí](https://drive.google.com/file/d/1d0ss6sh-jIAcrrlWSHHQI_CfRtiDqANX/view?usp=sharing))

# 7 Análisis Clúster.

## 7.1 Introducción.

El análisis de conglomerados o análisis clúster (AC) trata de clasificar individuos o casos asignándolos a grupos homogéneos, de manera que:

* Cada grupo, conglomerado o clúster contenga a **los casos más parecidos** entre sí, en términos de una serie de variables (**variables clasificadoras**).
* **Los grupos** contengan casos que, en general, **sean muy diferentes** a los casos del resto de grupos, de acuerdo con las variables consideradas.

En general, el proceso de determinación de los grupos, conglomerados o clústeres de casos es el siguiente:

* Se parte de un conjunto de **n** casos, y para cada uno de ellos se cuenta con el valor de **m** variables clasificadoras.
* Se establece una **medida de distancia** que cuantifica lo que dos casos se parecen, **considerando en conjunto** los valores que poseen para las variables clasificadoras.
* Se crean los grupos, conglomerados o clústeres con los casos que poseen entre sí una **menor distancia**. Existen dos **enfoques** principales a la hora de crear los grupos de casos a partir de las distancias observadas entre los casos: los *métodos jerárquicos* y los *métodos no-jerárquicos*.
* Finalmente, se **caracterizan** los grupos, conglomerados o clíusteres obtenidos, y se comparan unos con otros para extraer conclusiones.

En lo que respecta a la medida de **distancia** entre los casos, la medida más habitual es la **distancia euclídea**. Así, la distancia euclídea entre dos caso, i e i’, para las m variables clasificadoras x, será:

Esta distancia es muy sensible a la escala de las variables clasificadoras. Para evitar este inconveniente, se trabaja con las variables previamente **tipificadas**.

## 7.2 Métodos de agrupación jerárquicos.

Como se acaba de comentar, existen dos enfoques fundamentales de realizar el análisis clúster, dependiendo de cómo son los métodos de agrupación de los casos (y grupos de casos): el enfoque de los métodos jerárquicos, y el enfoque que reúne a los métodos no-jerárquicos.

Ambos enfoques tienen sus ventajas e inconvenientes, y pueden adaptarse mejor a cada problema concreto. Es importante seleccionar un buen método de agrupación, puesto que pueden proporcionar soluciones muy diferentes entre sí.

En los **métodos jerárquicos,** se van formando sucesivamente grupos como agrupación de otros grupos precedentes, hasta llegar a un único grupo que recoge a todos los individuos; tomando el proceso una **estructura piramidal** (también existen métodos jerárquicos descendientes, que parten de un único grupo que contiene a todos los casos, para acabar el n grupos de un solo caso, aunque son menos frecuentes).

Estos métodos suelen aplicarse cuando hay un número reducido de casos. También, cuando nuestro objetivo pasa por crear **grupos que recojan a todos los casos**, más que definir simplemente tipologías más o menos homogéneas de casos (lo que se obtiene caracterizando los grupos obtenidos). Es decir, cuando se incluyen en el análisis a todos los individuos, incluidos los *outliers*. De hecho, estos métodos pueden emplearse, de por sí, como técnicas de localización de *outliers*. Por último, también se suelen emplearse cuando se desconoce a priori el número de grupos, conglomerados o clústeres a formar.

Entre los métodos jerárquicos de agrupación más extendidos, figuran los siguientes:

* **Método del vecino más cercano (single linkage):** la distancia que se considera entre grupos es la distancia entre sus elementos más próximos.
* **Método del vecino más lejano (complete linkage):** la distancia que se considera entre grupos es la distancia entre sus elementos más lejanos.
* **Método de Ward (Ward method):** se unen los grupos que dan lugar a otro grupo cuyos casos tienen una menor suma de los cuadrados de sus distancias respecto al centro de dicho grupo (menor varianza).
* **Otros métodos:** vinculación intergrupos (average linkage between groups), vinculación intragrupos (whithin group)…

De entre ellos, ¿cuál elegir?

La cuestión no es fácil de resolver, y no tiene por qué tener una única respuesta. Por otro lado, cada método proporciona soluciones que pueden variar mucho entre sí. Una estrategia puede pasar por probar con varios métodos y se seleccionar la solución que parezca más coherente desde el punto de vista teórico, y estable desde el punto de vista empírico.

En la práctica, uno de los métodos más utilizados es el **método de Ward**, porque proporciona grupos muy homogéneos, ya que se basa en la minimización de la varianza o dispersión de los elementos que componen cada grupo con respecto a su centro de gravedad o **centroide.** Precisamente, este método será aplicado en el ejemplo práctico que desarrollaremos en R a continuación.

Vamos a considerar una serie de 4 variables que caracterizan a un grupo de 25 empresas de producción eléctrica mediante tecnología eólica. Nuestro objetivo es segmentar este conjunto de empresas, haciendo grupos homogéneos (conglomerados), y caracterizando a dichos grupos. Las variables clasificadoras son: el resultado del ejercicio (RES), el margen de beneficio (MARGEN), los fondos propios (FPIOS) y la solvencia (SOLVENCIA).

Dado que son pocos los casos (empresas) a segmentar, vamos a utilizar un métodos jerárquico de agrupación de casos. En concreto, utilizaremos el **método de Ward**. Así, comenzaremos creando un proyecto de RStudio donde trabajar. Por ejemplo, un proyecto llamado “cluster”. Vamos a ir a la carpeta del proyecto y vamos a guardar en ella los dos archivos de esta práctica: un archivo de Microsoft® Excel® llamado “[eolica\_25.xlsx](https://docs.google.com/spreadsheets/d/1iwD8-e19IM6n_ZykP49i69_f_dLpzosx/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true)” y un *script* denominado “[cluster\_eolica.R](https://drive.google.com/file/d/1pxGjprMpfP_yqNsaJ-oWKJtfVnSbVZUh/view?usp=sharing)”. Si abrimos el archivo de Microsoft® Excel®, comprobaremos que se compone de tres hojas. La primera muestra un aviso sobre la utilización de los datos, la segunda el nombre y definición de las variables disponibles, y la tercera (hoja “Datos”) guarda los datos que debemos importar desde R-Studio. Estos datos se corresponden con diferentes variables económico-financieras de 25 empresas productoras de electricidad mediante generación eólica.

La primera parte del *script* se ocupa de importar los datos de la hoja de cálculo, limpiando previamente la memoria (*Environment*) de posibles objetos almacenados en sesiones anteriores:

rm(list = ls())  
  
# DATOS  
  
# Importando  
  
library(readxl)  
eolicos <- read\_excel("eolica\_25.xlsx", sheet = "Datos",  
 na = c("n.d.", "s.d."))  
eolicos <- data.frame(eolicos, row.names = 1)  
summary (eolicos)

## RES ACTIVO FPIOS RENECO   
## Min. : -5662 Min. : 85745 Min. : -77533 Min. :-2.813   
## 1st Qu.: 2385 1st Qu.: 147743 1st Qu.: 28418 1st Qu.: 1.676   
## Median : 7121 Median : 230339 Median : 70033 Median : 3.949   
## Mean : 41118 Mean : 965938 Mean : 461020 Mean : 3.694   
## 3rd Qu.: 10615 3rd Qu.: 443467 3rd Qu.: 177707 3rd Qu.: 5.125   
## Max. :727548 Max. :13492812 Max. :6904824 Max. :12.406   
## NA's :1 NA's :2   
##   
## RENFIN LIQUIDEZ ENDEUDA MARGEN   
## Min. :-359.7730 Min. :0.078 Min. : 0.917 Min. :-615.625   
## 1st Qu.: 2.7280 1st Qu.:0.697 1st Qu.: 43.039 1st Qu.: 12.582   
## Median : 11.3380 Median :1.184 Median : 62.903 Median : 22.792   
## Mean : 0.3039 Mean :1.354 Mean : 63.599 Mean : 7.557   
## 3rd Qu.: 24.6330 3rd Qu.:1.550 3rd Qu.: 88.442 3rd Qu.: 39.476   
## Max. : 52.2610 Max. :5.330 Max. :140.745 Max. : 223.956   
##   
## SOLVENCIA APALANCA MATRIZ DIMENSION   
## Min. :-40.74 Min. :-6265.50 Length:25 Length:25   
## 1st Qu.: 11.56 1st Qu.: 13.33 Class :character Class :character   
## Median : 37.10 Median : 110.40 Mode :character Mode :character   
## Mean : 36.40 Mean : -39.90   
## 3rd Qu.: 56.96 3rd Qu.: 480.12   
## Max. : 99.08 Max. : 1019.62

Podemos observar cómo, en el *Environment,* ya aparece un objeto. Este objeto es una estructura de datos tipo *data frame*, se llama “eolicos” y contiene 12 columnas, una por cada una de las variables almacenadas en el archivo de Microsoft® Excel®. De estas variables, dos son de tipo cualitativo (atributos o factores), formadas por cadenas de caracteres: el nombre de la sociedad matriz (grupo empresarial) a la que pertenece (MATRIZ). En realidad, R consideró la primera columna de la hoja de Excel (NOMBRE) como una variable de tipo cualitativo o atributo (por lo que había 13 columnas); pero, al no ser una variable, sino el nombre de los casos o filas (empresas), redefinimos nuestro *data frame* diciéndole que esa primera columna contenía los nombres de los casos (filas).

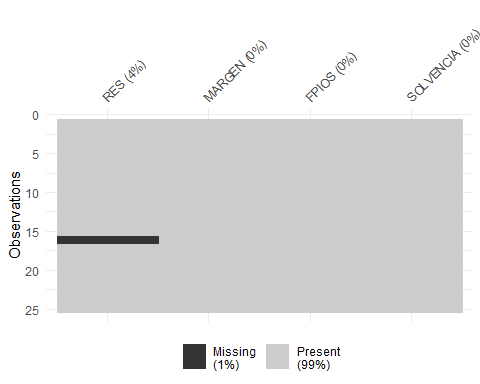
En el análisis solo vamos a considerar como variables clasificadoras para construir los grupos o conglomerados las variables RES, MARGEN, FPIOS y SOLVENCIA. Por ello, crearemos con ellas un nuevo *data frame* llamado “originales” a partir de la función select() del paquete {dplyr}:

# Seleccionando variables clasificadoras para el analisis.  
  
library(dplyr)  
originales <- eolicos %>% select(RES, MARGEN, FPIOS, SOLVENCIA)  
summary (originales)

## RES MARGEN FPIOS SOLVENCIA   
## Min. : -5662 Min. :-615.625 Min. : -77533 Min. :-40.74   
## 1st Qu.: 2385 1st Qu.: 12.582 1st Qu.: 28418 1st Qu.: 11.56   
## Median : 7121 Median : 22.792 Median : 70033 Median : 37.10   
## Mean : 41118 Mean : 7.557 Mean : 461020 Mean : 36.40   
## 3rd Qu.: 10615 3rd Qu.: 39.476 3rd Qu.: 177707 3rd Qu.: 56.96   
## Max. :727548 Max. : 223.956 Max. :6904824 Max. : 99.08   
## NA's :1

El siguiente paso consiste en localizar los posibles ***missing values***, ya que para realizar el análisis es necesario que todos los casos posean dato para todas las variables originales. Para tener una idea general, se puede utilizar la función vis\_miss() del paquete {visdat}, que nos localizará gráficamente los *missing values* de las diferentes variables, y calculará el porcentaje de casos que supone, con respecto al total de observaciones.

# Identificando missing values.  
  
library(visdat)  
vis\_miss(originales)



Una vez puesta de manifiesto la existencia de un *missing value*, localizaremos el caso concreto que lo contiene aplicando la función filter() de {dplyr}. Se trata de la empresa “Biovent Energía S.A.”

Sabemos que ante la existencia de *missing values*, se puede actuar de varios modos. Por ejemplo, **se puede intentar obtener por otro canal de información el conjunto de valores** que no están disponibles, **o recurrir a alguna estimación**. En caso de que esto sea difícil, se puede optar por **eliminar** estos casos, en especial cuando representan un porcentaje muy reducido respecto al total de casos. En nuestro ejemplo, supondremos que hemos optado por esta última vía, y eliminaremos este caso con otro filtro:

originales %>%  
 filter(is.na(RES) | is.na(MARGEN) | is.na(FPIOS) | is.na(SOLVENCIA)) %>%  
 select(RES, MARGEN, FPIOS, SOLVENCIA)

## RES MARGEN FPIOS SOLVENCIA  
## Biovent Energia SA NA 22.792 70033 38.082

originales <- originales %>%  
 filter(! is.na(RES) & ! is.na(MARGEN) & ! is.na(FPIOS) & ! is.na(SOLVENCIA))

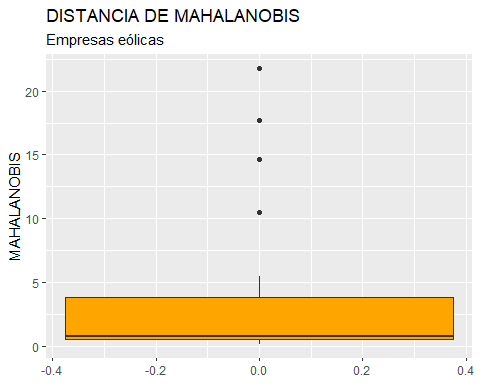
El *data frame* “originales” pasa a tener 24 observaciones, ya que se ha descartado a la empresa a la que le faltaba el dato del resultado del ejercicio (RES).

El siguiente paso es la **identificación de *outliers*.** Para realizar este proceso, y dado que en nuestro análisis contamos con 4 variables, primero “resumiremos” el valor que toman dichas variables para cada caso, mediante el cálculo de la *distancia de Mahalanobis*. De hecho, las distancias de los diferentes casos se almacenarán en una nueva variable, a la que llamaremos MAHALANOBIS, que se incorporará al *data frame* “originales” por medio de la función mutate() de {dplyr}, y la función mahalanobis(). Recordemos que, en los diferentes argumentos de esta función, el punto “.” hace referencia al *data frame* que está delante del operador pipe (%>%).

# Identificando outliers.  
  
originales <- originales %>% mutate(MAHALANOBIS = mahalanobis(.,  
 center = colMeans(.),  
 cov=cov(.)))

Una vez creada la variable MAHALANOBIS, se estudia la existencia en sus valores de *outliers* mediante la construcción de un diagrama de caja o *boxplot*:

library (ggplot2)  
ggplot(data = originales, map = (aes(y = MAHALANOBIS))) +  
 geom\_boxplot(fill = "orange") +  
 ggtitle("DISTANCIA DE MAHALANOBIS", subtitle = "Empresas eólicas") +  
 ylab("MAHALANOBIS")



En el gráfico se observa que existen, por encima de la caja, 4 outliers. Para identificarlos de modo concreto, hemos de calcular los cuartiles primero y tercero de la variable MAHALANOBIS y pasar el correspondiente filtro:

Q1M <- quantile (originales$MAHALANOBIS, c(0.25))  
Q3M <- quantile (originales$MAHALANOBIS, c(0.75))  
  
originales %>%  
 filter(MAHALANOBIS > Q3M + 1.5\*IQR(MAHALANOBIS) |  
 MAHALANOBIS < Q1M - 1.5\*IQR(MAHALANOBIS)) %>%  
 select(MAHALANOBIS, RES, MARGEN, FPIOS, SOLVENCIA)

## MAHALANOBIS RES MARGEN FPIOS SOLVENCIA  
## Holding De Negocios De GAS SL. 21.77393 727548.000 91.152 6904824.0 51.174  
## Global Power Generation SA. 17.67724 39995.000 22.403 1740487.0 86.917  
## WPD Wind Investment SL. 10.50395 -850.068 -302.027 108023.8 99.082  
## Sargon Energias SLU 14.61446 -2216.000 -615.625 -10985.0 -12.811

Estas empresas cuentan con un valor atípico en la variable MAHALANOBIS lo que, a su vez, implica que muestren valores atípicos en una o varias de las variables originales (RES, MARGEN, FPIOS, SOLVENCIA). En el desarrollo de otras técnicas, en este punto localizaríamos y eliminaríamos los *outliers*. En este caso **no** lo vamos a hacer, ya que queremos agrupar **todos los casos** que tenemos en el análisis. Precisamente, si hay algún caso que permanece aislado, sin agruparse con otros en el proceso de agrupación hasta las últimas etapas, quizá se trate de un candidato a *outlier*, por lo que el análisis clúster **también puede considerarse una técnica de localización de casos atípicos**.

Por último, borramos la variable MAHALANOBIS del data frame “originales”, puesto que ya ha cumplido la función de localizar los casos atípicos:

originales <- originales %>% select(-MAHALANOBIS)

La siguiente etapa y parte del script se refiere a la aplicación propia del análisis clúster al grupo de casos (24 empresas) que toman valores para las 4 variables incluidas en el análisis.

Los métodos de agrupación usualmente se basan en la **distancia euclídea**. Como la distancia euclídea es sensible a las unidades de medida de las diferentes variables clasificadoras, es preciso trabajar con las **variables tipificadas**, lo que lograremos creando, por ejemplo, un *data frame* “zoriginales” con la función scale(). Luego, aplicaremos el método seleccionado a este *data frame,* en lugar de al *data frame* que contiene los datos originales sin tipificar:

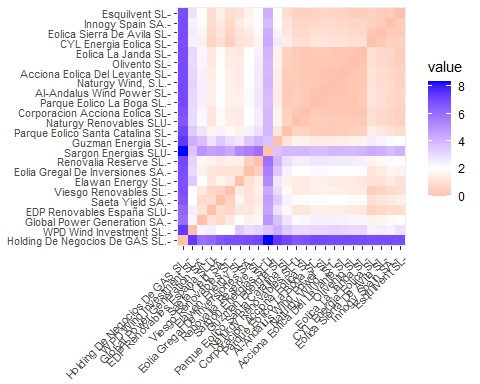
# CLUSTER JERARQUICO CON VARIABLES ORIGINALES.  
  
# Tipificando variables  
  
zoriginales <- data.frame(scale(originales))  
summary (zoriginales)

## RES MARGEN FPIOS SOLVENCIA   
## Min. :-0.3178 Min. :-3.79391 Min. :-0.3904 Min. :-2.13295   
## 1st Qu.:-0.2631 1st Qu.: 0.03333 1st Qu.:-0.3164 1st Qu.:-0.69365   
## Median :-0.2310 Median : 0.10993 Median :-0.2866 Median :-0.06402   
## Mean : 0.0000 Mean : 0.00000 Mean : 0.0000 Mean : 0.00000   
## 3rd Qu.:-0.2072 3rd Qu.: 0.19854 3rd Qu.:-0.2093 3rd Qu.: 0.63264   
## Max. : 4.6632 Max. : 1.32264 Max. : 4.5230 Max. : 1.73657

Este nuevo *data frame* contiene las mismas variables del análisis; pero tipificadas (obsérvese, en el summary(), las medias de las variables).

Previamente a aplicar un método de agrupación concreto, conviene calcular la **matriz de distancias** entre los casos, a la que llamaremos, por ejemplo, “d”. Esta matriz se calcula con la función dist(). Para visualizarla, una opción es representarla mediante el *gráfico de temperatura* que ofrece la función fviz\_dist() del paquete {factoextra}:

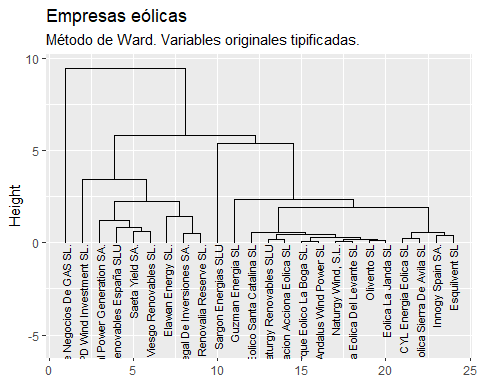
# Matriz de distancias  
  
d <- dist(zoriginales)  
library (factoextra)  
fviz\_dist(d, lab\_size = 8)



Los casos con intersecciones en tonos anaranjados tenderán a agruparse con mayor facilidad (o a agruparse antes); mientras que los casos cuyas intersecciones están en tonos azulados tenderán a pertenecer a grupos diferentes (o a agruparse más tarde). Puede observarse cómo las distancias de tres de las empresas que fueron identificadas como *outliers* (“Holding de Negocios de Gas S. L.”, “Sargon Energías SLU” y “WPD Wind Investment S. L.”) matienen grandes distancias (casillas azuladas) con el resto de empresas. No ocurre lo mismo con la compañía identificada como *outlier*, “Global Power Generation, S. A.”, que mantiene distancias más discretas como “Viesgo Renovables S. L.”, “Saeta Yield S. A.” o “EDP Renovables España S. L. U.”

Vamos a realizar el análisis clúster jerárquico mediante uno de los métodos más habituales, el de ***Ward***, como es común en las aplicaciones prácticas, ya que este método proporciona grupos muy homogéneos (mínima varianza). La función a utilizar es hclust(). La solución la guardaremos en el objeto (lista) que hemos llamado, por ejemplo, “cluster\_j”. Luego se visualizará el ***dendograma*** construido con la función fviz\_dend() del paquete {factoextra}, que permite personalizar el gráfico con una gramática similar a la utilizada con los gráficos del paquete {ggplot2}:

# Método de Ward.  
  
cluster\_j<-hclust(d, method="ward.D2")  
fviz\_dend(cluster\_j,  
 cex=0.6,  
 rect = FALSE,  
 labels\_track\_height = 5.5) +  
 labs(title = "Empresas eólicas",  
 subtitle = "Método de Ward. Variables originales tipificadas.") +  
 theme\_grey()



En el código anterior:

* **cluster\_j**: Es el objeto que contiene el dendrograma que se desea visualizar.
* **cex = 0.6**: Este argumento ajusta el tamaño del texto de las etiquetas en el dendrograma. Un valor de **0.6** significa que el texto será más pequeño que el tamaño predeterminado.
* **rect = FALSE**: Este argumento indica si se deben dibujar rectángulos alrededor de los clústeres en el dendrograma. **FALSE** significa que no se dibujarán rectángulos.
* **labels\_track\_height = 5.5**: Este argumento ajusta la altura de la pista de etiquetas, que es el espacio reservado para las etiquetas de los objetos en el dendrograma. Un valor de **5.5** proporciona más espacio para las etiquetas.

Además, el código incluye funciones adicionales para mejorar la visualización:

* **labs(title = "Empresas eólicas", subtitle = "Método de Ward. Variables originales tipificadas.")**: Esta función añade un título y un subtítulo al gráfico.
* **theme\_grey()**: Esta función aplica un tema gris al gráfico, que es el tema predeterminado en ggplot2, proporcionando un fondo gris claro y un estilo de texto específico.

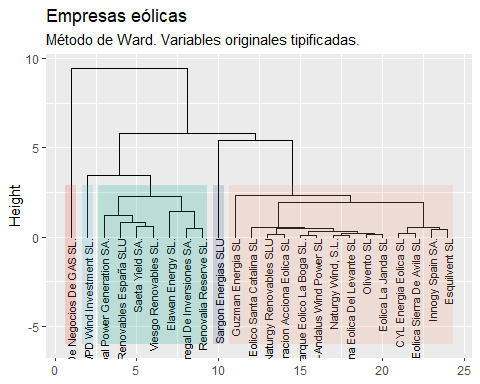
El eje vertical del *dendograma* recoge las distancias (o disimilitud) entre los casos y/o grupos previos que se van agrupando sucesivamente. La escala depende de cada método empleado. En el caso del método de *Ward*, la escala refleja la suma de cuadrados de la distancia de los casos dentro del clúster. Por otro lado, en este ejemplo, es interesante observar que tres de las empresas *outliers* (“Holding de Negocios de Gas S. L.”, “Sargon Energías SLU” y “WPD Wind Investment S. L.”) se agrupan con el resto de casos (o grupos nprecedentes) en una fase muy tardía del proceso de agrupamiento (muy cerca del único grupo). No es el caso de la tercera empresa identificada como *outlier*, “Gobal Power Generation S. A.”

Una cuestión importante consiste en determinar con **cuántos grupos** hemos de quedarnos. Aunque existen algoritmos y paquetes de R que aconsejan un número (por ejemplo, la función NbClust() del paquete {NbClust}); puede ser preferible que el propio investigador decida el número de grupos a crear, mediante la observación del dendograma, y de acuerdo a los objetivos de su propia investigación.

En este ejemplo, un número de grupos razonable podría ser 5, que contaría con el aval de mantener individualizadas a 3 de las empresas etiquetadas como *outliers*. Si se acepta esta opción, se podrá visualizar de nuevo el dendograma coloreando los grupos formados, con el código siguiente (debe modificarse el código para igualar el argumento k = al número de grupos seleccionado):

fviz\_dend(cluster\_j,  
 cex = 0.6,  
 k = 5, # número de grupos o conglomerados que se ha decidido formar!  
 k\_colors = "black",  
 labels\_track\_height = 5.5,  
 rect = TRUE,  
 rect\_border = "npg",  
 rect\_fill = TRUE) +  
 labs(title = "Empresas eólicas",  
 subtitle = "Método de Ward. Variables originales tipificadas.") +  
 theme\_grey()

## Warning in get\_col(col, k): Length of color vector was shorter than the number of clusters - color vector was recycled



En el código anterior:

* **k = 5**: Especifica el número de grupos o clústeres que se desea formar en el dendrograma. En este caso, se han decidido formar 5 grupos.
* **k\_colors = "black"**: Define el color de las etiquetas de los clústeres. Aquí, se ha elegido el color negro para las etiquetas.
* **rect = TRUE**:Significa que se dibujarán rectángulos delimitando los grupos formados.
* **rect\_border = "npg"**: Define el color del borde de los rectángulos que rodean los clústeres. **"npg"** es un conjunto de colores predefinidos (el de las publicaciones del *Nature Publishing Group*) en el paquete **{ggsci}**, que proporciona paletas de colores científicas.
* **rect\_fill = TRUE**: Indica si los rectángulos que rodean los clústeres deben estar rellenos. **TRUE** significa que los rectángulos estarán rellenos con el color especificado.

En el dendograma se aprecia cómo hay un grupo formado por 14 empresas, otro de 7, y finalmente 3 grupos individuales (que se corresponden con tres de las empresas que, al comienzo del *script*, identificamos como *outliers*).

A continuación vamos a **identificar con mayor detalle los casos** que integran cada uno de los grupos, así como a **caracterizar tales grupos** en función de los valores medios de las variables originales. Para ello, crearemos el vector de valores enteros que indica el grupo al que pertenece cada caso (empresa). A este vector se le llamará, por ejemplo, “whatcluster\_j”, y se construirá mediante la función cutree(), donde el primer argumento es el nombre del objeto que guarda la solución del análisis clúster (“cluster\_j”), y el segundo argumento es el número de grupos que hemos decidido crear (k = 5). Conviene convertir esta variable en un factor con la función as.factor(), para que deje de ser variable métrica, a efectos de incorporar una leyenda en gráficos posteriores, . Finalmente, ese factor se incorporará al *data frame* “originales” (importante: no a ”zoriginales”; sino al *data frame* que contiene a las variables no tipificadas):

# CARACTERIZACIÓN Y COMPOSICIÓN DE GRUPOS.  
  
originales$whatcluster\_j <- as.factor(cutree(cluster\_j, k=5))  
levels(originales$whatcluster\_j)

## [1] "1" "2" "3" "4" "5"

Una vez incorporado el grupo de pertenencia de cada empresa al *data frame* “originales”, se podrán calcular y almacenar las medias de cada grupo de las distintas variables originales, usando las funciones by\_group() y summarise() de {dplyr}. Los decimales se ajustarán utilizando la función round(). Toda la información se asigna al *data frame* “tablamedias” para poder representarla en una tabla mediante las facilidades que ofrecen los paquetes {knitr} y {kableExtra}:

# Tabla con centroides de grupos.  
  
tablamedias <- originales %>%  
 group\_by(whatcluster\_j) %>% summarise(obs = length(whatcluster\_j),  
 Resultado = round(mean(RES),0),  
 Margen = round(mean(MARGEN),2),  
 Fondos\_Propios = round(mean(FPIOS),0),  
 Solvencia = round(mean(SOLVENCIA),2))  
  
library (knitr)  
library (kableExtra)  
knitr.table.format = "html"  
  
tablamedias %>%  
 kable(format = knitr.table.format,  
 caption = "Método de Ward. 5 grupos. Medias de variables",  
 col.names = c("Clúster", "Observaciones", "Resultado", "Margen",  
 "Fondos Propios", "Solvencia")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped", "bordered", "condensed",  
 position = "center",  
 font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:nrow(tablamedias), bold= F, align = "c")

(#tab:unnamed-chunk-215)(#tab:unnamed-chunk-215)Método de Ward. 5 grupos. Medias de variables

Clúster

Observaciones

Resultado

Margen

Fondos Propios

Solvencia

1

1

727548

91.15

6904824

51.17

2

7

19778

102.31

523849

72.75

3

14

8850

19.75

56190

16.09

4

1

-850

-302.03

108024

99.08

5

1

-2216

-615.62

-10985

-12.81

Obviamente, también se podrían comparar las medias de los grupos, para cada variable, con un simple gráfico de barras. A fin de crear un método que valga para cualquier número de variables, realizaremos la tarea con un bucle. El código es el siguiente:

# Gráficos de centroides  
  
 # Vector de nombre de variables excluyendo la variable no deseada  
 variables <- setdiff(names(tablamedias), c("whatcluster\_j", "obs"))  
  
 # Lista para almacenar los gráficos  
 graficos.centroides <- list()  
  
 # Bucle para crear y almacenar los gráficos  
 for (i in seq\_along(variables)) {  
 var1 <- variables[[i]]  
 grafico <- ggplot(data= tablamedias,  
 map = (aes\_string(y = var1, x = "whatcluster\_j"))) +  
 geom\_bar(stat = "identity",  
 colour = "red",  
 fill = "orange",  
 alpha = 0.7) +  
 ggtitle(paste0(var1, ". Media por grupos."),  
 subtitle = "Empresas eólicas")+  
 xlab ("Grupo") +  
 ylab(var1)  
 graficos.centroides[[paste0("grafico\_", var1)]] <- grafico  
}

En el código anterior, setdiff() crea un vector “variables” que contiene todos los nombres de las columnas de tablamedias, excepto "whatcluster\_j" y "obs", que no son las variables originales. Luego se crea una lista vacía “graficos.centroides” para almacenar los gráficos generados. Con for (i in seq\_along(variables)) comienza el bucle, que recorre cada elemento del vector “variables”. En el código de gráfico, “var1” toma el nombre, en cada iteración, de la variable a representar. En el “mapeo”, es importante utilizar aes\_string(), que requiere que los nombres de las variables se pasen como cadenas de texto (entre comillas), lo que es útil cuando los nombres de las variables se generan dinámicamente o se pasan como argumentos de función, y es útil en programación cuando se necesita construir *mapeos estéticos* de manera programática. Finalmente, con graficos.centroides[[paste0("grafico\_", var1)]] <- grafico se guarda el gráfico en la lista “graficos.centroides” con un nombre basado en “var1”.

Los gráficos guardados en la lista “gráficos.centroides” se pueden agrupar en composiciones, de, por ejemplo, 2x2, utilizando el paquete {patchwork}. En esta ocasión, vamos a diseñar una función que pueda ser utilizada con facilidad para estos gráficos y otros más generados con {ggplot2} y guardados en una lista (antes hemos activado el paquete {patchwork}). En concreto, la función crea agrupaciones de 4 gráficos (la última podrá tener menos) para presentar todos los gráficos almacenados de un modo más condensado. Hemos llamado a la función: create\_patchwork()

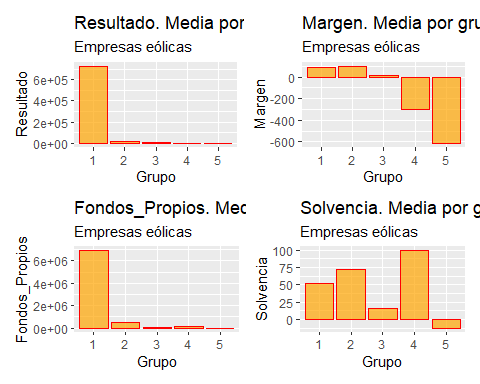
library (patchwork)   
  
# Función para crear agrupaciones de 2x2 con espacios en blanco si es necesario  
 create\_patchwork <- function(plot\_list) {   
 n <- length(plot\_list)   
 full\_rows <- n %/% 4   
 remaining <- n %% 4   
 patchworks <- list()  
   
 # Crear agrupaciones completas de 2x2   
   
 for (i in seq(1, full\_rows \* 4, by = 4)) {  
 patchworks <- c(patchworks,  
 list((plot\_list[[i]] | plot\_list[[i+1]]) /  
 (plot\_list[[i+2]] | plot\_list[[i+3]]))) }  
 # Crear la última agrupación con espacios en blanco si es necesario   
 if (remaining > 0) {   
 last\_plots <- plot\_list[(full\_rows \* 4 + 1):n]   
 empty\_plots <- lapply(1:(4 - remaining), function(x) ggplot() +  
 theme\_void())  
 last\_patchwork <- do.call(patchwork::wrap\_plots,  
 c(last\_plots, empty\_plots))  
 patchworks <- c(patchworks, list(last\_patchwork)) }  
 return(patchworks) }

Los elementos claves de la función son:

* **plot\_list**: Lista de los gráficos que se van a organizar.
* **n**: Número total de gráficos en **plot\_list**.
* **full\_rows**: Número de filas completas de 2x2 que se pueden formar (cada fila tiene 4 gráficos).
* **remaining**: Número de gráficos que sobran después de formar las filas completas.
* **patchworks <- list()**: inicialización de la lista para almacenar la lista de composiciones de gráficos que se van a almacenar.
* **seq(1, full\_rows \* 4, by = 4)**: Genera una secuencia de índices para iterar sobre los gráficos en grupos de 4.
* Dentro del bucle, se crean composiciones de 2x2 usando la sintaxis de **patchwork**:
  + **(plot\_list[[i]] | plot\_list[[i+1]]) / (plot\_list[[i+2]] | plot\_list[[i+3]])**: Combina cuatro gráficos en una disposición de 2x2.
* **patchworks <- c(patchworks, list(...))**: Añade cada nueva composición a la lista **patchworks**.
* **if (remaining > 0)**: Verifica si hay gráficos restantes después de formar las filas completas.
* **last\_plots**: Selecciona los gráficos restantes.
* **empty\_plots**: Crea gráficos vacíos (**ggplot() + theme\_void()**) para llenar los espacios hasta completar un grupo de 4.
* **last\_patchwork**: Combina los gráficos restantes y los gráficos vacíos en una última composición usando **patchwork::wrap\_plots**.
* **patchworks <- c(patchworks, list(last\_patchwork))**: Añade la última composición a la lista “patchworks”.
* **return(patchworks)** finaliza la ejecución de la función y devuelve la lista “patchworks".

Una vez creada la función, se pasa a nuestra lista de gráficos, “gráficos.centroides”, creando la lista de “grupos de gráficos” llamada “grupos-graficos.centroides”:

# Aplicar a gráficos de centroides.  
   
 grupos.graficos.centroides <- create\_patchwork(graficos.centroides)  
   
 # Presentar las composiciones  
 for (n in 1:length(grupos.graficos.centroides)){  
 print(grupos.graficos.centroides[[n]])  
 }



Hablando siempre en términos de la media (centroides), puede comprobarse cómo el grupo 1 destaca por su alto valor en el resultado y en los fondos propios, el grupo 2 por el alto margen y el alto valor de la solvencia, el tercer grupo por poseer valores discretos (pero no negativos) en las cuatro variables, el cuarto grupo por la elevada solvencia y el margen negativo, y el quinto grupo por el margen y la solvencia negativos.

Por otro lado, se pueden presentar en diferentes tablas las composiciones e informaciones de cada grupo. Vamos a automatizar de nuevo el proceso de generación de las tablas mediante el empleo de un *bucle.* Las diferentes tablas se irán guardando en una lista de nombre, por ejemplo, “tablascompo”:

# Tablas con composiciones de grupos  
  
 # Número de tablas y lista para guardarlas  
  
 numclusters <- nlevels(originales$whatcluster\_j)  
 tablascompo <- list()  
  
 # Bucle para generar las tablas  
  
 for (n in 1:numclusters){  
 tabla <- originales %>%  
 filter(whatcluster\_j == as.character(n)) %>%  
 select(RES, MARGEN, FPIOS, SOLVENCIA) %>%  
 kable(caption = paste("Método de Ward. Grupo ", n, "."),  
 col.names = c("Resultado", "Margen",  
 "Fondos Propios", "Solvencia"),  
 format.args = list(decimal.mark = ".", digits = 2)) %>%  
 kable\_styling(full\_width = FALSE,   
 bootstrap\_options = c("striped",  
 "bordered",  
 "condensed"),  
 position = "center",  
 font\_size = 12) %>%  
 row\_spec(0, bold = TRUE, align = "c")  
 tablascompo[[n]] <- tabla  
 }  
  
 # Presentar las tablas  
 for (n in 1:numclusters){  
 print(tablascompo[[n]])  
 }

Las tablas obtenidas serán:

Para representar los grupos gráficamente, tenemos la dificultad de contar con más de dos variables clasificadoras. Una idea es **generar todas las combinaciones de variables posibles**, generar los correspondientes gráficos de dispersión con los casos coloreados de modo diferente según el grupo de pertenencia, y presentar estos gráficos de modo compacto utilizando la función create\_patchwork() que se incluyó anteriormente para hacer mediante {patchwork}.composiciones de 2x2 gráficos.

Vamos a realizar la tarea de generar todos los gráficos de modo automatizado. Primero generaremos un vector con el nombre de todas las variables (excluyendo al *factor* whatcluster\_j) mediante la función setdiff(). Luego, crearemos una lista para ir almacenando los gráficos (lista “graficos”). Por último, calcularemos todas las combinaciones de nombres de variables posibles, con la función combn(), y las almacenaremos en la lista “combinaciones”. En esta función, el argumento **simplify = FALSE** le dice a la función que no vuelque el resultado a una matriz. En su lugar, devuelve una lista donde cada elemento de la misma es una combinación de los elementos del vector original.

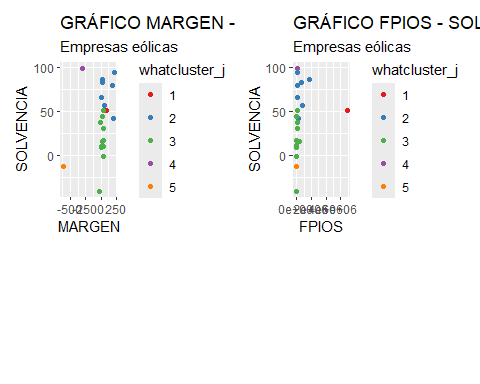
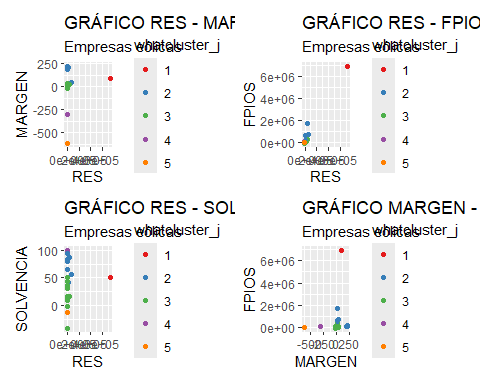
# Gráficos Variable vs Variable  
  
 # Lista de variables excluyendo la variable no deseada  
 variables <- setdiff(names(originales), "whatcluster\_j")  
  
 # Lista para almacenar los gráficos  
 graficos <- list()  
  
 # Generar todas las combinaciones posibles de pares de variables  
 combinaciones <- combn(variables, 2, simplify = FALSE)

El siguiente paso consiste en utilizar un bucle para generar los gráficos de dispersión de acuerdo a las combinaciones de variables obtenidas y guardarlos en la lista “graficos”. El bucle itera tantas veces como elementos guarda la lista “combinaciones” (argumento/función seq\_along()):

# Bucle para crear y almacenar los gráficos  
 for (i in seq\_along(combinaciones)) {  
 var1 <- combinaciones[[i]][1]  
 var2 <- combinaciones[[i]][2]  
 grafico <- ggplot(originales,  
 map = aes\_string(x = var1,  
 y = var2,  
 color = "whatcluster\_j")) +  
 geom\_point() +  
 labs(title = paste("GRÁFICO", var1, "-", var2),  
 subtitle = "Empresas eólicas") +  
 xlab (var1) +  
 ylab (var2) +  
 scale\_color\_brewer(palette = "Set1")   
 graficos[[paste0("grafico\_", var1, "\_", var2)]] <- grafico  
}

Una vez generados los gráficos de todas las combinaciones de variables (6 gráficos en el ejemplo), y almacenados en la lista “graficos”, se podrán reagrupar y presentar en composiciones de 2x2 mediante el empleo de la función create\_patchwork():

# Hacer agrupaciones con la función de patchworks creada anteriormente  
  
 gruposgraficos <- create\_patchwork(graficos)  
  
 # Presentar las composiciones  
 for (n in 1:length(gruposgraficos)){  
 print(gruposgraficos[[n]])  
 }



De los gráficos anteriores, pueden extraerse algunas conclusiones. Por ejemplo, en los tres primeros gráficos, destaca la posición de la única empresa que forma el grupo 1 (Holding de Negocios de Gas, S. L.), sobre todo en cuanto al resultado RES (eje x). En este sentido, mención especial merece el gráfico 2 (RES-FPIOS), debido a la especial situación de esta empresa en cuanto a Fondos Propios (FPIOS), lo que se aprecia también en los gráficos 4 y 6. El grupo 4 también está constituido únicamente por una empresa (WPD Wind Investment S. L.), si bien en los diferentes gráficos no muestra una separación tan notable con respecto al resto de empresas y grupos. Tan solo en los gráficos 1, 4 y 5 destaca, debido a la influencia de la variable de margen de beneficio (MARGEN). El grupo 5 es el tercer y último grupo constituido por una única empresa (Sargon Energías S. L. U.). En este caso, su comportamiento destaca en los mismos gráficos que el caso anterior, debido también a su margen de beneficio (MARGEN), visiblemente por debajo del valor presente en el resto de empresas. En cuanto a los grupos constituidos por más de una empresa, las empresas de ambos grupos se muestran relativamente próximas, pudiéndose concluir que las empresas del grupo 2 tienen, en general, mayor volumen de fondos propios (FPIOS) y coeficiente de solvencia (SOLVENCIA) que las del grupo 3; mientras que en cuanto a margen (MARGEN) y resultado (RES) muestran, globalmente, valores próximos.

## 7.3 Métodos de agrupación no-jerárquicos.

Dentro del análisis clúster, los métodos de agrupación no-jerárquicos se utilizan en casos en los que hay un elevado número de casos que clasificar. Son especialmente útiles cuando nuestro objetivo pasa por crear grupos que definan **una tipología de casos o individuos**, más que clasificar casos o individuos concretos. En definitiva, identificar subpoblaciones a partir de una muestra. Por eso es conveniente, previamente, detectar los *outliers* y, en su caso, eliminarlos; ya que podrían distorsionar las características de los grupos debido a la sensibilidad de los algoritmos a la presencia de casos atípicos.

Una diferencia clave con respecto a los métodos jerárquicos es que **es necesario decidir *a priori* el número de conclomerados** o grupos de casos a formar. Por otro lado, son métodos más eficientes, y permiten el traslado de casos de unos grupos a otros.

Aunque existen otros métodos, la técnica no-jerárquica más común es la de **k-medias**. Este es un método iterativo. Se establece un *centroide* inicial (“semilla”) para cada uno de los k grupos que se quieren crear, y se van asignando a cada grupo los casos que se sitúen más cerca de su centro. Una vez asignados los casos, se recalculan los *centroides* de los grupos, y se repite el proceso en una nueva iteración. El procedimiento termina cuando el algoritmo encuentra la solución convergente (estable). Precisamente, la elección de las “semillas” iniciales es otro de las debilidades que presenta el método, ya que de ello puede depender la obtención de soluciones diferentes.

¿Cómo fijar el **número de grupos o conglomerados a formar**? Hay ocasiones en que las que el investigador establecerá un número que le sea manejable o útil según los objetivos que persiga. Si esto no es así, y no se tiene claro el número de grupos a construir, se podrá optar por probar con varios números, y evaluar las soluciones obtenidas. También puede ayudar el realizar previamente un análisis jerárquico para estudiar el dendograma. Además, existen algoritmos, como *NbClust* en R, que sugieren un número de grupos en función de una batería de pruebas presentes en la literatura.

La segunda cuestión clave es **cómo determinar las “semillas”** o centroides iniciales. Una opción es generar las semillas de modo aleatorio, aunque no es un método muy conveniente. De hecho, cada vez que se aplicara el algoritmo de *k-medias*, podría obtenerse una solución diferente. Otra alternativa es la fijación de las “semillas” por parte del investigador. Una idea, en este sentido, es hacer un *clúster jerárquico* previo, y tomar los *centroides* de la solución final como “semillas” de k-medias. Otra posibilidad interesante es aplicar el método del *centroide más lejano*: se fija el primer centroide al azar, pero luego el 2º centroide coincidirá con el punto de datos más alejado de él. En general, el jº centroide coincidirá con el punto cuya distancia mínima a los centroides precedentes sea mayor. Se pretende que los centroides estén bien separados unos de otros. Una versión mejorada es el método *k-medias++*.

Para nuestro ejemplo práctico, vamos a volver a utilizar como variables clasificadoras las del ejemplo desarrollado para los métodos jerárquicos; pero esta vez para una muestra de 350 empresas de generación de electricidad mediante tecnología eólica. Estas variables son, de nuevo, resultado del ejercicio (RES), margen de beneficio (MARGEN), fondos propios (FPIOS), y solvencia (SOLVENCIA). Con base en ellas, queremos **establecer una serie de perfiles o tipologías** distintas de las empresas que componen el sector.

Trabajaremos, como en el ejemplo de clúster jerarquizado, en el proyecto llamado “cluster”. Vamos a ir a la carpeta del proyecto y vamos a guardar en ella los dos archivos de esta práctica: un archivo de Microsoft® Excel® llamado “[eolica\_350.xlsx](https://docs.google.com/spreadsheets/d/1Brtu_58lcfXL5ZTeOi9-xlxy7S9NfuxB/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true)” y un *script* denominado “[kmedias\_eolica.R](https://drive.google.com/file/d/1nOBc99SsMKYACitAVRD0nnpP8q-8FqJZ/view?usp=sharing)”. Si abrimos el archivo de Microsoft® Excel®, comprobaremos que se compone de tres hojas. La primera muestra un aviso sobre la utilización de los datos, la segunda el nombre y definición de las variables disponibles, y la tercera (hoja “Datos”) guarda los datos que debemos importar desde R-Studio. Estos datos se corresponden con diferentes variables económico-financieras de 350 empresas productoras de electricidad mediante generación eólica.

La primera parte del *script* se ocupa de importar los datos de la hoja de cálculo, limpiando previamente la memoria (*Environment*) de posibles objetos almacenados en sesiones anteriores:

# CLUSTER de k-medias productores eolicos. Disculpad la falta de tildes!!!!  
  
rm(list = ls())  
  
# DATOS  
  
 # Importando  
  
 library(readxl)  
 eolicos <- read\_excel("eolica\_350.xlsx", sheet = "Datos",  
 na = c("n.d.", "s.d."))  
 eolicos <- data.frame(eolicos, row.names = 1)  
 summary (eolicos)

## MARGEN SOLVENCIA COM FJUR   
## Min. :-4124.22 Min. :-83.97 Length:350 Length:350   
## 1st Qu.: 19.15 1st Qu.: 14.92 Class :character Class :character   
## Median : 42.93 Median : 38.78 Mode :character Mode :character   
## Mean : 32.60 Mean : 40.11   
## 3rd Qu.: 64.77 3rd Qu.: 68.16   
## Max. : 1790.12 Max. : 99.78   
##   
## ING NCOMP RES ACTIVO   
## Min. : 69.7 Min. : 0 Min. :-16107.21 Min. : 36.8   
## 1st Qu.: 822.3 1st Qu.: 2 1st Qu.: 48.58 1st Qu.: 3569.9   
## Median : 4060.7 Median : 72 Median : 751.07 Median : 17239.6   
## Mean : 8985.4 Mean : 1526 Mean : 2616.99 Mean : 50776.5   
## 3rd Qu.: 8408.4 3rd Qu.: 398 3rd Qu.: 3119.00 3rd Qu.: 41660.9   
## Max. :364989.0 Max. :72434 Max. : 78290.00 Max. :2429299.0   
## NA's :1 NA's :1   
##   
## FPIOS RENECO RENFIN LIQUIDEZ   
## Min. : -51817.4 Min. :-85.351 Min. :-687.418 Min. : 0.0070   
## 1st Qu.: 635.1 1st Qu.: 3.169 1st Qu.: 7.804 1st Qu.: 0.7738   
## Median : 3421.8 Median : 9.948 Median : 26.208 Median : 1.8440   
## Mean : 20814.8 Mean : 15.190 Mean : 70.773 Mean : 9.1875   
## 3rd Qu.: 11814.0 3rd Qu.: 22.402 3rd Qu.: 56.279 3rd Qu.: 3.8505   
## Max. :1382020.0 Max. :102.130 Max. :6788.328 Max. :1367.0610   
##   
## APALANCA AUTOFIN DIMENSION AUTOFINA   
## Min. : -7016.77 Min. : -3 Length:350 Length:350   
## 1st Qu.: 2.99 1st Qu.: 0 Class :character Class :character   
## Median : 48.62 Median : 1 Mode :character Mode :character   
## Mean : 974.23 Mean : 15120   
## 3rd Qu.: 248.31 3rd Qu.: 4   
## Max. :177381.90 Max. :4486914   
## NA's :53   
##   
## RENTALIQ VALORACION   
## Min. :-226.188 Length:350   
## 1st Qu.: 5.668 Class :character   
## Median : 15.148 Mode :character   
## Mean : 31.717   
## 3rd Qu.: 30.400   
## Max. :2264.254

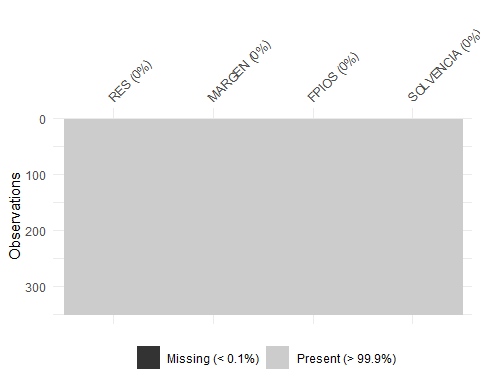
Podemos observar cómo, en el *Environment,* ya aparece un objeto. Este objeto es una estructura de datos tipo *data frame*, se llama “eolicos” y contiene 18 columnas, una por cada una de las variables almacenadas en el archivo de Microsoft® Excel®. R consideró la primera columna de la hoja de Excel (NOMBRE) como una variable de tipo cualitativo o atributo (por lo que había 19 columnas); pero, al no ser una variable, sino el nombre de los casos o filas (empresas), redefinimos nuestro *data frame* diciéndole que esa primera columna contenía los nombres de los casos (filas).

En el análisis solo vamos a considerar como variables clasificadoras para construir los grupos o conglomerados las variables RES, MARGEN, FPIOS y SOLVENCIA. Por ello, crearemos con ellas un nuevo *data frame* llamado “originales” a partir de la función select() del paquete {dplyr}:

## RES MARGEN FPIOS SOLVENCIA   
## Min. :-16107.21 Min. :-4124.22 Min. : -51817.4 Min. :-83.97   
## 1st Qu.: 48.58 1st Qu.: 19.15 1st Qu.: 635.1 1st Qu.: 14.92   
## Median : 751.07 Median : 42.93 Median : 3421.8 Median : 38.78   
## Mean : 2616.99 Mean : 32.60 Mean : 20814.8 Mean : 40.11   
## 3rd Qu.: 3119.00 3rd Qu.: 64.77 3rd Qu.: 11814.0 3rd Qu.: 68.16   
## Max. : 78290.00 Max. : 1790.12 Max. :1382020.0 Max. : 99.78   
## NA's :1

El siguiente paso consiste en localizar los posibles ***missing values***, ya que para realizar el análisis es necesario que todos los casos posean dato para todas las variables originales. Para tener una idea general, se puede utilizar la función vis\_miss() del paquete {visdat}, que nos localizará gráficamente los *missing values* de las diferentes variables, y calculará el porcentaje de casos que supone, con respecto al total de observaciones:

# Identificando missing values.  
  
 library(visdat)  
 vis\_miss(originales)



Una vez puesta de manifiesto la existencia de un *missing value*, localizaremos el caso concreto que lo contiene aplicando la función filter() de {dplyr}. El caso es la empresa “Desarrollos Eólicos de Teruel S. L.”:

originales %>%  
 filter(is.na(RES) | is.na(MARGEN) | is.na(FPIOS) | is.na(SOLVENCIA)) %>%  
 select(RES, MARGEN, FPIOS, SOLVENCIA)

## RES MARGEN FPIOS SOLVENCIA  
## Desarrollos Eolicos de Teruel SL NA 0 18890.1 37.285

Sabemos que ante la existencia de *missing values*, se puede actuar de varios modos. Por ejemplo, **se puede intentar obtener por otro canal de información el conjunto de valores** que no están disponibles, **o recurrir a alguna estimación**. En caso de que esto sea difícil, se puede optar por **eliminar** estos casos, en especial cuando representan un porcentaje muy reducido respecto al total de casos. En nuestro ejemplo, supondremos que hemos optado por esta última vía, y eliminaremos este caso con otro filtro:

originales <- originales %>%  
 filter(! is.na(RES) & ! is.na(MARGEN) & ! is.na(FPIOS) &  
 ! is.na(SOLVENCIA))

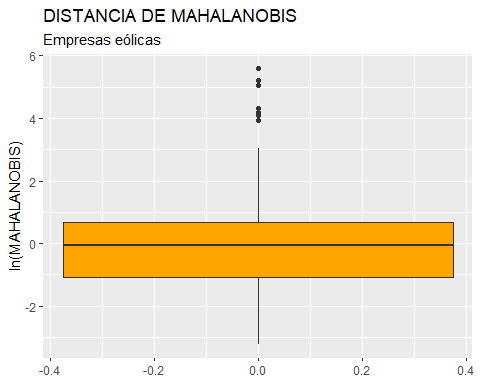
El *data frame* “originales” pasa a tener 349 observaciones, ya que se ha descartado a la empresa a la que le faltaba el dato del resultado del ejercicio (RES).

El siguiente paso es la **identificación de *outliers*.** Para realizar este proceso, y dado que en nuestro análisis contamos con 4 variables, primero “resumiremos” el valor que toman dichas variables para cada caso, mediante el cálculo de la *distancia de Mahalanobis*. De hecho, las distancias de los diferentes casos se almacenarán en una nueva variable, a la que llamaremos MAHALANOBIS, que se incorporará al *data frame* “originales” por medio de la función mutate() de {dplyr}, y la función mahalanobis(). Recordemos que, en los diferentes argumentos de esta función, el punto “.” hace referencia al *data frame* que está delante del operador pipe (%>%):

# Identificando outliers.  
  
 originales <- originales %>% mutate(MAHALANOBIS = mahalanobis((.),  
 center = colMeans(.),  
 cov=cov(.)))

Una vez creada la variable MAHALANOBIS, se estudia la existencia en sus valores de *outliers* mediante la construcción de un diagrama de caja o *boxplot* (para una mejor visión de la “caja”, emplearemos la escala logarítmica):

library (ggplot2)  
   
 ggplot(data = originales, map = (aes(y = log(MAHALANOBIS)))) +  
 geom\_boxplot(fill = "orange") +  
 ggtitle("DISTANCIA DE MAHALANOBIS", subtitle = "Empresas eólicas") +  
 ylab("ln(MAHALANOBIS)")



En el gráfico se observa que existen, por encima de la caja, varios outliers. Para identificarlos de modo concreto, hemos de calcular los cuartiles primero y tercero de la variable MAHALANOBIS y pasar el correspondiente filtro:

Q1M <- quantile (originales$MAHALANOBIS, c(0.25))  
 Q3M <- quantile (originales$MAHALANOBIS, c(0.75))  
  
 originales %>%  
 filter(MAHALANOBIS > Q3M + 1.5\*IQR(MAHALANOBIS) |  
 MAHALANOBIS < Q1M - 1.5\*IQR(MAHALANOBIS)) %>%  
 select(MAHALANOBIS)

## MAHALANOBIS  
## EDP Renovables España SLU 66.135161  
## Naturgy Renovables SLU 154.403714  
## Norvento Estelo SL. 11.646717  
## Acciona Eolica de Castilla LA Mancha SL 8.831525  
## Compañia Eolica Aragonesa SA 4.361055  
## CYL Energia Eolica SL 4.538600  
## Desarrollo de Energias Renovables de Navarra SA 5.958512  
## Molinos DEL Cidacos SA 5.395054  
## Enerfin Enervento SL. 20.615939  
## Danta de Energias SA 4.659270  
## Viesgo Renovables SL. 13.406912  
## Desarrollo Eolico LAS Majas XIX SL. 4.893162  
## Alectoris Energia Sostenible 6 SL. 5.831212  
## Green Capital Power SL 60.334525  
## Elawan Energy SL. 14.892019  
## OW Offshore SL 75.843770  
## Compañia Integral de Energias Renovables de Zaragoza Sociedad Limitada 5.206170  
## Desarrollos Eolicos DEL SUR de Europa Sociedad Limitada. 4.592127  
## Fuerzas Energeticas DEL SUR de Europa XV SL. 4.636275  
## Fuerzas Energeticas DEL SUR de Europa XXI SL. 5.386255  
## Parque Eolico de Adraño SL 5.127652  
## Parque Eolico de A Ruña SL 6.457498  
## Repsol Renovables S.A.U. 181.564290  
## Desarrollo Eolico de LA Muga SL 4.637771  
## Parque Eolico de Virxe DO Monte SL 5.360384  
## Parque Eolico Tahuna S.L. 12.646108  
## Alabe Proyectos Eolicos SA. 7.961243  
## Parsona Corporacion SL. 4.508346  
## Parque Eolico Jaufil SL. 5.262772  
## Parque Eolico LAS Lomas de Lecrin S.L. 4.442394  
## WPD Wind Investment SL. 8.780323  
## Energia Y Recursos Ambientales Internacional SL 8.703097  
## Bluefloat Energy International SL. 267.082180  
## Plafovolt SL 12.144060  
## Minicentrales Bouza Vella SL 50.866827

El filtro anterior nos ofrece un listado de 35 empresas consideradas *outliers*. Estas empresas cuentan con un valor atípico en la variable MAHALANOBIS lo que, a su vez, implica que seguramente posean valores atípicos en una o varias de las variables originales (RES, MARGEN, FPIOS, SOLVENCIA)**. La presencia de *outliers* puede hacer que se distorsione el proceso de obtención de grupos de casos a partir de los cuáles se extraen perfiles o tipologías de empresas**, debido a su influencia sobre el cálculo de los centroides de los grupos. Por tanto, procederemos a eliminar del análisis estas empresas, mediante el paso del correspondiente filtro, y la creación de un nuevo *data frame* sin casos outliers, al que llamaremos, por ejemplo, “originales\_so”. Por último, borramos la variable MAHALANOBIS del *data frame “*originales”, puesto que ya ha cumplido la función de localizar los casos atípicos:

originales\_so <- originales %>%  
 filter(MAHALANOBIS <= Q3M + 1.5\*IQR(MAHALANOBIS) &  
 MAHALANOBIS >= Q1M - 1.5\*IQR(MAHALANOBIS))  
  
 originales\_so <- originales\_so %>% select(-MAHALANOBIS)

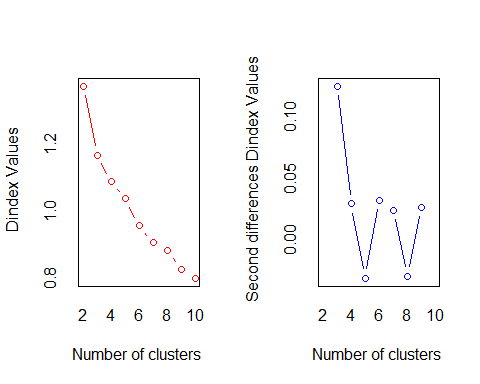
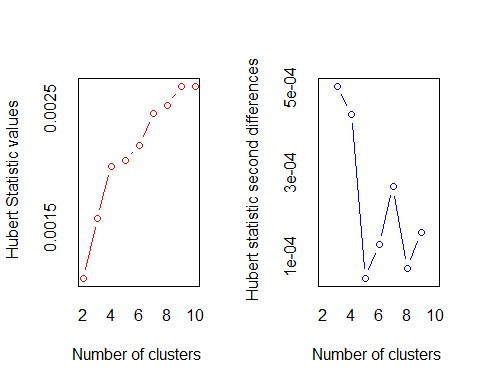
Una vez preparados los datos, vamos a proceder a aplicar la técnica de formación de conglomerados no-jerárquica de k-medias. Primeramente, y puesto que se basa en el cálculo de las distancias euclídeas entre casos, procederemos a tipificar los valores de las variables, utilizando la función scale(), y creando el data frame “zoriginales\_so”:

# CLUSTER K-MEDIAS CON VARIABLES ORIGINALES.  
  
 # Tipificando variables  
  
 zoriginales\_so <- data.frame(scale(originales\_so))  
 summary (zoriginales\_so)

## RES MARGEN FPIOS SOLVENCIA   
## Min. :-3.0216 Min. :-6.41731 Min. :-0.8852 Min. :-2.2102   
## 1st Qu.:-0.6348 1st Qu.:-0.33447 1st Qu.:-0.5943 1st Qu.:-0.8322   
## Median :-0.3982 Median : 0.08039 Median :-0.3900 Median :-0.1216   
## Mean : 0.0000 Mean : 0.00000 Mean : 0.0000 Mean : 0.0000   
## 3rd Qu.: 0.4271 3rd Qu.: 0.48147 3rd Qu.: 0.1449 3rd Qu.: 0.8381   
## Max. : 3.8714 Max. : 7.20005 Max. : 7.2526 Max. : 1.8874

Una cuestión clave es la determinación previa del número de grupos a formar. Si no se tiene decidido un número de conglomerados *a priori* como consecuencia del interés o de los objetivos de la propia investigación, se puede recurrir como orientación a algún paquete de R que propone el número de grupos a formar. Uno de estos paquetes es {NbClust}, y en concreto su función NbClust():

# Numero de grupos  
  
 library(NbClust)  
  
 result <- capture.output(NbClust(data = zoriginales\_so,  
 min.nc = 2,  
 max.nc = 10,  
 method = "kmeans"))



start\_line <- grep("\* Among all indices:", result)  
 end\_line <- grep("\* According to", result)  
 print(result[start\_line:end\_line])

## [1] "\* Among all indices: "   
## [2] "\* 4 proposed 2 as the best number of clusters "   
## [3] "\* 10 proposed 3 as the best number of clusters "   
## [4] "\* 1 proposed 4 as the best number of clusters "   
## [5] "\* 1 proposed 5 as the best number of clusters "   
## [6] "\* 4 proposed 6 as the best number of clusters "   
## [7] "\* 2 proposed 8 as the best number of clusters "   
## [8] "\* 1 proposed 10 as the best number of clusters "   
## [9] ""   
## [10] " \*\*\*\*\* Conclusion \*\*\*\*\* "  
## [11] " "   
## [12] "\* According to the majority rule, the best number of clusters is 3 "

La función NbClust() aplica una batería de entre 20-30 medidas e indicadores recogidos en la literatura científica desarrollados para determinar el número óptimo de grupos o clústeres a formar, y ofrece de modo los resultados. Precisamente, en el código anterior, se almacena el output que ofrece la función en el objeto “result”, luego se elige de entre todo ese *output* el texto donde se ofrece el resumen de los resultados (concretando la línea inicial y la final mediante la función grep()), y se ofrece ese resumen. Según el mismo, la mayoría de los índices e indicadores proponen 3 grupos como la segmentación más adecuada.

Una vez decidido el número de grupos (en el ejemplo, 3), se pasa tal número a una constante “k” para manejarlo con más comodidad en el resto del código. Después, se aplica el método de ***k-medias***, en la versión de la función KMeans\_rcpp() del paquete {ClusterR}, que permite obtener las “semillas” (centroides iniciales) mediante el algoritmo *kmeans++*, que ofrece buenos resultados frente a otras posibilidades de obtención de “semillas”, como la generación puramente aleatoria. La función set.seed() fija la secuencia de generación de números aleatorios, de modo que, si se ejecuta varias veces el mismo código, se obtendrá la misma solución:

k <- 3 # poner aquí número de grupos decidido!!!!  
  
 # Aplicando k-means con inicializacion kmeans++  
  
 library (ClusterR)  
  
 set.seed(123)  
  
 cluster\_k <-KMeans\_rcpp (zoriginales\_so,  
 clusters = k,  
 num\_init = 10,  
 max\_iters = 100,  
 initializer = "kmeans++")

En la función KMeans\_rcpp(), el primer argumento es el *data frame* con las variables clasificadoras (en sus versiones tipificadas). El segundo es el número de clústeres o grupos a obtener (que lo hemos asignado anteriormente al parámetro “**k**”). El tercero, num\_init =, es el número de veces que se repite el procedimiento a fin de retener la mejor solución. Max\_iters = fija el máximo de iteraciones del procedimiento de k*-medias* hasta obtener una solución estable. Initializer = define el método de obtención de las semillas (en nuestro caso, *k-means++*). La solución final se guarda en el objeto “cluster\_k”, como se puede apreciar en el *Environment*.

Dentro de la solución, el vector con el grupo de pertenencia de cada empresa se obtiene con el elemento “$clusters”. Conviene guardar ese vector como factor. En concreto, se guardará como el factor **“whatcluster\_k”**, integrado dentro del *data frame* “originales\_so”:

originales\_so$whatcluster\_k <- as.factor(cluster\_k$clusters)  
 summary(originales\_so)

## RES MARGEN FPIOS   
## Min. :-6521.00 Min. :-297.14 Min. : -3194.0   
## 1st Qu.: 63.34 1st Qu.: 21.33 1st Qu.: 691.8   
## Median : 716.01 Median : 43.05 Median : 3420.0   
## Mean : 1814.51 Mean : 38.84 Mean : 8628.7   
## 3rd Qu.: 2992.77 3rd Qu.: 64.05 3rd Qu.: 10564.2   
## Max. :12494.11 Max. : 415.80 Max. :105490.0   
##   
## SOLVENCIA whatcluster\_k  
## Min. :-25.66 1: 59   
## 1st Qu.: 16.52 2:158   
## Median : 38.28 3: 97   
## Mean : 42.00   
## 3rd Qu.: 67.66   
## Max. : 99.78

A continuación vamos a **caracterizar los grupos** formados en función de las medias de las variables originales (coordenadas de los centroides). Así, se podrán mostrar en pantalla las medias de cada grupo de las distintas variables originales, usando las funciones by\_group() y summarise() de {dplyr}. Los decimales se ajustarán utilizando la función round(). Toda la información se asigna al *data frame* “tablamedias”, para poder posteriormente representado en una tabla mediante las facilidades que ofrecen los paquetes {knitr} y {kableExtra}:

# CARACTERIZANDO GRUPOS FORMADOS  
   
 # Tabla con centroides de grupos.  
  
 tablamedias <- originales\_so %>%  
 group\_by(whatcluster\_k) %>% summarise(obs = length(whatcluster\_k),  
 Resultado = round(mean(RES),0),  
 Margen = round(mean(MARGEN),2),  
 Fondos\_Propios = round(mean(FPIOS),0),  
 Solvencia = round(mean(SOLVENCIA),2))  
  
 library (knitr)  
 library (kableExtra)  
 knitr.table.format = "html"  
  
 tablamedias %>%  
 kable(format = knitr.table.format,  
 caption = "Método de k-medias. 3 grupos. Medias de variables",  
 col.names = c("Clúster", "Observaciones", "Resultado", "Margen",  
 "Fondos Propios", "Solvencia")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped", "bordered", "condensed",  
 position = "center",  
 font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:nrow(tablamedias), bold= F, align = "c")

(#tab:unnamed-chunk-241)(#tab:unnamed-chunk-241)Método de k-medias. 3 grupos. Medias de variables

Clúster

Observaciones

Resultado

Margen

Fondos Propios

Solvencia

1

59

6035

68.85

28311

57.76

2

158

708

18.07

3745

17.88

3

97

1050

54.42

4612

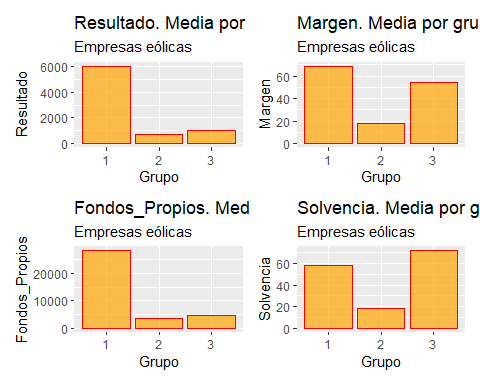
71.70

Obviamente, también se podrían comparar las medias de los grupos, para cada variable, con un gráfico de barras. El código es el siguiente:

# Gráficos de centroides  
  
 # Vector de nombre de variables excluyendo la variable no deseada  
 variables <- setdiff(names(tablamedias), c("whatcluster\_k", "obs"))  
  
 # Lista para almacenar los gráficos  
 graficos.centroides <- list()  
  
 # Bucle para crear y almacenar los gráficos  
 for (i in seq\_along(variables)) {  
 var1 <- variables[[i]]  
 grafico <- ggplot(data= tablamedias,  
 map = (aes\_string(y = var1, x = "whatcluster\_k"))) +  
 geom\_bar(stat = "identity",  
 colour = "red",  
 fill = "orange",  
 alpha = 0.7) +  
 ggtitle(paste0(var1, ". Media por grupos."),  
 subtitle = "Empresas eólicas")+  
 xlab ("Grupo") +  
 ylab(var1)  
 graficos.centroides[[paste0("grafico\_", var1)]] <- grafico  
 }

Para hacer composiciones de 4 gráficos (que en este caso será solo una, dado que hemos almacenado en la lista “graficos.centroides” 4 elementos correspondientes a las 4 variables clasificadoras), volveremos a utilizar la función create\_patchwork(), que ya se mostró en el ejemplo de clúster jerárquico. De nuevo se incluye su código y se aplica a la lista de gráficos “graficos.centroides”:

# Función para crear agrupaciones de 2x2 con espacios en blanco si necesario  
  
 library (patchwork)   
  
 create\_patchwork <- function(plot\_list) {  
 n <- length(plot\_list)  
 full\_rows <- n %/% 4  
 remaining <- n %% 4  
   
 patchworks <- list()  
   
 # Crear agrupaciones completas de 2x2  
 for (i in seq(1, full\_rows \* 4, by = 4)) {  
 patchworks <- c(patchworks,  
 list((plot\_list[[i]] + plot\_list[[i+1]]) /  
 (plot\_list[[i+2]] + plot\_list[[i+3]])))  
 }  
   
 # Crear la última agrupación con espacios en blanco si es necesario  
 if (remaining > 0) {  
 last\_plots <- plot\_list[(full\_rows \* 4 + 1):n]  
 empty\_plots <- lapply(1:(4 - remaining),  
 function(x) ggplot() + theme\_void())  
 last\_patchwork <- do.call(patchwork::wrap\_plots,  
 c(last\_plots, empty\_plots))  
 patchworks <- c(patchworks, list(last\_patchwork))  
 }  
   
 return(patchworks)  
 }  
  
 # Aplicar función a gráficos de centroides.  
 grupos.graficos.centroides <- create\_patchwork(graficos.centroides)  
  
 # Presentar las composiciones  
 for (n in 1:length(grupos.graficos.centroides)){  
 print(grupos.graficos.centroides[[n]])  
 }



La lista “grupos.graficos.centroides” almacena las composiciones de 4 (o menos) gráficos generadas. En este ejemplo, solo tiene un elemento, que es mostrado al ejecutar el bucle de presentación de composiciones.

Por medio de los gráficos se pueden establecer algunas conclusiones, comparando las medias de las variables para cada grupo de casos formado (coordenadas de los centroides). Por ejemplo, el grupo 1 destaca por tener, en media, un resultado y unos fondos propios muy superiores a los del resto de grupos. En cuanto a la solvencia, también tiene en media el mayor valor, aunque la diferencia no parece tan pronunciada con respecto al grupo 3. El grupo 2 se caracteriza por tener, en media, los valores inferiores en las 4 variables clasificadoras, siempre de un modo que se antoja muy pronunciado. Por último, el grupo 3 tiene, en media, el mayor valor de solvencia y, en cuanto al margen, se sitúa relativamente cerca del margen medio del grupo 1.

El análisis gráfico anterior se puede complementar de un modo estadítico, a fin de verificar si las diferencias observadas entre los valores medios de los grupos, para cada variable, son significativas. Para ello, y teniendo en cuenta que los grupos se pueden considerar submuestras que representan a subpoblaciones, se puede aplicar alguna prueba de comparaciones múltiples de las medias de los grupos formados, de modo que se pueda confirmar, para cierta significación estadística (usualmente 0,05), **si las diferencias en las medias de los grupos para cada variable son estadísticamente relevantes** (significativas) o no.

Una prueba clásica para llevar a cabo esta tarea es aplicar el *test de comparaciones múltiples de Tuckey*. No obstante, y dado que esta prueba requiere del cumplimiento de ciertos requisitos previos (normalidad de los grupos, varianzas homogéneas); hemos optado por la ***prueba robusta de Kruskal-Wallis*.** Para cada una de las variable originales, realizaremos un gráfico múltiple de diagramas de caja, y procederemos a mostrar los resultados de la prueba.

En primer lugar, vamos a definir el vector con el nombre de las variables que entran en el análisis (vector “variables”), e inicializaremos las listas para guardar los gráficos y los resultados de la prueba para cada variable (“graficos\_kw” y “tablas\_kw”, respectivamente). También se activa el paquete {pgirmess}, que contiene la función para proceder a realizar la *prueba de comparaciones múltiples de Kruskal y Wallis*, kruskalmc():

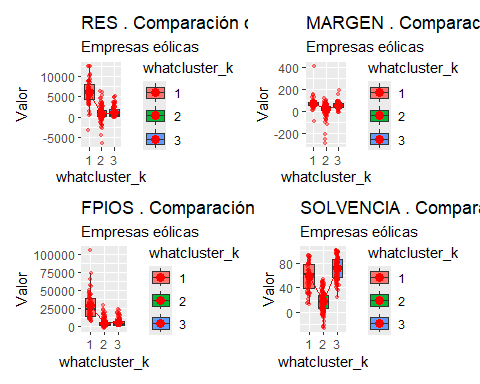
# ¿Centroides estadísticamente significativos?  
  
 # Vector de nombre de variables excluyendo la variable no deseada  
 variables <- setdiff(names(originales\_so), "whatcluster\_k")  
  
 # Inicializar listas para almacenar gráficos y tablas  
 graficos\_kw <- list()  
 tablas\_kw <- list()  
 library(pgirmess)

Luego, haremos un bucle para que se realice el gráfico de caja y se presente en una tabla el resultado de la *prueba de comparaciones múltiples de Kruskal y Wallis* para cada una de las variables. En la función para realizar la prueba, kruskalmc(), la solución se guarda en un objeto, por ejemplo, “datos\_kmc”, y los argumentos son, por un lado, la variable analizada y el factor que se utiliza para denominar los grupos (“whatcluster\_k”), unidos por el símbolo “~”:

# Bucle para generar gráficos y análisis  
 for (i in seq\_along(variables)) {  
 variable <- variables[i]  
   
 # Crear el gráfico  
 p <- ggplot(data = originales\_so,  
 aes\_string(x = "whatcluster\_k",  
 y = variable,  
 fill = "whatcluster\_k")) +  
 geom\_boxplot(outlier.shape = NA) +  
 stat\_summary(fun = "mean",  
 geom = "point",  
 size = 3,  
 col = "red") +  
 stat\_summary(fun = "mean",  
 geom = "line",  
 col = "red",  
 aes(group = TRUE)) +   
 geom\_jitter(width = 0.1,  
 size = 1,  
 col = "red",  
 alpha = 0.40) +  
 ggtitle(paste(variable, ". Comparación de grupos."),  
 subtitle = "Empresas eólicas") +  
 ylab("Valor")  
   
 # Almacenar el gráfico en la lista  
 graficos\_kw[[i]] <- p  
   
 # Realizar el análisis de Kruskal-Wallis  
 datos\_kmc <- kruskalmc(as.formula(paste(variable, "~ whatcluster\_k")),  
 data = originales\_so)  
 tabla <- datos\_kmc$dif.com %>%  
 kable(format = knitr.table.format,  
 caption = paste("k-medias. Diferencias de Centroides",  
 variable),  
 col.names = c("Diferencias centros",  
 "Diferencias críticas",  
 "Significación")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = c("striped",  
 "bordered",  
 "condensed"),  
 position = "center",  
 font\_size = 11) %>%  
 row\_spec(0, bold = T, align = "c") %>%  
 row\_spec(1:nrow(datos\_kmc$dif.com), bold = F, align = "c")  
   
 # Almacenar la tabla en la lista  
 tablas\_kw[[i]] <- tabla  
 }

Una vez almacenados, para cada variable clasificadora, los gráficos de caja y las tablas con los resultados de la prueba de comparaciones múltiples, quedan por realizar dos tareas: Agrupar los gráficos en composiciones de 4 (o menos) mediante la función create\_patchwork() en la lista “gruposgraficos\_kw”, y presentar todos los elementos:

# Crear composiciones de gráficos.  
  
 gruposgraficos\_kw <- create\_patchwork(graficos\_kw)  
  
 # Mostrar los gráficos y tablas almacenados  
  
 for (i in seq\_along(gruposgraficos\_kw)) {  
 print(gruposgraficos\_kw[[i]])  
 }



for (i in seq\_along(tablas\_kw)) {  
 print(tablas\_kw[[i]])  
 }

De los resultados del análisis de comparaciones múltiples de Kruskal y Wallis obtenidos, y teniendo en cuenta una significación estadística de 0,05; se obtienen las siguientes conclusiones:

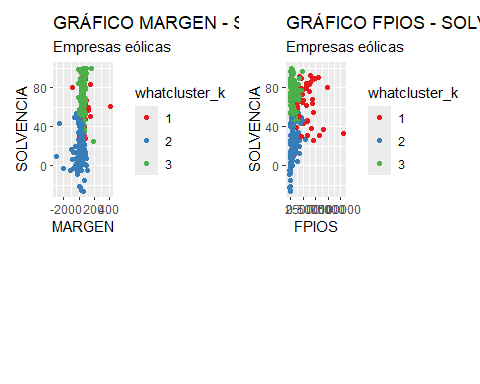
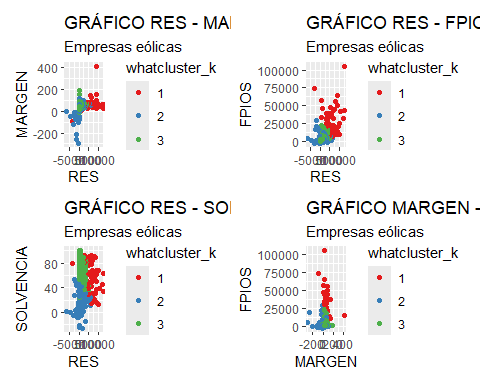
* Variable de resultado (RES): en términos medios, solo existen diferencias estadísticamente significativas entre los resultados del ejercicio de las empresas del grupo 1, con respecto a los del grupo 2 y 3.
* Variable de margen de beneficio (MARGEN): en términos medios, solo existen diferencias estadísticamente significativas entre el margen de beneficio de las empresas del grupo 2, en relación con los márgenes de las empresas de los grupos 1 y 3.
* Variable de fondos propios (FPIOS): en términos medios, solo existen diferencias estadísticamente significativas en el volumen de fondos propios de las empresas del grupo 1, con respecto a los del grupo 2 y 3.
* Variable de solvencia (SOLVENCIA): en términos medios, todas las diferencias en el grado de solvencia de las empresas de los distintos grupos son estadísticamente significativas.

Finalmente. al centrarse nuestro objetivo en obtener perfiles de empresas más que en clasificar empresas concretas, y puesto que la muestra es numerosa, carece de sentido crear tablas con la información detallada de las empresas que componen cada uno, si bien la construcción de estas tablas es fácil, pudiéndose emplear el código que para tal objetivo se expuso en el apartado del análisis jerárquico. Lo que sí puede ser más interesante es mostrar como los elementos de cada grupo se disponen en los diagramas de dispersión producto de cruzar las variables clasificadoras entre sí. Para generar estos gráficos de un modo automatizado, primero generaremos un vector con el nombre de todas las variables (excluyendo al *factor* whatcluster\_k mediante la función setdiff(). Luego, crearemos una lista para ir almacenando los gráficos (lista “graficos”).Después, calcularemos todas las combinaciones de nombres de variables posibles, con la función combn(), y las almacenaremos en la lista “combinaciones”. En esta función, el argumento **simplify = FALSE** le dice a la función que no vuelque el resultado a una matriz:

# GRÁFICOS Variable vs Variable  
  
 # Lista de variables excluyendo la variable no deseada  
 variables <- setdiff(names(originales\_so), "whatcluster\_k")  
  
 # Lista para almacenar los gráficos  
 graficos <- list()  
  
 # Generar todas las combinaciones posibles de pares de variables  
 combinaciones <- combn(variables, 2, simplify = FALSE)

Una vez almacenados los elementos anteriores, procederemos a crear los gráficos de dispersión mediante un bucle. Estos gráficos se guardarán en la lista “graficos”, que se pasará por la función create\_patchwork() para que se sinteticen en composiciones de 4 (o menos) gráficos. Estas composiciones se almacenan en la lista “gruposgraficos”, y se presentan mediante un nuevo bucle:

# Bucle para crear y almacenar los gráficos  
 for (i in seq\_along(combinaciones)) {  
 var1 <- combinaciones[[i]][1]  
 var2 <- combinaciones[[i]][2]  
 grafico <- ggplot(originales\_so,  
 map = aes\_string(x = var1,  
 y = var2,  
 color = "whatcluster\_k")) +  
 geom\_point() +  
 labs(title = paste("GRÁFICO", var1, "-", var2),  
 subtitle = "Empresas eólicas") +  
 xlab (var1) +  
 ylab (var2) +  
 scale\_color\_brewer(palette = "Set1")   
 graficos[[paste0("grafico\_", var1, "\_", var2)]] <- grafico  
 }  
  
 # Aplicar función de composiciones de patchwork  
  
 gruposgraficos <- create\_patchwork(graficos)  
  
 # Presentar las composiciones  
 for (n in 1:length(gruposgraficos)){  
 print(gruposgraficos[[n]])  
 }



A partir de los gráficos anteriores se aprecia cómo, en coherencia con los resultados obtenidos en el análisis de las medias de los grupos, las empresas del grupo 1 poseen, en general valores en las variables de resultados y fondos propios superiores a los del resto de grupos (gráficos 1, 2, 3, 4 y 6). En cuanto al margen, el grueso de las empresas de los tres grupos poseen márgenes muy parecidos, con una leve tendencia de algunas empresas del grupo 3 a ser menores, y del grupo 1 a ser mayores (gráficos 1, 4 y 5). Con referencia a la solvencia, se distingue con claridad que las empresas del grupo 3 tienen, en general, un mayor nivel, destacando también la dispersión de la variable a lo largo del grupo 1 (gráficos 3, 5 y 6).

## 7.4 Materiales para realizar las prácticas del capítulo.

En esta sección se muestran los links de acceso a los diferentes materiales (*scripts*, datos…) necesarios para llevar a cabo los contenidos prácticos del capítulo.

**Datos (en formato Microsoft (R) Excel (R)):**

* eolica\_25.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1iwD8-e19IM6n_ZykP49i69_f_dLpzosx/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true))
* eolica\_350.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1Brtu_58lcfXL5ZTeOi9-xlxy7S9NfuxB/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true))

**Scripts:**

* cluster\_eolica.R ([obtener aquí](https://drive.google.com/file/d/1pxGjprMpfP_yqNsaJ-oWKJtfVnSbVZUh/view?usp=sharing))
* kmedias\_eolica.R ([obtener aquí](https://drive.google.com/file/d/1nOBc99SsMKYACitAVRD0nnpP8q-8FqJZ/view?usp=sharing))

# 8 Análisis de la varianza.

## 8.1 Introducción.

El análisis de la varianza (ANOVA) puede considerarse una generalización del contraste de hipótesis de medias poblacionales iguales para el caso de poblaciones normales y con varianzas desconocidas, pero iguales. La generalización consiste en poder considerar más de dos poblaciones. La **hipótesis nula** será la que afirma que **las medias poblacionales de la variable métrica en estudio, para todas las poblaciones, son iguales**. La hipótesis alternativa, por su lado, afirmará que existe al menos dos poblaciones con medias diferentes. Como todo contraste, para llevarlo acabo hemos de tener una muestra representativa de cada población, y fijar un nivel de significación (usualmente 0.05).

También se puede considerar el ANOVA como un **tipo especial de análisis de regresión**, en la que la variable dependiente es una variable métrica, y las variables explicativas son atributos o factores (en escala nominal u ordinal). La misión de los factores es clasificar a los casos que constituyen nuestra muestra en distintas submuestras, cada una representativa de una de las subpoblaciones cuyas medias en la variable en estudio se quiere comparar.

## 8.2 ANOVA de un solo factor.

Aunque se pueden realizar ANOVAs con más de un atributo o factor, en este ejemplo nos ceñiremos al caso más simple, en el que solo hay un atributo o factor que se ocupa de distribuir los casos de la muestra entre los distintos grupos o submuestras (a partir de las categoría o nivel que toma cada caso).

En concreto, en esta práctica, comprobaremos **si** **la dimensión del grupo empresarial al que pertenecen las empresas eólicas** (medida en función del número de empresas integradas en el grupo empresarial, y concretada en el factor DIMENSION) **tiene una influencia significativa sobre la rentabilidad económica** (variable RENECO), **en términos medios**. Para ello se ha seleccionado una muestra constituida por 50 empresas productoras de electricidad mediante tecnología eólica. Así, la población, constituida por todas las empresas de generación eléctrica eólica de España, queda dividida en tres subpoblaciones: la subpoblación de empresas que pertenecen a grupos empresariales de DIMENSION (según el número de filiales contenidas) “GRANDE”, la subpoblación de empresas que pertenecen a grupos empresariales de DIMENSION “MEDIA”, y la subpoblación de empresas que pertenecen a grupos empresariales de DIMENSION “PEQUEÑA”. Cada una de estas subpoblaciones tendrán sus respectivas rentabilidades económicas medias, que desconocemos (ya que no tenemos los datos de la población, es decir, de todas las empresas eólicas del país; sino solo de una muestra de 50 empresas). Lo que si tenemos para cada subpoblación es una submuestra que la representa (parte de las 50 empresas de la muestra, que queda fraccionada en tres según el factor DIMENSION). Y de cada submuestra, tenemos la correspondiente rentabilidad media muestral. Lo que comprobaremos con el contraste de ANOVA, en definitiva, es si las diferencias observadas entre las rentabilidades medias de cada submuestra son lo suficientemente amplias como para pensar que, puede considerarse que existen diferencias importantes (significativas) entre las rentabilidades medias de las subpoblaciones (considerando todas las empresas eólicas que conforman la población).

Los datos están almacenados en el archivo de Microsoft Excel “[eolica\_50.xls](https://docs.google.com/spreadsheets/d/1PVRwLo1eAbFLOgSscLZw0bD9rgMk2X-w/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true)”, y el script con el código del ejemplo se halla contenido en el fichero “[anova\_cluster.R](https://drive.google.com/file/d/1p4p17OudRprwkpF7wvvF1gwHZG82uv89/view?usp=sharing)”. Vamos a suponer que trabajaremos en un proyecto de RStudio al que denominaremos “anova”.

Una vez abierto el *script* en el editor de RStudio , comprobaremos que la primera parte del código está dedicada a la limpieza de la memoria (*Environment*) y a la importación de los datos. Para ello, activaremos el paquete {readxl} y utilizaremos la función read\_excel(), indicando en los argumentos el archivo a explorar, y la *hoja* en la cual se encuentran los datos (hoja “Datos”). También hemos de prestar atención a la cuestión de si existen en la hoja de *Excel* anotaciones en las celdas donde no haya dato, para completar adecuadamente el argumento na= . Los datos se almacenarán en el *data frame* “datos”. En este *data frame*, la primera columna no es una verdadera variable, sino que se compone de los nombres de los casos o empresas. Con una línea de código adicional transformaremos esa primera columna en el *nombre de las filas*, de modo que tal columna abandona su *rol* de variable. En definitiva, el código para importar los datos es:

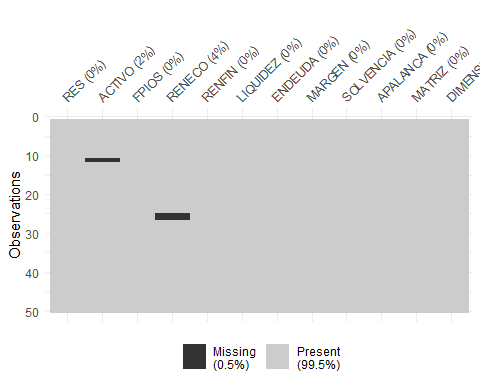
# Análisis ANOVA de un factor.  
  
rm(list = ls())  
  
# DATOS  
  
library (readxl)  
datos <- read\_excel("eolica\_50.xlsx", sheet = "Datos",  
 na = c("n.d.", "s.d."))  
datos <- data.frame(datos, row.names = 1)  
summary (datos)

## RES ACTIVO FPIOS   
## Min. :-5268.6 Min. : 25355 Min. : -10985   
## 1st Qu.: 919.6 1st Qu.: 35512 1st Qu.: 1897   
## Median : 2321.2 Median : 50301 Median : 17256   
## Mean : 4986.7 Mean : 189142 Mean : 83164   
## 3rd Qu.: 4380.4 3rd Qu.: 101035 3rd Qu.: 44871   
## Max. :42737.0 Max. :2002458 Max. :1740487   
## NA's :1   
##   
## RENECO RENFIN LIQUIDEZ   
## Min. :-2.708 Min. :-263.639 Min. : 0.0140   
## 1st Qu.: 1.817 1st Qu.: 1.251 1st Qu.: 0.6462   
## Median : 3.957 Median : 14.322 Median : 1.1550   
## Mean : 5.758 Mean : 32.071 Mean : 4.2370   
## 3rd Qu.: 8.038 3rd Qu.: 36.193 3rd Qu.: 1.9185   
## Max. :35.262 Max. : 588.190 Max. :128.4330   
## NA's :2   
##   
## ENDEUDA MARGEN SOLVENCIA   
## Min. : 0.917 Min. :-2248.16 Min. :-24.465   
## 1st Qu.: 37.272 1st Qu.: 14.74 1st Qu.: 4.327   
## Median : 74.683 Median : 23.86 Median : 25.317   
## Mean : 66.646 Mean : -17.77 Mean : 33.353   
## 3rd Qu.: 95.672 3rd Qu.: 45.88 3rd Qu.: 62.727   
## Max. :124.465 Max. : 400.90 Max. : 99.082   
##   
## APALANCA MATRIZ DIMENSION   
## Min. :-6905.772 Length:50 Length:50   
## 1st Qu.: 7.586 Class :character Class :character   
## Median : 126.208 Mode :character Mode :character   
## Mean : 784.430   
## 3rd Qu.: 763.952   
## Max. :12244.351

Antes de proceder al *análisis de la varianza* propiamente dicho, hemos de preparar nuestros datos mediante la localización de *missing values* y *outliers*. Los *outliers*, en el ANOVA, pueden tener una gran influencia (ya que se trabaja en *términos medios*) sobre los resultados, por lo que deben ser tratados convenientemente.

Para localizar los casos concretos de ***missing values***, puede recurrirse a utilizar las herramientas de manejo de *data frames* del paquete {dplyr}. Previamente, realizaremos una **copia** del *data frame* original, “datos”, a la que llamaremos “muestra”, que es con la que trabajaremos (para mantener la integridad del primer *data frame*). Con la función vismiss() del paquete {visdat} podemos tener una visión gráfica general de los valores faltantes, en especial en el caso de la variable RENECO y el factor DIMENSION. Si hay casos faltantes en una de estas variables, los identificaremos filtrando el *data frame* con la función filter() de {dplyr}. Ante la existencia de *missing values*, se puede actuar de varios modos. Por ejemplo, **se puede intentar obtener por otro canal de información el conjunto de valores** de RENECO que no están disponibles, **o recurrir a alguna estimación** para los mismos y asignarlos. En caso de que esto sea difícil, se puede optar, simplemente, por **eliminar** estos casos, en especial cuando representan un porcentaje muy reducido respecto al total de casos. En nuestro ejemplo, vamos a suponer que hemos optado por esta última vía, al no conseguir unos valores más o menos verosímiles de RENECO para las empresas de las que se carece de dato. Así, estos casos, finalmente, será exluidos del análisis, utilizando para ello un nuevo filtro:

# Missing values  
  
 library (dplyr)  
 library(visdat)  
 vis\_miss(datos)



datos %>% filter(is.na(RENECO) | is.na(DIMENSION)) %>%  
 select(RENECO, DIMENSION)

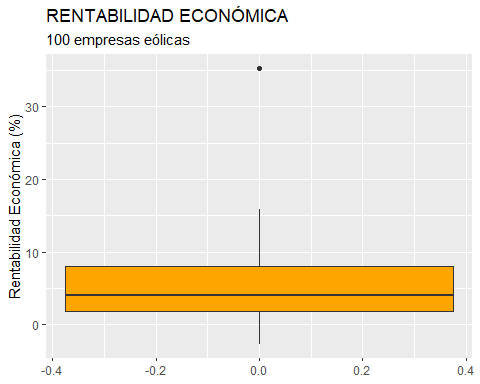
## RENECO DIMENSION  
## Sargon Energias SLU NA MEDIA  
## Viesgo Renovables SL. NA MEDIA

muestra <- datos %>%  
 filter(! is.na(RENECO) & ! is.na(DIMENSION))

Tras el código anterior, se han eliminado del *data frame* “muestra” las empresas “Sargon Energías S. L. U.” y “Viesgo Renovables S. L.”, empresas pertenecientes a grupos empresariales de dimensión media (DIMENSION), debido a que carecían de dato de rentabilidad económica (RENECO).

Una vez tratados los casos con valores perdidos o *missing values*, **es necesario detectar la presencia de *outliers*** o casos atípicos en la muestra, que pudieran desvirtuar los resultados derivados del ANOVA. Para ello, realizaremos un ***boxplot*** o gráfico de caja. Aplicaremos, por ejemplo, el código siguiente, que utiliza la gramática del paquete {ggplot2}:

# Outliers  
  
 library (ggplot2)  
 ggplot(data = muestra,  
 map = (aes(y = RENECO))) +  
 geom\_boxplot(fill = "orange") +  
 ggtitle("RENTABILIDAD ECONÓMICA",  
 subtitle = "100 empresas eólicas") +  
 ylab("Rentabilidad Económica (%)")



En el gráfico de caja se aprecia claramente que existe un *outlier*. Para identificar tal empresa, calcularemos, respecto a la variable RENECO, el primer y tercer cuartiles, que nos servirán para construir el filtro que capturará el caso atípico. Así, en el código, las dos primeras filas calculan los cuartiles primero (Q1) y tercero (Q3) mediante la función quantile(). Luego se filtran, mediante la función filter() de {dplyr}, los *outliers*, calculadoscomo aquellos casos con valores de RENECO mayores que Q3 más 1,5 veces el *rango intercuartílico* de la variable; o menores que Q1 menos 1,5 veces dicho rango intercuartílico. Para calcular el rango intercuartílico se recurre a la función IQR(). Finalmente, con select() se muestran los casos en la consola de RStudio:

Q1 <- quantile (muestra$RENECO, c(0.25))  
 Q3 <- quantile (muestra$RENECO, c(0.75))  
  
 muestra %>%  
 filter(RENECO > Q3 + 1.5\*IQR(RENECO) |  
 RENECO < Q1 - 1.5\*IQR(RENECO)) %>%  
 select(RENECO)

## RENECO  
## Molinos Del Ebro SA 35.262

La empresa identificada como outlier es “Molinos del Ebro S. A”. Como ocurría con los *missing values*, el tratamiento de los *outliers* depende de la información que se tenga, existiendo varias alternativas (corrección del dato, estimación, etc.) Si no se tiene información fiable, y los *outliers* no representan una gran proporción respecto al total de casos, puede optarse por su eliminación de la muestra, como haremos en este ejemplo. Podemos hacerlo creando un nuevo *data frame* a partir de “muestra”; pero sin ese caso. Ese nuevo *data frame* se llamará, por ejemplo, “**muestra\_so**”:

muestra\_so <- muestra %>% filter(RENECO <= Q3 + 1.5\*IQR(RENECO) &  
 RENECO >= Q1 - 1.5\*IQR(RENECO))

Una vez preparados los datos, vamos a presentar los grupos de empresas eólicas y sus rentabilidades económicas medias. Para ello, diseñaremos una tabla con la función kable() del paquete {knitr}, y personalizada con algunas funciones incluidas en el paquete {kableExtra}. Para construir la tabla, hemos de crear anteriormente un pequeño *data frame*, llamado por ejemplo “tablamedias”, en el que cada caso o fila sea uno de los grupos en que queda dividida la muestra a partir de los niveles del factor “DIMENSION”, y que contenga tres variables: la DIMENSION de cada grupo o submuestra, su número de casos contenidos (variable “observaciones”), y las respectivas rentabilidades medias (variable “media”):

# Visualizando número de frecuencias y medias de los grupos con dplyr:  
   
 library (knitr)  
 library (kableExtra)  
 knitr.table.format = "html"   
  
 tablamedias <- muestra\_so %>%  
 group\_by(DIMENSION) %>%  
 summarise (observaciones = length(DIMENSION),  
 media = mean(RENECO))  
  
 tablamedias %>%  
 kable(format = knitr.table.format,  
 caption = "Rentabilidad Económica. Medias por grupos (tamaño matriz).",  
 col.names = c("Tamaño", "Observaciones", "Rentabilidad Económica")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped", "bordered", "condensed",  
 position = "center",  
 font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:nrow(tablamedias), bold= F, align = "c")

(#tab:unnamed-chunk-257)(#tab:unnamed-chunk-257)Rentabilidad Económica. Medias por grupos (tamaño matriz).

Tamaño

Observaciones

Rentabilidad Económica

GRANDE

24

2.721167

MEDIA

13

7.241154

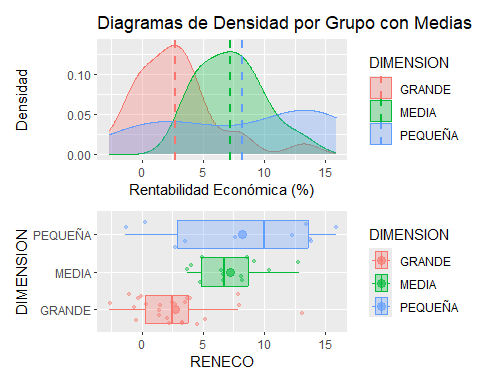
PEQUEÑA

10

8.167200

Gráficamente, las tres submuestras de empresas pueden caracterizarse, en cuanto a la rentabilidad económica (RENECO), mediante dos tipos de gráficos: gráficos de densidad y gráficos de caja. Los gráficos serán construidos utilizando las facilidades del paquete {ggplot2}. Además, combinaremos los gráficos en una composición utilizando el paquete {patchwork}. El código es el siguiente:

# Graficando casos y medias.  
  
 # Crear el gráfico de densidad  
 gdensidad <- ggplot(data= muestra\_so, aes(x = RENECO, color = DIMENSION, fill = DIMENSION)) +  
 geom\_density(alpha = 0.3) +  
 geom\_vline(data = tablamedias, aes(xintercept = media, color = DIMENSION), linetype = "dashed", size = 1) +  
 labs(title = "Diagramas de Densidad por Grupo con Medias",  
 x = "Rentabilidad Económica (%)",  
 y = "Densidad") +  
 theme\_grey()  
   
 # Crear box-plot  
 gbox <- ggplot(data = muestra\_so,  
 map = (aes(y = DIMENSION,  
 x = RENECO,  
 color = DIMENSION,  
 fill = DIMENSION))) +  
 geom\_boxplot(outlier.shape = NA,  
 alpha = 0.3) +  
 stat\_summary(fun = "mean",  
 geom = "point",  
 size = 3,  
 map = aes(col = DIMENSION),  
 alpha = 0.60) +  
 geom\_jitter(width = 0.1,  
 size = 1,  
 map = (aes(col = DIMENSION)),  
 alpha = 0.40) +  
 labs(tittle ="Diagramas de caja por Grupo con Medias",  
 xlab = "Rentabilidad Económica (%)",  
 ylab = "Submuestras")  
  
 # Combinar gráficos.  
   
 library(patchwork)  
   
 gdensidad / gbox



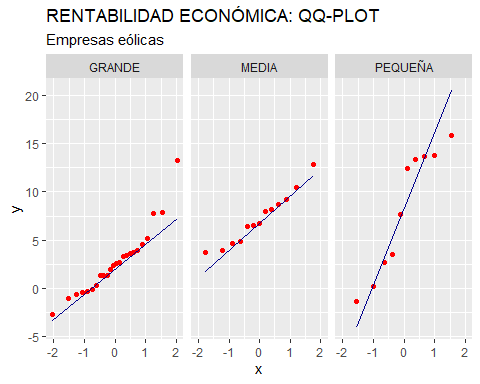
Cada grupo o submuestra “representa” a una subpoblación, según la dimensión que tenga la matriz empresarial. El ANOVA lo que intenta determinar es si, observadas las diferencias en las medias muestrales de la variable en estudio (aquí, RENECO), esas diferencias pueden considerarse o no estadísticamente significativas a nivel poblacional. En realidad, la hipótesis nula a contrastar es que ninguna de las diferencias entre las medias de la variable de las subpoblacionales es estadísticamente significativa; mientras que la hipótesis alternativa es que existe al menos una diferencia significativa entre las medias de las subpoblaciones. En definitiva, la **hipótesis nula** a contrastar sugiere **que no existen diferencias entre las rentabilidades económicas medias de los grupos** (subpoblaciones) de empresas eólicas (discriminadas por el tamaño del grupo empresarial al que pertenecen). De ser así, el factor DIMENSION no tendría una influencia estadísticamente significativa sobre el valor medio de la variable RENECO.

Las conclusiones a las que lleguemos con el *contraste F de ANOVA* serán válidas en la medida en que se cumplan las **hipótesis de normalidad y homogeneidad en las varianzas** de la variable dependiente o métrica (RENECO), en los tres grupos o subpoblaciones en que queda dividida la población atendiendo los niveles o categorías del factor (DIMENSION). Hemos de contrastar, pues, ambas hipótesis, a partir de la información de las tres submuestras que tenemos y que representan, respectivamente, a cada una de esas subpoblaciones.

En cuanto a la hipótesis de **normalidad**, se pueden usar dos vías para verificar su cumplimiento: el análisis gráfico y la realización de contrastes estadísticos.

El análisis gráfico puede llevarse a cabo mediante la realización *de gráficos QQ* de la variable RENECO, a partir de las submuestras o grupos de empresas en que queda fragmentada la muestra a partir de la variable DIMENSION. Estos gráficos pueden generarse con la gramática del paquete {ggplot2}:

# PRERREQUISITOS / HIPÓTESIS ANOVA   
  
 # Normalidad / Gráfico QQ  
  
 ggplot(data = muestra\_so,  
 aes(sample = RENECO)) +  
 stat\_qq(colour = "red") +   
 stat\_qq\_line(colour = "dark blue") +  
 ggtitle("RENTABILIDAD ECONÓMICA: QQ-PLOT",  
 subtitle = "Empresas eólicas") +  
 facet\_grid(. ~ DIMENSION)



Los *gráficos QQ* parecen inidicar que la variable RENECO tiene un comportamiento distante a la Ley Normal en las submuestras (y, por tanto, en las correspondientes subpoblaciones), al localizarse, algunos de los puntos, relativamente alejados de la diagonal, especialmente en el caso de las empresas pertenecientes a matrices de dimensión “grande”. Para extraer una conclusión de un modo más preciso, vamos a realizar el contraste de normalidad de *Shapiro-Wilk*:

# Normalidad: Shapiro-Wilk para cada grupo  
 normalidad <- muestra\_so %>%  
 group\_by(DIMENSION) %>%  
 summarise(shapiro\_p\_value = round(shapiro.test(RENECO)$p.value, 3)) %>%  
 mutate(decide = if\_else(shapiro\_p\_value > 0.05,  
 "NORMALIDAD",  
 "NO-NORMALIDAD"))  
  
 tablashapiro <- normalidad %>%  
 kable(format = knitr.table.format,  
 caption = "Normalidad (Shapiro-Wilks)",  
 col.names = c("Dimensión", "p-valor", "Conclusión")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped", "bordered", "condensed",  
 position = "center",  
 font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:nrow(normalidad), bold= F, align = "c")  
  
 tablashapiro

En el código anterior, se crea un pequeño *data frame* denominado “normalidad”, que incluye, para cada submuestra definida por los niveles del factor DIMENSION, la variable “decide” con los *p-valores* de la prueba de *Shapiro-Wilk*, y un atributo llamado “decide”, creado mediante la función mutate() de {dplyr}, que adopta la categoría “NORMALIDAD” o “NO-NORMALIDAD” dependiendo de los p-valores. Posteriormente, el data frame “normalidad” se presenta como una tabla de nombre “tablashapiro”.

(#tab:unnamed-chunk-261)(#tab:unnamed-chunk-261)Normalidad (Shapiro-Wilks).

Dimensión

p-valor

Conclusión

GRANDE

0.038

NO-NORMALIDAD

MEDIA

0.698

NORMALIDAD

PEQUEÑA

0.134

NORMALIDAD

En esta prueba, la hipótesis nula equivale al supuesto de normalidad. Para un 5% de significación estadística, un **p-valor superior a 0,05 implicará el no-rechazo de la hipótesis nula de normalidad**. En el ejemplo, la submuestra de empresas pertenecientes a matrices de dimensión “grande” llevan a pensar que esta subpoblación no sigue una distribución normal. En los otros dos casos, en cambio, podemos aceptar la existencia de normalidad.

En cuanto a la **homogeneidad de las varianzas**, contrastamos este supuesto mediante la prueba de *Bartlett*.

# Homogeneidad en las varianzas  
  
 bartlett.test(muestra\_so$RENECO ~ muestra\_so$DIMENSION)

##   
## Bartlett test of homogeneity of variances  
##   
## data: muestra\_so$RENECO by muestra\_so$DIMENSION  
## Bartlett's K-squared = 9.389, df = 2, p-value = 0.009146

El p-valor es menor que 0,05; luego **se rechaza la hipótesis nula de homogeneidad de las varianzas de los grupos (subpoblaciones)**.

Puesto que en uno de los grupos (subpoblaciones) se rechaza la hipótesis de normalidad, y (sobre todo) puesto que no se puede considerar una dispersión similar en las tres subpoblaciones (varianzas homogéneas), los resultados de la prueba ***F de ANOVA*** pierden validez, y habría que optar por **una alternativa robusta**. No obstante, **a modo ilustrativo**, seguiremos adelante con la prueba *F de ANOVA*.

El contraste *F de ANOVA* de igualdad en las medias de la variable en estudio (rentabilidad económica, RENECO) de las distintas (sub)poblaciones (grupos de empresas según el tamaño del grupo empresarial de pertenencia) se realiza en R mediante la función aov(), que guardaremos, por ejemplo, como el objeto “Datos.aov”, y que se almacenará en forma resumida en la lista “summary\_aov”, de un solo elemento. Este elemento, que contiene la solución resumida del contraste ANOVA, se convierte en un *data frame* (de nombre “aov\_table”) con el objetivo de presentarlo como una tabla de kable():

# Test F de ANOVA  
  
Datos.aov <- aov(muestra\_so$RENECO ~ muestra\_so$DIMENSION)  
summary\_aov <- summary(Datos.aov)  
  
 # Extraer los resultados del ANOVA  
 aov\_table <- as.data.frame(summary\_aov[[1]])  
  
 # Convertir la tabla en una tabla de kable  
 aov\_table %>%  
 kable(format = knitr.table.format,  
 caption = "Resultados del ANOVA",  
 col.names = c("Grados Libertad",  
 "Suma cuadrados",  
 "Media suma cuadrados",  
 "Estadístico F",  
 "p-valor")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped", "bordered", "condensed",  
 position = "center",  
 font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:nrow(aov\_table), bold= F, align = "c")

(#tab:unnamed-chunk-264)(#tab:unnamed-chunk-264)Resultados del ANOVA

Grados Libertad

Suma cuadrados

Media suma cuadrados

Estadístico F

p-valor

muestra\_so$DIMENSION

2

289.4459

144.72296

8.720764

0.0006453

Residuals

44

730.1895

16.59522

NA

NA

El valor del estadístico F de ANOVA es de 8,721; con un p-valor asociado de 0,0006. Como el p-valor es menor que 0,05, **se rechaza la hipótesis nula de medias iguales**; por lo que podremos afirmar (para una significación del 5%) que **el tamaño o dimensión del grupo empresarial de pertenencia influye, en media, en la rentabilidad económica obtenida**.

## 8.3 Comparaciones múltiples.

Cuando el resultado del contraste *F de ANOVA* es de rechazo de la hipótesis nula, se presenta otra cuestión interesante. La hipótesis alternativa dice que existe al menos una diferencia entre las medias de dos grupos que es “importante”. Pero no tienen porque ser todas. Entonces, cabe preguntarse que diferencias concretas entre medias son estadísticamente significativas, y cuáles no. Para dilucidar esta cuestión se han desarrollado diversas pruebas de **comparaciones múltiples**. Una de ellas es la prueba *HSD de Tuckey*. Un modo de obtener los resultados de esta prueba es ejecutarla a partir de las funciones del paquete {emmeans}. En concreto, los resultados de la prueba *HSD de* *Tuckey* se obtendrán aplicando la función pairs() a un objeto creado previamente con la función emmeans(), al que hemos llamado, por ejemplo, “medias”, y que tiene como argumentos el nombre de nuestra solución del contraste ANOVA anterior y, entrecomillado, el nombre de la variable que actúa como factor que divide a la muestra en los tres grupos (submuestras) comparados (DIMENSION). Los resultados se han guardado en el objeto “pares”, que posteriormente se ha transformado en un *data frame* para poder ser presentado como una tabla de kable():

# COMPARACIONES MÚLTIPLES  
  
library(emmeans)  
medias <- emmeans(Datos.aov, "DIMENSION")  
pares <- pairs(medias)  
pares\_df <- as.data.frame(pares)  
pares\_df %>%  
 kable(format = knitr.table.format,  
 caption = "Resultados comparaciones múltiples",  
 col.names = c("Grupos",  
 "Diferencia estimada",  
 "Desviación Típica",  
 "Grados de libertad",  
 "Estadístico t",  
 "p-valor")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped", "bordered", "condensed",  
 position = "center",  
 font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:nrow(pares\_df), bold= F, align = "c")

(#tab:unnamed-chunk-266)(#tab:unnamed-chunk-266)Resultados comparaciones múltiples

Grupos

Diferencia estimada

Desviación Típica

Grados de libertad

Estadístico t

p-valor

GRANDE - MEDIA

-4.5199872

1.402862

44

-3.2219752

0.0066466

GRANDE - PEQUEÑA

-5.4460333

1.533294

44

-3.5518532

0.0026160

MEDIA - PEQUEÑA

-0.9260462

1.713498

44

-0.5404418

0.8518522

En la tabla generada, cada fila recoge la diferencia entre la rentabilidad media de las empresas incluidas en las submuestras de los distintos niveles de dimensión de los grupos empresariales a los que perteneces dichas empresas. En la última columna, se muestran los *p-valores.* La hipótesis nula implica que la diferencia entre las rentabilidades medias de los dos grupos implicados son tan pequeñas que pueden considerarse nulas. Por tanto, p-valores muy pequeños (menores que 0,05) implican un rechazo de esta hipótesis, y la admisión de que esas difrencias son significativamente distintas a 0, o sea, “importantes”. Por tanto, en el ejemplo se concluye que las rentabilidad media del grupo de empresas de matrices de dimensión “grande” difiere significativamente con respecto a las rentabilidades medias de los otros dos grupos. En cambio, las rentabilidades medias de los grupos de empresas de matrices de dimensión “media” y “pequeña” no difieren significativamente.

## 8.4 ¿Y si no se cumplen las condiciones para realizar el contraste F de ANOVA?

En el ejemplo anterior hemos visto cómo, aunque hemos seguido el procedimiento de realización del contraste F de ANOVA y la prueba HSD de Tuckey de comparaciones múltiples; no sé cumplían algunos de los requisitos necesarios para confiar en los resultados de estos contrastes: la normalidad de las subpoblaciones (medida a través de las submuestras) y la homogeneidad de las varianzas de estas.

Cuando esto ocurre, es necesario recurrir a técnicas robustas, puesto que el comportamiento de la variable analizada en las subpoblaciones (grupos) no se ajusta al necesario para poder aplicar los contrastes anteriores.

Una prueba robusta que puede suplir al contraste F de ANOVA es la de Kruskal-Wallis. En esta prueba, la hipótesis nula vuelve a ser que no existen diferencias significativas entre las medias de la variable estudiada de las diferentes subpoblaciones (representadas por las correspondientes submuestras), mientras que la hipótesis alternativa apuesta porquela existencia al menos una diferencia significativa. Para aplicar la prueba de Kruskal-Wallis en R, puede recurrirse a la función kruskal.test() del paquete {pgirmess}. Así, en nuestro ejemplo, tenemos:

#Cuando se incumplen las hipotesis de ANOVA (test robusto)  
  
library(pgirmess)  
  
Datos.K <- kruskal.test(muestra\_so$RENECO ~ muestra\_so$DIMENSION)  
Datos.K

##   
## Kruskal-Wallis rank sum test  
##   
## data: muestra\_so$RENECO by muestra\_so$DIMENSION  
## Kruskal-Wallis chi-squared = 14.718, df = 2, p-value = 0.0006368

Puede comprobarse cómo el *p-valor* es muy pequeño (menor a 0,05), por lo que hemos de rechazar la hipótesis nula de medias iguales y admitir que **existe al menos dos grupos (subpoblaciones) en los que la diferencia entre la rentabilidad económica media es significativa**.

De nuevo, si se rechaza la hipótesis nula, cabe preguntarse cuáles son los grupos o subpoblaciones concretas cuyas medias de rentabilidad económica son significativamente diferentes. Kruskal-Wallis desarrollaron también la versión robusta de la prueba *HSD de Tuckey*, que se incluye en {pgirmess}, con la función kruskalmc(). En el siguiente código se aplica la función, cuya solución se guarda en el objeto “Datos.kmc”. El elemento de la solución “dif.com”, que contiene las comparaciones entre las medias, se pasa a un *data frame* para poder ser mostrado como una tabla de kable():

# Comparaciones múltiples  
  
 Datos.kmc <- kruskalmc(muestra\_so$RENECO ~ muestra\_so$DIMENSION)  
  
 # Convertir los resultados a un data frame  
  
 Datos.kmc.df <- as.data.frame(Datos.kmc$dif.com)  
  
 Datos.kmc.df %>%  
 kable(format = knitr.table.format,  
 caption = "Kruskal-Wallis. Múltiples diferencias",  
 col.names = c("Diferencias medias",  
 "Diferencias críticas",  
 "Significación")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = c("striped",  
 "bordered",  
 "condensed"),  
 position = "center",  
 font\_size = 11) %>%  
 row\_spec(0, bold = T, align = "c") %>%  
 row\_spec(1:nrow(Datos.kmc.df), bold = F, align = "c")

(#tab:unnamed-chunk-269)(#tab:unnamed-chunk-269)Kruskal-Wallis. Múltiples diferencias

Diferencias medias

Diferencias críticas

Significación

GRANDE-MEDIA

16.381410

11.30376

TRUE

GRANDE-PEQUEÑA

13.758333

12.35473

TRUE

MEDIA-PEQUEÑA

2.623077

13.80676

FALSE

En la tabla generada se comprueba cómo las diferencias que la media de la rentabilidad económica de la subpoblación de empresas pertenecientes a matrices de dimensión “grande” son significativas (para un 0,05 de significación). En cambio, las rentabilidades económicas medias de las empresas pertenecientes a matrices de dimensiones “media” y “pequeña” no difieren entre sí de modo significativo.

## 8.5 Materiales para realizar las prácticas del capítulo.

En esta sección se muestran los links de acceso a los diferentes materiales (*scripts*, datos…) necesarios para llevar a cabo los contenidos prácticos del capítulo.

**Datos (en formato Microsoft (R) Excel (R)):**

* eolica\_50.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1PVRwLo1eAbFLOgSscLZw0bD9rgMk2X-w/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true))

**Scripts:**

* anova\_eolica.R [(obtener aquí](https://drive.google.com/file/d/1p4p17OudRprwkpF7wvvF1gwHZG82uv89/view?usp=sharing))

# 9 Análisis de Regresión Lineal Múltiple.

## 9.1 Introducción.

El análisis de regresión (lineal) múltiple es una de las técnicas de análisis de dependencias más profusamente utilizadas. En el modelo de regresión múltiple la variable dependiente tiene escala métrica. Las variables explicativas pueden ser métricas o ser atributos.

Según los datos de los que se alimenta el modelo, se aplicarán diferentes métodos de estimación, especificaciones y pruebas:

* Series temporales.
* Datos de corte transversal.
* Paneles de datos.

En este capítulo nos centraremos en los modelos estimados con base en datos de corte transversal (es decir, las variables tienen datos referentes a distintos casos o individuos: personas, empresas, países, etc.)

La construcción de un modelo de regresión cuenta con una serie de **etapas**, que son:

1. **Especificación del modelo:** establecer las variables que entrarán a formar parte del modelo (dependiente, explicativas).
2. **Estimación:** calcular el valor de los parámetros o coeficientes estructurales del modelo.
3. **Contraste y validación:** verificar si el modelo estimado cumple con las hipótesis que garantizan unas buenas propiedades y si es adecuado para representar la realidad.
4. **Utilización del modelo:** a efectos de previsión, análisis estructural o simulación de escenarios.

Vamos a partir del modelo básico de regresión (MBR). Es cierto que, para superar ciertas carencias de este, se ha procedido a desarrollar especificaciones y métodos de estimación más elaborados; pero no es menos cierto que no es conveniente “quemar” etapas sin conocer las características del modelo fundamental, como cimiento donde se posan modelados más complejos.

En el MBR vamos a suponer que existen:

* Una variable dependiente *y*.
* k variables explicativas .
* Variable o ***perturbación aleatoria*** u, que recoge el efecto conjunto de todas aquellas variables que afectan al comportamiento de *y* pero que no están explicitadas en la especificación como variables *x*.
* El tamaño de la muestra es *n*.

El modelo que se plantea es:

con .

O, en notación matricial:

Donde *y* es un vector (nx1), *X* una matriz (nxk), un vector (kx1), y *u* un vestor (nx1).

En el MBR, la perturbación aleatoria *u* debe cumplir con una serie de **hipótesis básicas**: normalidad en su comportamiento, esperanza nula, homoscedasticidad o varianza constante, y ausencia de autocorrelación (covarianza nula entre diferentes elementos del vector de la perturbación). Estas hipótesis, junto a las de permanencia estructural (valores de los elementos de constantes a lo largo de la muestra), no endogeneidad o regresores no-estocásticos (covarianza nula entre la matriz *X* y el vector *u*), y rango pleno (las columnas de la matriz X o variables explicativas no han de ser combinaciones lineales unas de otras); permiten que el MBR pueda ser estimado por el método de mínimos cuadrados ordinarios (MCO), obteniendo estimadores con las mejores propiedades: **insesgadez, eficiencia, consistencia**.

En la medida en que alguna o algunas de las hipótesis básicas no se cumplan, la calidad de los estimadores MCO perderan calidad, en el sentido de no gozar de las propiedades deseables, desde un punto de vista inferencial. En tal caso, podrán aplicarse otros métodos de estimación, diversos métodos econométricos, o asumir que los estimadores carecen de algunas de las propiedades deseables.

El modelo estimado será:

Y el **error o residuo** será, para cada observación, . El vector de residuos se considera una estimación del vector de perturbaciones aleatorias. Es por ello que el vector de residuos se utiliza para verificar el cumplimiento de las hipótesis del modelo básico referentes al comportamiento de la perturbación (normalidad, homoscedasticidad, ausencia de autocorrelación…)

Tras estas breves notas formales del MBR, pasaremos a construir un modelo que intentará explicar el comportamiento de la rentabilidad económica de un grupo de empresas en función de una serie de variables aleatorias.

## 9.2 Especificación del MBR. Datos de corte transversal.

Vamos a explicar, mediante un modelo de regresión múltiple, el comportamiento de la rentabilidad económica (RENECO) de las empresas de producción eléctrica mediante tecnología eólica en función del resultado del ejercicio (RES), el activo (ACTIVO), del grado de endeudamiento (ENDEUDA), del grado de apalancamiento (APALANCA), y del tamaño del grupo corporativo (matriz) al que pertenece (DIMENSION). Para ello se ha seleccionado una muestra constituida por 50 empresas.

Supondremos que trabajamos en un proyecto de RStudio de nombre, por ejemplo, “regresion”. Los datos de las empresas se encuentran en la hoja “Datos” del archivo de Microsoft® Excel® “[eolica\_50.xlsx](https://docs.google.com/spreadsheets/d/1oq17lI4trI8k5oeDgj_fYg9WhbRIqDLG/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true)”. El script con el código que vamos a ir ejecutando se llama “[regresion\_eolica.R](https://drive.google.com/file/d/1ilNEAA3-Co9kbs5Gv4arwCBCcCnbn0zY/view?usp=sharing)”.

En primer lugar, como de costumbre, hemos de preparar los datos: importarlos en R y gestionar *missing values* y *outliers*.

Así, una vez abierto el *script* en el editor de RStudio , comprobaremos que la primera parte del código está dedicada a la limpieza de la memoria (*Environment*) y a la importación de los datos. Para ello, activaremos el paquete {readxl} y utilizaremos la función read\_excel(), indicando en los argumentos el archivo a explorar, y la *hoja* en la cual se encuentran los datos (hoja “Datos”). También hemos de prestar atención a la cuestión de si existen en la hoja de *Excel* anotaciones en las celdas donde no haya dato, para completar adecuadamente el argumento na= . Los datos se almacenarán en el *data frame* “datos”. En este *data frame*, la primera columna no es una verdadera variable, sino que se compone de los nombres de los casos o empresas. Con una línea de código adicional transformaremos esa primera columna en el *nombre de las filas*, de modo que tal columna abandona su *rol* de variable. En definitiva, el código para importar los datos es:

## Regresion multiple empresas eolicas. Disculpen la falta de tildes.  
  
rm(list = ls())  
  
# DATOS  
  
 # Importando  
  
 library(readxl)  
 eolicos <- read\_excel("eolica\_50.xlsx", sheet = "Datos",  
 na = c("n.d.", "s.d."))  
 eolicos <- data.frame(eolicos, row.names = 1)  
 summary (eolicos)

## RES ACTIVO FPIOS   
## Min. :-5268.6 Min. : 25355 Min. : -10985   
## 1st Qu.: 919.6 1st Qu.: 35512 1st Qu.: 1897   
## Median : 2321.2 Median : 50301 Median : 17256   
## Mean : 4986.7 Mean : 189142 Mean : 83164   
## 3rd Qu.: 4380.4 3rd Qu.: 101035 3rd Qu.: 44871   
## Max. :42737.0 Max. :2002458 Max. :1740487   
## NA's :1   
##   
## RENECO RENFIN LIQUIDEZ   
## Min. :-2.708 Min. :-263.639 Min. : 0.0140   
## 1st Qu.: 1.817 1st Qu.: 1.251 1st Qu.: 0.6462   
## Median : 3.957 Median : 14.322 Median : 1.1550   
## Mean : 5.758 Mean : 32.071 Mean : 4.2370   
## 3rd Qu.: 8.038 3rd Qu.: 36.193 3rd Qu.: 1.9185   
## Max. :35.262 Max. : 588.190 Max. :128.4330   
## NA's :2   
##   
## ENDEUDA MARGEN SOLVENCIA   
## Min. : 0.917 Min. :-2248.16 Min. :-24.465   
## 1st Qu.: 37.272 1st Qu.: 14.74 1st Qu.: 4.327   
## Median : 74.683 Median : 23.86 Median : 25.317   
## Mean : 66.646 Mean : -17.77 Mean : 33.353   
## 3rd Qu.: 95.672 3rd Qu.: 45.88 3rd Qu.: 62.727   
## Max. :124.465 Max. : 400.90 Max. : 99.082   
##   
## APALANCA MATRIZ DIMENSION   
## Min. :-6905.772 Length:50 Length:50   
## 1st Qu.: 7.586 Class :character Class :character   
## Median : 126.208 Mode :character Mode :character   
## Mean : 784.430   
## 3rd Qu.: 763.952   
## Max. :12244.351

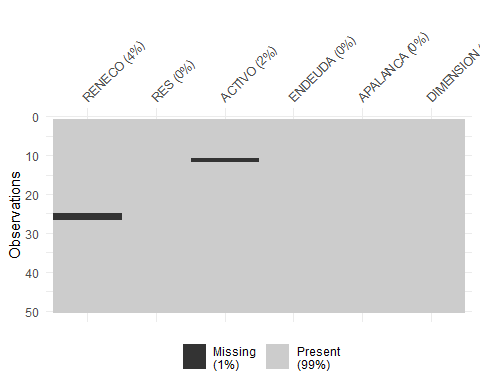
Posteriormente, seleccionaremos las variables que vamos a utilizar en el análisis, almacenándolas en otro data frame de nombre, por ejemplo, “originales”:

# Seleccionando variables clasificadoras para el analisis  
  
 library(dplyr)  
 originales<-select(eolicos,  
 RENECO,  
 RES,  
 ACTIVO,  
 ENDEUDA,  
 APALANCA,  
 DIMENSION)  
 summary (originales)

## RENECO RES ACTIVO   
## Min. :-2.708 Min. :-5268.6 Min. : 25355   
## 1st Qu.: 1.817 1st Qu.: 919.6 1st Qu.: 35512   
## Median : 3.957 Median : 2321.2 Median : 50301   
## Mean : 5.758 Mean : 4986.7 Mean : 189142   
## 3rd Qu.: 8.038 3rd Qu.: 4380.4 3rd Qu.: 101035   
## Max. :35.262 Max. :42737.0 Max. :2002458   
## NA's :2 NA's :1   
##   
## ENDEUDA APALANCA DIMENSION   
## Min. : 0.917 Min. :-6905.772 Length:50   
## 1st Qu.: 37.272 1st Qu.: 7.586 Class :character   
## Median : 74.683 Median : 126.208 Mode :character   
## Mean : 66.646 Mean : 784.430   
## 3rd Qu.: 95.672 3rd Qu.: 763.952   
## Max. :124.465 Max. :12244.351

Para localizar los casos concretos de ***missing values***, puede recurrirse a utilizar las herramientas de manejo de *data frames* del paquete {dplyr}. Con la función vismiss() del paquete {visdat} podemos tener una visión gráfica general de los valores faltantes. Si hay casos faltantes en una de las variables, los identificaremos filtrando el *data frame* con la función filter() de {dplyr}. Ante la existencia de *missing values*, se puede actuar de varios modos. Por ejemplo, **se puede intentar obtener por otro canal de información el conjunto de valores** de las variables que no están disponibles, **o recurrir a alguna estimación** para los mismos y asignarlos. En caso de que esto sea difícil, se puede optar, simplemente, por **eliminar** estos casos, en especial cuando representan un porcentaje muy reducido respecto al total de casos. En nuestro ejemplo, vamos a suponer que hemos optado por esta última vía. Así, estos casos, finalmente, será exluidos del análisis, utilizando para ello un nuevo filtro:

# Identificando missing values.  
  
 library(visdat)  
 vis\_miss(originales)



originales %>%  
 filter(is.na(RENECO) | is.na(RES) | is.na(ACTIVO) |  
 is.na(ENDEUDA) | is.na(APALANCA) | is.na(DIMENSION)) %>%  
 select(RENECO, RES, ACTIVO, ENDEUDA, APALANCA, DIMENSION)

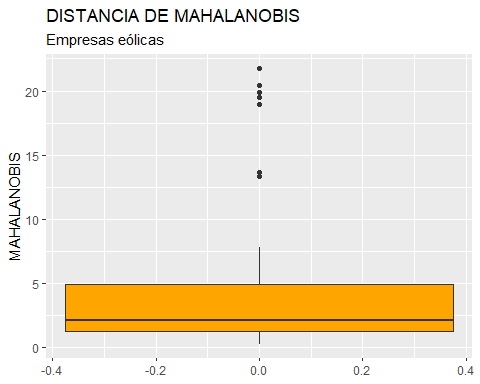
## RENECO RES ACTIVO ENDEUDA APALANCA DIMENSION  
## La Caldera Energia Burgos SL 2.643 511.304 NA 110.636 -1019.288 GRANDE  
## Sargon Energias SLU NA -2216.000 85745 112.811 -879.289 MEDIA  
## Viesgo Renovables SL. NA 4609.000 269730 34.116 13.330 MEDIA

originales <- originales %>%  
 filter(! is.na(RENECO) & ! is.na(RES) & ! is.na(ACTIVO) &  
 ! is.na(ENDEUDA) & ! is.na(APALANCA) & ! is.na(DIMENSION))

Tras el código anterior, se han eliminado del *data frame* “originales” las empresas “Sargon Energías S. L. U.” y “Viesgo Renovables S. L.”, debido a que carecían de dato de rentabilidad económica (RENECO), y “La Caldera Energía Burgos, S. L.”, al no tener dato sobre el valor de sus activos (ACTIVO).

Una vez tratados los casos con valores perdidos o *missing values*, **es necesario detectar la presencia de *outliers*** o casos atípicos en la muestra, que pudieran desvirtuar los resultados del análisis de regresión. Para realizar esta etapa, y dado que en nuestro análisis contamos con 5 variables, primero “resumiremos” el valor que toman dichas variables para cada caso, mediante el cálculo de la *distancia de Mahalanobis*. De hecho, las distancias de los diferentes casos se almacenarán en una nueva variable, a la que llamaremos MAHALANOBIS, que se incorporará al *data frame* “originales” por medio de la función mutate() de {dplyr}, y la función mahalanobis(). Recordemos que, en los diferentes argumentos de esta función, el punto “.” hace referencia al *data frame* que está delante del operador pipe (%>%). Posteriormente, construiremos el diagrama de caja de MAHALANOBIS para verificar la existencia de *outliers* (puntos), e identificaremoslos casos correspondientes con el filtro adecuado. Obtaremos por crear un nuevo *data frame*, “originales\_so”, aplicando un nuevo filtro que elimine esos casos localizados como *outliers*:

# Identificando outliers.  
  
 originales <- originales %>%  
 mutate(MAHALANOBIS = mahalanobis(select(.,  
 RENECO,  
 RES,  
 ACTIVO,  
 ENDEUDA,  
 APALANCA),  
 center = colMeans(select(.,  
 RENECO,  
 RES,  
 ACTIVO,  
 ENDEUDA,  
 APALANCA)),  
 cov=cov(select(.,  
 RENECO,  
 RES,  
 ACTIVO,  
 ENDEUDA,  
 APALANCA))))  
  
 library (ggplot2)  
 ggplot(data = originales, map = (aes(y = MAHALANOBIS))) +  
 geom\_boxplot(fill = "orange") +  
 ggtitle("DISTANCIA DE MAHALANOBIS", subtitle = "Empresas eólicas") +  
 ylab("MAHALANOBIS")



Q1M <- quantile (originales$MAHALANOBIS, c(0.25))  
 Q3M <- quantile (originales$MAHALANOBIS, c(0.75))  
   
 originales %>% filter(MAHALANOBIS > Q3M + 1.5\*IQR(MAHALANOBIS) |  
 MAHALANOBIS < Q1M - 1.5\*IQR(MAHALANOBIS)) %>%  
 select(MAHALANOBIS)

## MAHALANOBIS  
## Corporacion Acciona Eolica SL 13.67263  
## Parque Eolico Sierra De Las Carbas SL 13.34797  
## Naturgy Renovables SLU 19.45720  
## Global Power Generation SA. 19.88409  
## Saeta Yield SA. 20.40512  
## Molinos Del Ebro SA 21.73955  
## Elecdey Lezuza SA 18.97057

originales\_so <- originales %>%  
 filter(MAHALANOBIS <= Q3M + 1.5\*IQR(MAHALANOBIS) &  
 MAHALANOBIS >= Q1M - 1.5\*IQR(MAHALANOBIS))   
 originales <- originales %>% select(-MAHALANOBIS)  
 originales\_so <- originales\_so %>% select(-MAHALANOBIS)

Se han localizado 7 empresas con valores de la *distancia de Mahalanobis* atípicos, lo que hace pensar que, a su vez, estas empresas registran valores atípicos en una o varias de las variables originales. Como ya hemos señalado, eliminamos estos 7 casos de la muestra. Así, el *data frame* “originales\_so”, que es el que emplearemos para estimar el modelo, cuenta con 40 observaciones.

Una de las variables explicativas es un atributo o factor, llamado DIMENSION, que cuenta con 3 niveles: “GRANDE”, “MEDIANA” y “PEQUEÑA”, según el número de empresas integradas en la matriz a la que pertenece cada empresa de la muestra. Hemos de informar a R de la condición de atributo o factor de esta variable (pues de momento solo la contempla como una variable cualitativa o alfanumérica). Para ello ejecutaremos el código:

# Convertir variable DIMENSION en Factor.  
  
 originales\_so$DIMENSION <- as.factor(originales\_so$DIMENSION)  
 levels(originales\_so$DIMENSION)

## [1] "GRANDE" "MEDIA" "PEQUEÑA"

Una vez preparadas todas las variables, es el momento de **especificar y estimar una regresión múltiple inicial**, que contenga todas las variables explicativas candidatas a formar parte de la versión final.

Para especificar y estimar este modelo inicial, deben de cargarse previamente algunos paquetes que es necesario utilizar: {knitr} y {kableExtra} para construir tablas con los resultados de la regresión, y {broom}. Este último paquete es fundamental, ya que permite disponer de un modo cómodo de todos los elementos de los que se compone un modelo estimado (no solo lineal, como es nuestro caso).

Por otro lado, El MBR lineal, estimado mediante el método de mínimos cuadrados ordinarios (MCO), se obtendrá mediante la función lm(). Los resultados los guardaremos en un objeto de nombre, por ejemplo, “ecua0”. Luego, se realizará un summary() para ver dicha estimación.

# ESPECIFICACION Y ESTIMACION  
  
 # Cargar las librerías necesarias  
  
 library (knitr)  
 library (kableExtra)  
 library (broom)  
 library (car) # para obtener el vif  
  
 # Especificar el modelo de regresión lineal  
 ecua0 <- lm(data = originales\_so,  
 RENECO ~ RES + ACTIVO + ENDEUDA + APALANCA + DIMENSION)  
 summary(ecua0)

El resultado es:

##   
## Call:  
## lm(formula = RENECO ~ RES + ACTIVO + ENDEUDA + APALANCA + DIMENSION,   
## data = originales\_so)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -7.0302 -2.1468 -0.1034 1.5821 6.7527   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 5.045e+00 1.575e+00 3.203 0.00301 \*\*   
## RES 9.962e-04 2.137e-04 4.661 4.99e-05 \*\*\*  
## ACTIVO -3.060e-05 8.938e-06 -3.423 0.00167 \*\*   
## ENDEUDA -6.297e-03 1.859e-02 -0.339 0.73698   
## APALANCA -4.197e-04 3.508e-04 -1.196 0.24007   
## DIMENSIONMEDIA 1.462e+00 1.516e+00 0.964 0.34195   
## DIMENSIONPEQUEÑA 1.781e+00 1.563e+00 1.139 0.26282   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 3.43 on 33 degrees of freedom  
## Multiple R-squared: 0.5943, Adjusted R-squared: 0.5205   
## F-statistic: 8.056 on 6 and 33 DF, p-value: 2.122e-05

Podemos observar, en cuanto a la **bondad del modelo**, cómo es capaz de recoger el 52% de la varianza o comportamiento de la rentabilidad económica (RENECO), atendiendo al valor del coeficiente de determinación lineal corregido (*Adjusted R-squared*). Los coeficientes o parámetros estimados, en su conjunto, son estadísticamente significativos para una significación de 0,05 (p-valor muy pequeño, en el *contraste F de significación conjunta*). En cuanto a los coeficientes estimados considerados individualmente, encontramos que son **estadísticamente significativos** tanto el término independiente (intercept), como los asociados a las variables RES y ACTIVO, a juzgar por los *p-valores* correspondientes al *contraste t de significación individual*. En ambos casos, además, son coeficientes con signo positivo, lo que se interpreta como que ambas variables influyen sobre RENECO de modo que, a mayor valor de estas variables, en general se obtiene una mayor rentabilidad económica.

Por último, es conveniente advertir que el factor DIMENSION se especifica mediante dos variables dicotómicas que representan a dos de los niveles del factor (“MEDIA” y “PEQUEÑA”). sus coeficientes muestran el efecto relativo de ese nivel en relación con el nivel que no es especificado explicitamente (“GRANDE”), ya que no se pueden especificar todos los niveles de un factor para no generar un problema de multicolinealidad perfecta.

Hay otras informaciones importantes a la hora de valorar la especificación (inicial) del modelo que no se recogen en el summary() del mismo. Además, vamos a presentar todo en modo de tablas diseñadas con kable(). Para poder hacer esto no solo con este modelo inicial, sino con cualquier otra estimación, vamos a integrar el código correspondiente en una función de R. Llamaremos a esta función presenta\_modelo(), y recibirá como input la estimación lineal de un modelo, ofreciendo como output tres tablas contenidas en una *lista*: la primera es una *versión* del summary(), la segunda es una tabla con otras informaciones adicionales, como el valor del *Criterio de Información de Akaike* (AIC), y la tercera contiene los valores del *factor de inflación de la varianza* (VIF) de las variables del modelo. Previamente a mostrar el código de la función, fijaremos un parámetro de nombre, por ejemplo, “knitr.table.format”, para recoger el formato en el que se generarán las tablas diseñadas:

#diseña salida ordenador  
  
 knitr.table.format = "html"

El código de la función se basa en el aprovechamiento, a su vez, de dos de las funciones del paquete {broom}. Para realizar la versión en tabla del summary() del modelo, se aplica la función tidy(). Esta función crea un data frame donde se almacenan las columnas con las distintas informaciones (coeficientes, desviaciones típicas, valores del estadístico t, p-valores…) con tantas filas como variables explicativas especificadas. La función glance(), por su lado, extre las siguientes informaciones:

1. **r.squared**: El coeficiente de determinación R², que indica el porcentaje de variación explicada por el modelo.
2. **adj.r.squared**: El R² ajustado, que toma en cuenta los grados de libertad.
3. **sigma**: El error estándar de los residuos.
4. **statistic**: El estadístico F del modelo.
5. **p.value**: El valor p asociado con el estadístico F, que indica la significancia global del modelo.
6. **df**: Los grados de libertad del numerador del estadístico F.
7. **logLik**: El logaritmo de la verosimilitud del modelo.
8. **AIC**: El criterio de información de Akaike.
9. **BIC**: El criterio de información bayesiano.
10. **deviance**: La desviación del modelo.

El código es, en definitiva:

# Definir la función de presentación de resultados: presenta\_modelo() #####  
  
 presenta\_modelo <- function(modelo) {  
  
 # Lista de piezas  
 modelo\_piezas <-list()  
   
 # Aplicar la función tidy() al modelo  
 resultados <- tidy(modelo)  
   
 # Seleccionar las columnas deseadas  
 resultados <- resultados[, c("term",  
 "estimate",  
 "std.error","statistic",  
 "p.value")]  
 # Añadir la columna 'stars' según los valores de 'p.value'  
 resultados$stars <- cut(resultados$p.value,  
 breaks = c(-Inf, 0.001, 0.01, 0.05, 0.1, Inf),  
 labels = c("\*\*\*", "\*\*", "\*", "·", " "),  
 right = FALSE)  
 # formatear los valores de la columna "estimate" a 5 decimales  
 resultados$estimate <- formatC(resultados$estimate,  
 format = "f",  
 digits = 5)  
   
 # Crear la tabla con kable  
 tabla1 <- resultados %>%  
 kable(format = knitr.table.format,  
 caption = "Modelo Lineal",  
 col.names = c("Variable", "Coeficiente", "Desv. Típica",  
 "Estadístico t", "p-valor", "Sig."),  
 digits = 3,  
 align = c("l", "c", "c", "c", "c", "c")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped",  
 "bordered",  
 "condensed",  
 position = "center",  
 font\_size = 11)  
   
 modelo\_piezas[[1]] <- tabla1  
   
 # Aplicar la función glance  
 estadisticos <- glance(modelo)  
 estadisticos <- estadisticos[,c("r.squared",  
 "adj.r.squared",  
 "sigma",  
 "statistic",  
 "p.value",  
 "AIC",  
 "nobs")]  
 # Crear la tabla con kable  
 tabla2 <- estadisticos %>%  
 kable(format = knitr.table.format,  
 caption = "Estadísticos del modelo",  
 col.names = c("R2", "R2 ajustado", "Sigma", "Estadístico F",  
 "p-valor", "AIC", "num. observaciones"),  
 digits = 3,  
 align = "c") %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped",  
 "bordered",  
 "condensed",  
 position = "center",  
 font\_size = 11)  
   
 modelo\_piezas[[2]] <- tabla2   
  
 # Obtener VIF  
 vif\_df <- as.data.frame(vif(modelo))  
  
 # Añadir nombres de filas  
 library(tibble)  
 vif\_df <- vif\_df %>%  
 rownames\_to\_column(var = "Variable")  
   
 # Crear tabla con kable  
 tabla3 <- vif\_df[,1:2] %>%  
 kable(format = knitr.table.format,  
 caption = "Factor de inflación de la varianza",  
 col.names = c("Variable","Valor VIF"),  
 digits = 3,  
 align = "c") %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped",  
 "bordered",  
 "condensed",  
 position = "center",  
 font\_size = 11)  
   
 modelo\_piezas[[3]] <- tabla3  
   
 return(modelo\_piezas)  
 }  
############################################################################

Una vez definida la función, podemos aplicarla al modelo estimado inicial, guardando las tres tablas generadas en la lista “modelo\_0”, que pueden ser visualizadas:

modelo\_0 <- presenta\_modelo(ecua0)  
  
 modelo\_0[[1]]

(#tab:unnamed-chunk-282)(#tab:unnamed-chunk-282)Modelo Lineal

Variable

Coeficiente

Desv. Típica

Estadístico t

p-valor

Sig.

(Intercept)

5.04540

1.575

3.203

0.003

\*\*

RES

0.00100

0.000

4.661

0.000

\*\*\*

ACTIVO

-0.00003

0.000

-3.423

0.002

\*\*

ENDEUDA

-0.00630

0.019

-0.339

0.737

APALANCA

-0.00042

0.000

-1.196

0.240

DIMENSIONMEDIA

1.46168

1.516

0.964

0.342

DIMENSIONPEQUEÑA

1.78055

1.563

1.139

0.263

modelo\_0[[2]]

(#tab:unnamed-chunk-282)(#tab:unnamed-chunk-282)Estadísticos del modelo

R2

R2 ajustado

Sigma

Estadístico F

p-valor

AIC

num. observaciones

0.594

0.521

3.43

8.056

0

220.431

40

modelo\_0[[3]]

(#tab:unnamed-chunk-282)(#tab:unnamed-chunk-282)Factor de inflación de la varianza

Variable

Valor VIF

RES

1.690

ACTIVO

1.873

ENDEUDA

1.279

APALANCA

1.318

DIMENSION

1.558

Los resultados de la primera y segunda tabla ya fueron comentados en casi su totalidad anteriormente, al comentar el summary() del modelo. Se ha añadido el valor del *Criterio de Información de Akaike* (AIC). Esta es una medida basada en la *función de verosimilitud* que permite comparar la adecuación de especificaciones alternativas para representar la realidad, de modo que, **a menor AIC, mejor especificación.**

La tercera tabla muestra los valores del *factor de la inflación de la varianza* (VIF) para cada variable explicativa. El VIF **mide el riesgo de que, debido a la influencia de la variable en cuestión, exista un problema de multicolinealidad** entre las variables explicativas del modelo. Un valor de 5/10 (dependiendo de los autores) sugiere que puede existir un problema de multicolinealidad importante. En el caso del modelo inicial, ningún valor del VIF sugiere un posible problema de multicolinealidad.

La búsqueda de una especificación alternativa que sea más adecuada puede atender a múltiples estrategias del analista, y del propio propósito con el que se quiere utilizar el modelo. Por tanto, implica una gran carga de subjetividad. No obstante, existen métodos automatizados para que, una vez se tiene la estimación inicial, se obtenga una especificación más sencilla (*Principio de Parsimonia*) sin una pérdida grande de bondad de la regresión. Por ejemplo, un método es el *step / backward* que, en función del *Criterio de Información de Akaike (AIC)*, irá probando a estimar especificaciones más simples que disminuyan de AIC (lo que implica una mejor especificación). En nuestro caso, se aplicará con el código:

ecuaDEF <- step(ecua0, scale = 0,  
 direction = c("backward"),  
 trace = 1, steps = 1000, k = 2)

## Start: AIC=104.92  
## RENECO ~ RES + ACTIVO + ENDEUDA + APALANCA + DIMENSION  
##   
## Df Sum of Sq RSS AIC  
## - DIMENSION 2 17.324 405.63 102.66  
## - ENDEUDA 1 1.350 389.65 103.06  
## - APALANCA 1 16.843 405.14 104.61  
## <none> 388.30 104.92  
## - ACTIVO 1 137.886 526.19 115.07  
## - RES 1 255.671 643.97 123.15  
##   
## Step: AIC=102.66  
## RENECO ~ RES + ACTIVO + ENDEUDA + APALANCA  
##   
## Df Sum of Sq RSS AIC  
## - ENDEUDA 1 0.27 405.90 100.69  
## <none> 405.63 102.66  
## - APALANCA 1 27.08 432.71 103.25  
## - ACTIVO 1 278.04 683.67 121.54  
## - RES 1 386.92 792.55 127.45  
##   
## Step: AIC=100.69  
## RENECO ~ RES + ACTIVO + APALANCA  
##   
## Df Sum of Sq RSS AIC  
## <none> 405.90 100.69  
## - APALANCA 1 34.56 440.46 101.96  
## - ACTIVO 1 280.94 686.83 119.73  
## - RES 1 394.36 800.26 125.84

summary (ecuaDEF)

##   
## Call:  
## lm(formula = RENECO ~ RES + ACTIVO + APALANCA, data = originales\_so)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -6.3668 -2.5872 -0.3775 1.4791 7.3184   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 5.759e+00 8.530e-01 6.752 6.97e-08 \*\*\*  
## RES 1.110e-03 1.877e-04 5.914 9.05e-07 \*\*\*  
## ACTIVO -3.644e-05 7.301e-06 -4.992 1.54e-05 \*\*\*  
## APALANCA -5.376e-04 3.070e-04 -1.751 0.0885 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 3.358 on 36 degrees of freedom  
## Multiple R-squared: 0.5759, Adjusted R-squared: 0.5406   
## F-statistic: 16.3 on 3 and 36 DF, p-value: 7.444e-07

El resultado del algoritmo es una especificación del modelo, cuya estimación se almacena en memoria con el nombre, por ejemplo, “ecuaDEF”. Se ha mostrado el summary() del modelo. También se puede aplicar la función presenta\_modelo(), para obtener un *output* de la estimación más detallado:

modelo\_DEF <- presenta\_modelo(ecuaDEF)  
  
 modelo\_DEF[[1]]

(#tab:unnamed-chunk-284)(#tab:unnamed-chunk-284)Modelo Lineal

Variable

Coeficiente

Desv. Típica

Estadístico t

p-valor

Sig.

(Intercept)

5.75901

0.853

6.752

0.000

\*\*\*

RES

0.00111

0.000

5.914

0.000

\*\*\*

ACTIVO

-0.00004

0.000

-4.992

0.000

\*\*\*

APALANCA

-0.00054

0.000

-1.751

0.088

·

modelo\_DEF[[2]]

(#tab:unnamed-chunk-284)(#tab:unnamed-chunk-284)Estadísticos del modelo

R2

R2 ajustado

Sigma

Estadístico F

p-valor

AIC

num. observaciones

0.576

0.541

3.358

16.295

0

216.204

40

modelo\_DEF[[3]]

(#tab:unnamed-chunk-284)(#tab:unnamed-chunk-284)Factor de inflación de la varianza

Variable

Valor VIF

RES

1.360

ACTIVO

1.304

APALANCA

1.054

En el modelo definitivo (ecuaDEF), tan solo permanecen 3 variables explicativas: RES, ACTIVO (ambas significativas para una significación de 0,05) y APALANCA (significativa para una significación de 0,1). Una diferencia importante respecto al modelo inicial es que, el signo del coeficiente asociado a ACTIVO pasa a ser negativo. También el grado de apalancamiento tiene asociado un coeficiente negativo (a mayor apalancamiento, menor rentabilidad económica). SI observamos la segunda tabla, puede observarse que el valor de AIC es de 216,20, inferior al valor de AIC del modelo inicial (220,4). La bondad del ajuste, medida por medio del R2 ajustado, es de 0,541; superior al de modelo inicial (0,521). Finalmente, en la tercera tabla se muestran unos valores de VIF bajos, por lo que se descartan problemas de multicolinealidad.

Una vez decidida la especificación (final) del modelo, es necesario desarrollar la **etapa de contrastación de las hipótesis básicas del modelo**, con la intención de determinar el grado en que los estimadores obtenidos mediante MCO gozan de buenas propiedades, o si es necesario aplicar métodos de estimación alternativos, métodos econométricos específicos, o incluso re-especificar el modelo).

Es conveniente, previamente al análisis de cada hipótesis, generar varios gráficos de gran utilidad.

El primer paso, no obstante, es recurrir a la función augment() del paquete {broom}. Esta función, aplicada a un modelo, genera algunas series de datos fundamentales relacionadas con el mismo, y las almacena junto a las variables del modelo especificado en un data frame. En nuestro caso, hemos llamado al data frame “series\_ecuaDEF”, y hemos cambiado el nombre a las series que vamos a utilizar posteriormente: los valores ajustados de la variable dependiente (RENECO.est), los residuos (residuos), y los valores de la distancia de Cook (cooksd). Además, hemos creado una variable llamada ORDEN para asignar un valor correlativo a cada observación o caso de la muestra:

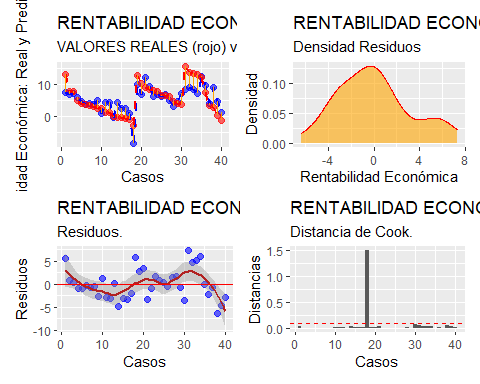
# MODELO FINAL: CONTRASTACIÓN.  
  
 series\_ecuaDEF <- augment(ecuaDEF)  
 series\_ecuaDEF <- series\_ecuaDEF %>%  
 rename(RENECO.est = .fitted,  
 residuos = .resid,  
 cooksd = .cooksd)  
 series\_ecuaDEF$ORDEN = c(1:nrow(series\_ecuaDEF))  
 summary (series\_ecuaDEF)

## .rownames RENECO RES   
## Length:40 Min. :-2.708 Min. :-5268.6   
## Class :character 1st Qu.: 2.103 1st Qu.: 892.2   
## Mode :character Median : 4.404 Median : 2321.2   
## Mean : 5.547 Mean : 2812.2   
## 3rd Qu.: 8.270 3rd Qu.: 3951.2   
## Max. :15.882 Max. :12819.0   
##   
## ACTIVO APALANCA RENECO.est   
## Min. : 25355 Min. :-3037.8 Min. :-8.564   
## 1st Qu.: 33162 1st Qu.: 21.4 1st Qu.: 3.970   
## Median : 45903 Median : 146.5 Median : 5.620   
## Mean : 78664 Mean : 868.9 Mean : 5.547   
## 3rd Qu.: 81312 3rd Qu.: 1080.1 3rd Qu.: 7.432   
## Max. :443467 Max. : 8049.4 Max. :12.464   
##   
## residuos .hat .sigma   
## Min. :-6.3668 Min. :0.02906 Min. :3.107   
## 1st Qu.:-2.5872 1st Qu.:0.03774 1st Qu.:3.350   
## Median :-0.3775 Median :0.05202 Median :3.385   
## Mean : 0.0000 Mean :0.10000 Mean :3.356   
## 3rd Qu.: 1.4791 3rd Qu.:0.10120 3rd Qu.:3.403   
## Max. : 7.3184 Max. :0.53601 Max. :3.405   
##   
## cooksd .std.resid ORDEN   
## Min. :0.0000143 Min. :-1.9446 Min. : 1.00   
## 1st Qu.:0.0007106 1st Qu.:-0.7888 1st Qu.:10.75   
## Median :0.0083866 Median :-0.1147 Median :20.50   
## Mean :0.0514405 Mean : 0.0116 Mean :20.50   
## 3rd Qu.:0.0271051 3rd Qu.: 0.4557 3rd Qu.:30.25   
## Max. :1.4848788 Max. : 2.4566 Max. :40.00

Una vez obtenidas todas las series necesarias, diseñaremos los gráficos que facilitarán la comprensión y contraste del modelo final. El primero compara los valores reales de RENECO con las estimaciones del modelo, RENECO.est. Lógicamente, es deseable que, para cada caso, la distancia entre ambos puntos sea mínima. El segundo muestra los residuos. El tercero es el *gráfico de densidad* de los residuos, y el cuarto representa, para cada caso, el *valor de la distancia de Cook*. Los valores altos de la distancia de Cook en un modelo de regresión indican observaciones que tienen una influencia significativa en los coeficientes estimados del modelo. Son considerados “altos” los valores que superan el valor inverso de 4 por el número de observaciones muestrales.

Los 4 gráficos se agrupan mediante la gramática del paquete {patchwork}. Finalmente, se genera una tabla a partir de un filtro para identificar los casos concretos que tienen valores “altos” de la distancia de Cook. En definitiva, el código es:

# Gráficos.  
   
 g\_real\_pred <- ggplot(data = series\_ecuaDEF) +  
 geom\_point(aes(x = ORDEN, y = RENECO.est),  
 size= 2,  
 alpha= 0.6,  
 color = "blue") +   
 geom\_point(aes(x = ORDEN, y = RENECO),  
 size= 2,  
 alpha= 0.6,  
 color = "red") +  
 geom\_line(aes(x = ORDEN, y = RENECO.est),  
 color = "blue",  
 linetype = "dashed",  
 size= 1) +  
 geom\_line(aes(x = ORDEN, y = RENECO),  
 color = "red",  
 linetype = "dashed",  
 size= 1) +  
 geom\_segment(aes(x = ORDEN, xend = ORDEN, y = RENECO.est, yend = RENECO),  
 color = "orange") +  
 ggtitle("RENTABILIDAD ECONÓMICA.",  
 subtitle= "VALORES REALES (rojo) vs PREDICCIONES (azul).") +  
 xlab("Casos") +   
 ylab("Rentabilidad Económica: Real y Predicción")  
  
 g\_resid <- ggplot(data = series\_ecuaDEF, aes(x = ORDEN, y = residuos)) +  
 geom\_point(size=2, alpha= 0.6, color = "blue") +  
 geom\_smooth(color = "firebrick", span = 0.5) +  
 geom\_hline(yintercept = 0, color = "red")+  
 ggtitle("RENTABILIDAD ECONÓMICA.", subtitle= "Residuos.")+  
 xlab("Casos") +   
 ylab("Residuos")  
  
 g\_hresid <- ggplot(data = series\_ecuaDEF, map = aes(x = residuos)) +  
 geom\_density(colour = "red", fill = "orange", alpha = 0.6) +  
 ggtitle("RENTABILIDAD ECONÓMICA", subtitle = "Densidad Residuos")+  
 xlab("Rentabilidad Económica") +  
 ylab("Densidad")  
  
 g\_cook <- ggplot(data = series\_ecuaDEF, aes(x = ORDEN, y = cooksd)) +  
 geom\_bar(stat = "identity") +  
 geom\_hline(yintercept = 4/nrow(series\_ecuaDEF),  
 linetype = "dashed",  
 color = "red") +  
 ggtitle("RENTABILIDAD ECONÓMICA.", subtitle= "Distancia de Cook.")+  
 xlab("Casos") +   
 ylab("Distancias")  
  
 library (patchwork)  
  
 (g\_real\_pred | g\_hresid) / (g\_resid | g\_cook)  
  
 tablaCook <- series\_ecuaDEF %>%  
 filter ( cooksd > 4/nrow(series\_ecuaDEF)) %>%  
 select (.rownames, cooksd) %>%  
 kable(format = knitr.table.format,  
 caption = "Casos destacados distancia de Cook",  
 col.names = c("Caso","Distancia de Cook"),  
 digits = 3,  
 align = c("l","c")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped",  
 "bordered",  
 "condensed",  
 position = "center",  
 font\_size = 11)  
  
 tablaCook



(#tab:unnamed-chunk-288)(#tab:unnamed-chunk-288)Casos destacados distancia de Cook

Caso

Distancia de Cook

Innogy Spain SA.

1.485

De los gráficos anteriores se desprende, en general, que los residuos parecen mantener un comportamiento conforme a una distribución normal, y que hay una observación o caso con una distancia de Cook que puede influir de modo relevante en el valor de los coeficientes estimados. Esta empresa es identificada como “Innogy Spain S.A.” Podría estudiarse en detalle este caso o incluso reestimar el modelo sin su presencia, a fin de comprobar el efecto que tiene esta observación sobre la estimación en general.

Tras estudiar los gráficos, vamos a pasar a contrastar las hipótesis básicas del MBR referidas a la *forma funcional*, y la *normalidad* y *comportamiento homoscedástico* de la perturbación aleatoria. Para ello se aplicarán las pruebas *Ramsey-Reset*, *Shapiro-Wilk* y *Breusch-Pagan*, respectivamente. Los resultados de las tres pruebas se presentarán condensados en una tabla, en la que además se incluirá la conclusión del contraste, para una significación de 0,05. Para crear la tabla, primero se generará un data frame con sus elementos, denominado, por ejemplo, “check\_hipotesis”. El código es:

# Hipótesis básicas MBR.  
  
 library (lmtest)  
 reset\_test <- resettest(ecuaDEF, data= originales\_so) # F. Funcional  
 shapiro\_test <- shapiro.test(series\_ecuaDEF$residuos) # Normalidad  
 bp\_test <- bptest(ecuaDEF) # Homoscedasticidad  
  
 # Tabla resultados  
  
 # Crear un data frame con los resultados  
  
 check\_hipotesis <- data.frame(  
 "Tipo\_de\_prueba" = c("Forma Funcional",  
 "Normalidad de perturbación aleatoria",  
 "Homoscedasticidad de perturbación aleatoria"),  
 "Prueba" = c("Ramsey-Reset",  
 "Shapiro-Wilk",  
 "Breusch-Pagan"),  
 "Estadistico" = c(reset\_test$statistic,  
 shapiro\_test$statistic,  
 bp\_test$statistic),  
 "P\_valor" = c(reset\_test$p.value,  
 shapiro\_test$p.value,  
 bp\_test$p.value),  
 "Conclusion" = c(ifelse(reset\_test$p.value >= 0.05,  
 "F. funcional correcta",  
 "F. funcional incorrecta"),  
 ifelse(shapiro\_test$p.value >= 0.05,  
 "Normalidad",  
 "No-Normalidad"),  
 ifelse(bp\_test$p.value >= 0.05,  
 "Homoscedasticidad",  
 "Heteroscedasticidad")))  
  
 row.names(check\_hipotesis) <- NULL  
  
 # Crear la tabla con kable  
   
 tabla\_check <- check\_hipotesis %>%  
 kable(format = knitr.table.format,  
 caption = "Contrastes de hipótesis del MBR",  
 col.names = c("Tipo de prueba",  
 "Prueba",  
 "Estadístico",  
 "P-valor",  
 "Conclusión"),  
 digits = 3,  
 align = c("l", "c", "c", "c", "c")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped",  
 "bordered",  
 "condensed",  
 position = "center",  
 font\_size = 11)  
  
 tabla\_check

(#tab:unnamed-chunk-290)(#tab:unnamed-chunk-290)Contrastes de hipótesis del MBR

Tipo de prueba

Prueba

Estadístico

P-valor

Conclusión

Forma Funcional

Ramsey-Reset

6.907

0.003

F. funcional incorrecta

Normalidad de perturbación aleatoria

Shapiro-Wilk

0.963

0.212

Normalidad

Homoscedasticidad de perturbación aleatoria

Breusch-Pagan

5.773

0.123

Homoscedasticidad

En la tabla de resultados del contraste de las hipótesis básicas del MBR, encontramos que el modelo no plantea problemas de falta de normalidad o heteroscedasticidad en la perturbación aleatoria. En cambio, la prueba de *Ramsey-Reset* rechaza la hipótesis nula de especificación lineal correcta. En concreto, este contraste plantea estas dos hipótesis:

* **Hipótesis nula (H0)**: El modelo está correctamente especificado, es decir, no hay errores de especificación. En términos más técnicos, esto significa que las combinaciones no lineales de los valores ajustados no tienen poder explicativo adicional sobre la variable dependiente.
* **Hipótesis alternativa (H1)**: El modelo está mal especificado, lo que implica que las combinaciones no lineales de los valores ajustados sí tienen poder explicativo adicional sobre la variable dependiente.

Si se asume que las variables especificadas son las correctas, y no hay omisión de variables relevantes; un rechazo de la hipótesis nula implica que la relación funcional planteada (lineal), no es correcta.

El incumplimiento de la hipótesis de linealidad podría acarrear que los estimadores MCO obtenidos adolecen de la pérdida de la propiedad de insesgadez.

Si el modelo supera razonablemente la fase de contraste de las hipótesis básicas; podría ser utilizado para los objetivos planteados en la investigación: análisis estructural, previsión, simulación. En este ejemplo, vamos a realizar un ejercicio de simulación.

Vamos a obtener la respuesta de la rentabilidad económica (RENECO), ante 3 escenarios alternativos. Dichos escenarios están guardados en la hoja “Simula” del archivo de Microsoft® Excel® “[eolica\_escenarios.xlsx](https://docs.google.com/spreadsheets/d/1QUqpOHC_XK38MBggb3hYibDReHVfd7wL/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true)”. Hay que tener en cuenta que en los escenarios se aportan valores para algunas variables que no aparecen en el modelo final. Lógicamente, lo importante son los datos del escenario correspondientes a las variables que sí están especificadas. Los escenarios se importan y se almacenan en el *data frame* de nombre, por ejemplo, “escenario”:

# SIMULACIÓN  
  
 # Cargar escenario de Excel  
  
 escenario <- read\_excel("eolica\_escenarios.xlsx", sheet = "Simula")  
 escenario <- data.frame(escenario, row.names = 1)  
 escenario

## RES ACTIVO ENDEUDA APALANCA DIMENSION  
## ESCENARIO A 2500 50000 20.25 0.238 GRANDE  
## ESCENARIO B 2500 25000 50.00 50.000 MEDIA  
## ESCENARIO C 3000 5000 90.00 100.000 PEQUEÑA

Mediante la función predict(), se generará la respuesta a los escenarios propuestos, por parte del modelo definitivo (ECUADEF). Esta respuesta se guardará en el *data frame* de nombre, por ejemplo, “estimación”, que luego uniremos al *data frame* “escenario” mediante la función cbind(), creando un único *data frame* llamado, por ejemplo, “simulacion”:

# Simulación con el modelo  
  
 estimacion <-predict (object= ecuaDEF,  
 newdata = escenario,  
 interval="prediction",  
 level=0.95)  
 estimacion

## fit lwr upr  
## ESCENARIO A 6.711704 -0.2161380 13.63955  
## ESCENARIO B 7.596000 0.6413923 14.55061  
## ESCENARIO C 8.852940 1.8523982 15.85348

simulacion <- cbind(escenario, estimacion)

Finalmente, vamos a presentar la simulación en forma de tabla. En primer lugar, vamos a dar un formato específico a cada variable, cara a su volcado a la tabla, con la función format(). El argumento nsmall= indica el número mínimo de decimales que habrá a la derecha del punto decimal:

# Formatear las columnas con el número mínimo de decimales deseado  
  
 simulacion$ENDEUDA <- format(simulacion$ENDEUDA, nsmall = 3)  
 simulacion$APALANCA <- format(simulacion$APALANCA, nsmall = 3)  
 simulacion$fit <- format(simulacion$fit, nsmall = 3)  
 simulacion$lwr <- format(simulacion$lwr, nsmall = 3)  
 simulacion$upr <- format(simulacion$upr, nsmall = 3)

Tras definir el formato numérico de las variables, se construirá la tabla, de nombre, por ejemplo, “tablasimula”:

# Tabla  
  
 tablasimula <- simulacion %>%  
 kable(format = knitr.table.format,  
 caption = "Simulación Modelo Rentabilidad Económica",  
 col.names = c("Escenario",  
 "Resultado",  
 "Activo",  
 "Endeuda",  
 "Apalancamiento",  
 "Dimensión",  
 "Previsión",  
 "Inferior 95%",  
 "Superior 95%"),  
 digits = 3) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped",  
 "bordered",  
 "condensed",  
 position = "center",  
 font\_size = 11) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 row\_spec(1:(nrow(simulacion)), bold= F, align = "c")  
  
 tablasimula

(#tab:unnamed-chunk-295)(#tab:unnamed-chunk-295)Simulación Modelo Rentabilidad Económica

Escenario

Resultado

Activo

Endeuda

Apalancamiento

Dimensión

Previsión

Inferior 95%

Superior 95%

ESCENARIO A

2500

50000

20.250

0.238

GRANDE

6.711704

-0.2161380

13.63955

ESCENARIO B

2500

25000

50.000

50.000

MEDIA

7.596000

0.6413923

14.55061

ESCENARIO C

3000

5000

90.000

100.000

PEQUEÑA

8.852940

1.8523982

15.85348

## 9.3 Materiales para realizar las prácticas del capítulo.

En esta sección se muestran los links de acceso a los diferentes materiales (*scripts*, datos…) necesarios para llevar a cabo los contenidos prácticos del capítulo.

**Datos (en formato Microsoft (R) Excel (R)):**

* eolica\_50.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1oq17lI4trI8k5oeDgj_fYg9WhbRIqDLG/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true))
* eolica\_escenarios.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1QUqpOHC_XK38MBggb3hYibDReHVfd7wL/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true))
* **Scripts:**
* regresion\_eolica.R [(](https://drive.google.com/file/d/1p4p17OudRprwkpF7wvvF1gwHZG82uv89/view?usp=sharing)[obtener aquí](https://drive.google.com/file/d/1ilNEAA3-Co9kbs5Gv4arwCBCcCnbn0zY/view?usp=sharing))

# 10 Análisis de Datos Cualitativos.

## 10.1 Introducción.

En numerosas ocasiones la información con que el analista debe enfrentarse es de naturaleza **cualitativa**, esto es, la información se recoge en características no numéricas o **atributos** (o factores o variables categóricas o cualitativas). Cuando ocurre esto, es necesario recurrir a técnicas de explotación específicas para este tipo de datos. Así, en este capítulo vamos a explorar algunos métodos para extraer información útil cuando los datos de **los que disponemos son categóricos.**

En estos casos, la información suele sintetizarse y presentarse mediante las denominadas “**tablas de contingencia**”. En este tipo de tablas, se muestran las frecuencias conjuntas, es decir, el número de casos que comparten los distintos niveles o categorías de los diferentes factores.

Cuando se trabaja con varios factores o atributos, representados en su correspondiente tabla de contingencia, uno de los análisis más interesantes es determinar si existe **asociación** entre los factores o atributos. Esto es, si se aprecia algún tipo de relación estadística entre estas variables categóricas o cualitativas, en el sentido de si se puede afirmar que el hecho de que los casos de la muestra tomen ciertos niveles o categorías en unos factores, hace que estos mismos casos tiendan a tomar ciertos niveles o categorías de otro u otros factores.

## 10.2 Tablas de contingencia, y asociación entre dos atributos o factores.

Comenzaremos con el caso más sencillo, en el que los casos de nuestra muestra se distribuyen entre las categorías o niveles de dos atributos o factores. En este caso, la tabla de contingencia que presente los datos de esta situación será una simple tabla de doble entrada. En las filas de la tabla se dispondrán las categorías de uno de los factores, y en las columnas de la tabla se situarán las categorías del otro factor.

Para ilustrar el tratamiento de una tabla de contingencia bidimensional, vamos a trabajar con los datos correspondientes a 51 empresas de generación eléctrica presentes en el archivo de Microsoft® Excel® “[eolica\_contingencia.xlsx](https://docs.google.com/spreadsheets/d/1N5QIRKbm6OKAOaN8DB2YFDptpZ_0pbPc/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true)”. Este archivo cuenta con 3 hojas, los datos en cuestión se encuentran en la que tiene por nombre “Datos”.

Dentro de “Datos”, existen dos variables categóricas, atributos o factores: la variable DIMENSION, que tiene, a su vez, tres categorías dependiendo del número de empresas que integradas en la matriz de la empresa en cuestión (“GRANDE”, “MEDIA” o “REDUCIDA”); y la variable VALORACION, que cuenta con tres categorías dependiendo de la opnión de un panel de expertos en cuanto a la situación económica de la empresa (“OPTIMA”, “NORMAL”, “PESIMA”). Nuestra intención es analizar si existe **evidencia sobre asociación** entre las dos variables categóricas, en el sentido de que los casos tienden a concentrarse con cierta facilidad en ciertas combinaciones de categorías de uno y otro factor, y/o a no posicionarse en otras combinaciones; o si por el contrario no se puede afirmar que exista asociación de este tipo.

El código de R que iremos aplicando se encuentra en el *script* “[correspondencias\_eolica.R](https://drive.google.com/file/d/1u8gJa3jQH2cwuQ8KsISQ9esHKrFAO_au/view?usp=sharing)”. Trabajaremos en un proyecto específico de RStudio, si así lo valoramos como conveniente (en este ejemplo, el proyecto “correspondencias”).

Al abrir el *script* en el editor de RStudio, lo primero que veremos será la instrucción para limpiar la memoria de objetos. Tras ello, se procede a la importación de los datos del fichero de *Excel*. Tras importar las variables, se procede a ajustar la importación para que la primera columna pase a ser el nombre de los casos (filas) del *data frame* que recibe y almacena los datos. Este *data frame* se denomina, por ejmplo, “eolicas”. Con un summary() comprobamos que las variables se han importado correctamente:

# Analisis de correspondencias simple de eolicas  
# Disculpen por la falta de tildes!  
  
rm(list = ls())  
  
# DATOS  
  
 # Importando datos  
  
 library (readxl)  
 eolicas <- read\_excel("eolica\_contingencia.xlsx", sheet ="Datos")  
 eolicas <- data.frame(eolicas, row.names = 1)  
 summary (eolicas)

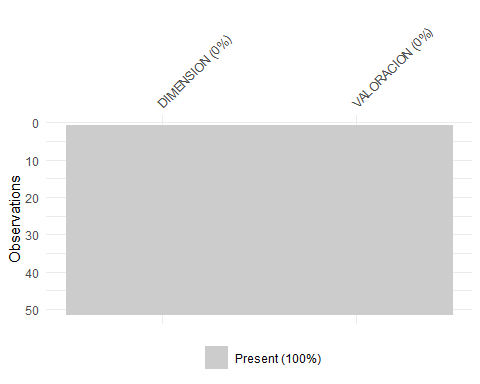
## MARGEN SOLVENCIA COM   
## Min. :-1159.297 Min. :-66.95 Length:51   
## 1st Qu.: 5.677 1st Qu.: 16.32 Class :character   
## Median : 34.252 Median : 42.98 Mode :character   
## Mean : 47.643 Mean : 45.76   
## 3rd Qu.: 53.388 3rd Qu.: 80.31   
## Max. : 1790.123 Max. :100.00   
## NA's :9   
##   
## FJUR ING NCOMP   
## Length:51 Min. : 0.29 Min. : 0   
## Class :character 1st Qu.: 97.59 1st Qu.: 1   
## Mode :character Median : 1241.39 Median : 31   
## Mean : 10705.06 Mean : 1655   
## 3rd Qu.: 6766.47 3rd Qu.: 170   
## Max. :255852.00 Max. :72434   
## NA's :9   
##   
## RES ACTIVO FPIOS   
## Min. :-3274.32 Min. : 3.0 Min. : -1919.42   
## 1st Qu.: 0.30 1st Qu.: 160.5 1st Qu.: 71.57   
## Median : 57.76 Median : 2420.5 Median : 888.78   
## Mean : 2203.99 Mean : 67897.9 Mean : 8991.25   
## 3rd Qu.: 622.08 3rd Qu.: 16379.6 3rd Qu.: 6413.52   
## Max. :78290.00 Max. :2429299.0 Max. :148251.00   
## NA's :5   
##   
## RENECO RENFIN LIQUIDEZ   
## Min. :-103.156 Min. :-596.79 Min. : 0.006   
## 1st Qu.: 0.000 1st Qu.: 0.00 1st Qu.: 0.667   
## Median : 3.165 Median : 12.94 Median : 1.833   
## Mean : 6.559 Mean : 10.94 Mean : 17.853   
## 3rd Qu.: 13.329 3rd Qu.: 37.99 3rd Qu.: 6.010   
## Max. : 95.013 Max. : 144.04 Max. :541.752   
## NA's :2   
##   
## APALANCA AUTOFIN DIMENSION   
## Min. :-312.62 Min. :-0.401 Length:51   
## 1st Qu.: 0.00 1st Qu.: 0.243 Class :character   
## Median : 18.73 Median : 0.508 Mode :character   
## Mean : 333.41 Mean : 3.468   
## 3rd Qu.: 266.68 3rd Qu.: 4.093   
## Max. :6197.54 Max. :42.156   
## NA's :20   
##   
## AUTOFINA VALORACION   
## Length:51 Length:51   
## Class :character Class :character   
## Mode :character Mode :character

A continuación, creamos otro *data frame*, de nombre, por ejemplo, “originales”, con los dos atributos analizados: el factor DIMENSION y el factor VALORACION. Se comprueba si existen observaciones con *missing values* y, en tal caso, se eliminan dichas observaciones aplicando el filtro apropiado, a partir de la función filter() del paquete {dplyr}:

# Seleccionando factores/atributos para el analisis  
  
 library(dplyr)  
 originales<-select(eolicas, DIMENSION, VALORACION)  
 summary (originales)

## DIMENSION VALORACION   
## Length:51 Length:51   
## Class :character Class :character   
## Mode :character Mode :character

# Identificando missing values.  
  
 library(visdat)  
 vis\_miss(originales)



originales %>% filter(is.na(DIMENSION) | is.na(VALORACION)) %>%  
 select(DIMENSION, VALORACION)

## [1] DIMENSION VALORACION  
## <0 rows> (o 0- extensión row.names)

originales <- originales %>%  
 filter(! is.na(DIMENSION) & ! is.na(VALORACION))

En el ejemplo, se comprueba que no existen casos con *missing values*, por lo que la tabla queda en blanco y no se descarta ninguna observación.

La siguiente etapa es la construcción de la “**tabla de contingencia**”. La función fundamental es table() que, como sabemos, hace un recuento de los casos (frecuencias) en los que una variable toma un determinado valor (o en los que un atributo adopta una determinada categoría o nivel). Cuando esta función se aplica a más de una variable o atributo, hace un recuento de los casos que adoptan las posibles combinacionesse entre valores/categorías o niveles de las variables o atributos implicados. Cuando se trata de dos atributos, por tanto, table() construye una tabla de contingencia mediante la formulación de una tabla de doble entrada. En nuestro ejemplo, a la tabla de contingencia la hemos denominado, por ejemplo, “tab.originales”. Luego, la hemos presentado con un formato de tabla elaborado con la función kable() del paquete {knitr}, y algunas funciones adicionales del paquete {kableExtra}:

# TABLA DE CONTINGENCIA  
  
 tab.originales <- table(originales)  
  
 library(knitr)  
 library(kableExtra)  
 knitr.table.format = "html"  
  
 addmargins(tab.originales) %>%  
 kable(format = knitr.table.format,  
 caption="Empresas eólicas") %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped", "bordered", "condensed",  
 position = "center",  
 font\_size = 12) %>%  
 add\_header\_above(c(DIMENSION = 1, VALORACION = 3, " " = 1),  
 bold=T,  
 line=T) %>%  
 row\_spec(0, bold= T, align = "c") %>%  
 column\_spec(1, bold = T)

(#tab:unnamed-chunk-300)(#tab:unnamed-chunk-300)Empresas eólicas

NORMAL

OPTIMA

PESIMA

Sum

GRANDE

9

6

4

19

MEDIA

0

6

4

10

REDUCIDA

4

15

3

22

Sum

13

27

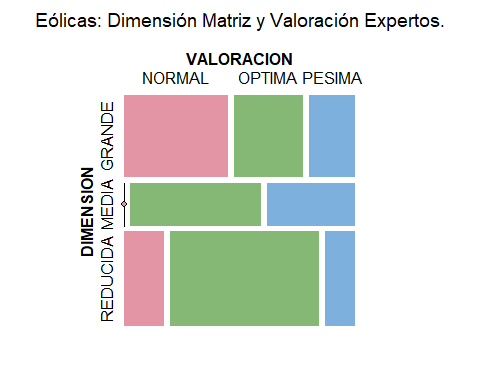
11

51

Es de destacar que, en el código de la tabla, se ha añadido la función addmargin() para que se añadan las frecuencias marginales de las categorias de ambos atributos (sumas de filas y columnas). Además, se ha utilizado la función add\_header\_above() del paquete {kableExtra} para añadir filas superiores al encabezado, que ocupen distintas cantidades de columnas.

La tabla también se puede representar gráficamente mediante la función mosaic() de la librería {vcd}, con lo que se percibirá mejor la magnitud de las frecuencias conjuntas (celdas de la tabla). A mayor frecuencia, mayor área del rectángulo correspondiente:

library (vcd)  
 mosaic(tab.originales,  
 main = "Eólicas: Dimensión Matriz y Valoración Expertos.",  
 shade = T,  
 gp = shading\_Marimekko(tab.originales),  
 main\_gp = gpar(fontsize = 14),  
 sub\_gp = gpar(fontsize = 12))



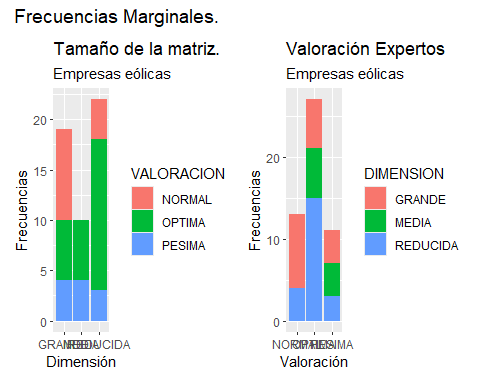
Los argumentos de la función mosaic() juegan el siguiente papel:

1. **tab.originales**: Es la tabla de contingencia que contiene los datos a visualizar en el gráfico mosaico. Debe ser una tabla de contingencia creada con la función table() o una matriz con datos categóricos.
2. **main**: Título principal del gráfico.
3. **shade**: Si se establece en TRUE, aplica un coloreado a las celdas del mosaico; si es FALSE todas los rectángulos serán del mismo color.
4. **gp**: Parámetros gráficos para personalizar la apariencia del gráfico. En este caso, se usa shading\_Marimekko() para aplicar un sombreado específico. Hay otras funciones de parámetros gráficos como shading\_hcl(), shading\_max, o cualquier función personalizada que devuelva un objeto de clase gpar.
5. **main\_gp**: Parámetros gráficos para el título principal, como el tamaño de la fuente. Se pueden especificar atributos como, por medio de la función gpar(), como fontsize, fontfamily, col, etc.
6. **sub\_gp**: Igual que el caso anterior; pero para el subtítulo del gráfico de mosaico.

En nuestro ejemplo, podemos apreciar con claridad como una importante proporción de los casos (empresas) se concentran en la combinación de valoración óptima y dimensión de la empresa matriz reducida. También es destacable la combinación de valoración normal y dimensión de la compañía matriz grande. Por el lado opuesto, destaca la combinación de valoración normal y dimensión de la matriz media, que no posee ningún caso (frecuencia 0), y valoración pésima y dimesión de la matriz reducida.

Otro modo de visualizar la estructura de la tabla es hacer gráficos de barras que muestren las frecuencias de las categorías de cada factor (frecuencias marginales), aunque en cada barra se pueda diferenciar, además, los casos o frecuencias que pertenecen a las categorías del otro factor. En nuestro caso:

# Representando frecuencias de categorias en factores  
  
 library (ggplot2)  
 library (patchwork)  
  
 g1 <- ggplot(originales, mapping= aes(x= DIMENSION, fill = VALORACION)) +  
 geom\_bar() +  
 ggtitle("Tamaño de la matriz.", subtitle = "Empresas eólicas") +   
 ylab("Frecuencias") +  
 xlab("Dimensión")  
  
 g2 <- ggplot(originales, mapping= aes(x= VALORACION, fill = DIMENSION)) +  
 geom\_bar() +  
 ggtitle("Valoración Expertos", subtitle = "Empresas eólicas") +   
 ylab("Frecuencias") +  
 xlab("Valoración")  
  
 (g1 + g2) + plot\_annotation(title = "Frecuencias Marginales.",  
 theme = theme(plot.title = element\_text(size = 14)))



Se observa cómo, en cuanto al atributo o factor DIMENSION, la categoría más frecuente es la dimensión “REDUCIDA”, mientras que la categoría que menos se da en la muestra es, destacadamente, “MEDIA”. Además, como ya se ha comentado, destaca que la mayor parte de los casos de la categoría “REDUCIDA” tienen una valoración de “OPTIMA” en el factor VALORACION. También llama la atención que en la categoría “MEDIA” no existen casos con valoración “NORMAL”. En cuanto al gráfico del atributo o factor “VALORACION”, el mayor número de frecuencias, de modo muy destacado, se concentran en la categoría “OPTIMA”, estando las otras dos categorías bastante igualadas en cuanto al número de casos. Es reseñable también que, en la categoría “OPTIMA”, la mayor parte de casos tienen una dimensión “REDUCIDA”.

Como ya hemos dicho, una de las cuestiones más importantes en el análisis de tablas de contingencia es **determinar si existe asociación entre ambos atributos o factores** (valoración de las empresas y dimensión de las compañías matrices), en el sentido de poder plantear que los casos (empresas) concentradas en ciertas categorías concretas de uno de los factores o atributos tienden a concentrarse, simultáneamente, en ciertas categorías concretas del otro factor o atributo; o al revés: que el hecho de que los casos no tiendan a concentrarse en ciertas categorías de uno de los factores o atributos está relacionado con que no se concentren en ciertas categorías del otro factor o atributo. En nuestro ejemplo, ya hemos señalado algunas combinaciones de categorías que podrían llevar a pensar a que existe cierto grado de asociación entre los atributos DIMENSION y VALORACION.

Existen diferentes pruebas para verificar la posible existencia de asociación entre dos factores. Una de ellas es el ***contraste o prueba de asociación de Pearson****.* Este contraste se base en un estadístico del contraste, que bajo la hipótesis nula de que no existe asociación sigue una distribución Chi/Ji Cuadrado. El estadístico, en realidad, es una medida global de **lo distante que está la tabla de contingencia observada (sus frecuencias conjuntas) respecto a la estructura “ideal” que tendría que tener si existiera independencia** “total” entre ambos atributos. Así, el estadístico del contraste de la *prueba de asociación de Pearson* es:

donde:

* ( ) es la frecuencia observada en la celda ( (i, j) ).
* ( ) es la frecuencia esperada en la celda ( (i, j) ), calculada como ( ).
* ( ) es el total de la fila ( i ).
* ( ) es el total de la columna ( j ).
* ( ) es el total general de todas las observaciones.

Los *residuos estandarizados* son una medida de la desviación de las frecuencias observadas respecto a las frecuencias teóricas o esperadas (en caso de independencia perfecta entre atributos) en una tabla de contingencia. Se utilizan para identificar celdas que contribuyen significativamente a la asociación entre las variables. La fórmula para calcular los residuos estandarizados es:

donde:

* ( ) es el residuo estandarizado para la celda en la fila ( i ) y la columna ( j ).

El paquete {stats} de R (que se carga automáticamente, por defecto, al iniciar R, por lo que no hay que “activarlo”), contiene la función chisq.test() que efectúa la prueba de asociación de *Pearson*. El código y resultado de la prueba es:

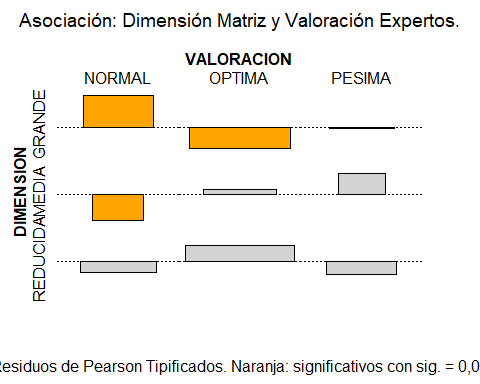
# INDEPENDENCIA / ASOCIACION  
   
 # Test de asociación de Pearson (Ji-Cuadrado)  
   
 Prueba\_asoc\_Pearson <- chisq.test(tab.originales)  
 Prueba\_asoc\_Pearson

##   
## Pearson's Chi-squared test  
##   
## data: tab.originales  
## X-squared = 11.494, df = 4, p-value = 0.02154

En nuestro caso, el *p-valor* es menor que 0,05, luego **se rechaza la hipótesis nula de independencia** de los atributos o factores, y admitimos que **existe asociación entre ambos.**

Precisamente, el paquete {vcd} ofrece la posibilidad de, para cada frecuencia conjunta, visualizar la diferencia estandarizada entre el valor observado y el que debería darse en el caso de que existiera independencia perfecta entre los dos atributos o factores (*residuo de Pearson*). Para ello, se aplica la función assoc(), destinada a construir un gráfico con los residuos de *Pearson* de la tabla de contingencia. Los residuos, si son estadísticamente significativos para una significación de 0,05, se colorean de naranja, y en caso contrario de gris. Para determinar los colores concretos, se ha creado la función custom\_shading(), que se pasa en el argumento gp= para personalizar el color concreto de cada barra del gráfico:

Residuos\_std <- Prueba\_asoc\_Pearson$stdres  
  
 # Definir una función de sombreado personalizada  
 custom\_shading <- function(residuals, cutoff = 1.96) {  
 # Crear una matriz de colores basada en los residuos estandarizados  
 colors <- ifelse(abs(residuals) > cutoff, "orange", "lightgray")  
 return(colors)  
 }  
   
 # Aplicar la función de sombreado en el gráfico mosaico  
   
 assoc(tab.originales,  
 main = "Asociación: Dimensión Matriz y Valoración Expertos.",  
 sub = "Residuos de Pearson Tipificados. Naranja: significativos con sig. = 0,05",  
 compress = FALSE,  
 gp = gpar(fill = custom\_shading(Residuos\_std)),  
 main\_gp = gpar(fontsize = 14),  
 sub\_gp = gpar(fontsize = 12))



Puede observarse cómo en las combinaciones GRANDE/NORMAL, GRANDE/OPTIMA y MEDIA/NORMAL los residuos de Pearson son estadísticamente significativos (lo que, a su vez, indica que las frecuencias correspondientes están muy alejadas de las que debería haber en caso de independencia perfecta), lo que llevaría que la prueba haya rechazado la hipótesis de independencia entre los atributos o factores.

## 10.3 Análisis de correspondencias.

El *análisis de correspondencias* es una técnica destinada a **representar visualmente una tabla de contingencia, en un gráfico bidimensional**. Puede utilizarse para representar una tabla de dos atributos o factores (análisis de correspondencias simple) o de más de dos (análisis de correspondencias múltiple). Cuando en el gráfico bidimensional se representan más de dos atributos o factores, este análisis de convierte, además, en una **técnica de reducción de la dimensión de la información.** De hecho, se puede afirmar que el análisis de correspondencias es una suerte de análisis de **componentes principales aplicado a variables categóricas**.

De acuerdo a lo anterior, se pueden establecer los siguientes paralelismos:

* Las componentes en el análisis de componentes principales equivalen a los **ejes o dimensiones** del análisis de correspondencias.
* La varianza total o *comunalidad* de las variables originales métricas del análisis de componentes principales pasa a ser, en el análisis de correspondencias, la **inercia total**.
* La varianza que es capaz de asumir cada componente en el análisis de componentes principales ahora se denomina **inercia principal** (del eje o dimensión en cuestión).
* Las puntuaciones de las componentes para un caso son, ahora, las **coordenadas** de una categoría de uno de los atributos o factores.
* El papel de las cargas de las variables en las componentes principales lo asumen las **contribuciones** de las categorías o niveles de los factores o atributos en los ejes o dimensiones.

Como principal resultado del análisis se obtendrá un gráfico de dos ejes (dimensiones) en el cuál **se situarán más próximas las categorías de uno y otro factor que mantengan entre sí cierta tendencia a asociarse**.

Existen varios paquetes de R que permiten desarrollar un análisis de correspondencias simple (de dos atributos o factores), como es nuestro ejemplo. Hemos optado, en el ejemplo, por utilizar el paquete {FactoMineR}. Este paquete dispone de la función CA(), que es la encargada de realizar los cálculos del análisis.

En el siguiente código, el análisis de correspondencias se aplica a la tabla de contingencia “tab.originales”, almacenándose la solución en la lista “aceolicas”. Luego, se crea la lista “SolucionCA” para almacenar las tres tablas que recogerán los resultados. Después se crea un *data frame*, “EigenCA”, con el elemento “eig” de la solución del análisis. Ese elemento es un *data frame* de dos columnas (dimensiones o ejes del análisis), para cada una de las cuales se reúnen tres informaciones: la inercia principal o varianza recogida por la dimensión o eje, el porcentaje que supone respecto a la inercia total puesta en juego, y el porcentaje acumulado. El *data frame* se pasa a formato “tabla” de kable() con el nombre “TableEigenCA”, que se guarda en la lista “SolucionCA” como su primer elemento.

Después se generan dos tablas con los mismos elementos, para cada uno de los atributos o factores del análisis. La primera corresponde al atributo DIMENSION. Esta tabla cuenta con una fila por cada una de las categorías de DIMENSION, esto es, “GRANDE”, “MEDIA” y “REDUCIDA”. La segunda corresponde al atributo VALORACION, y cuenta con una fila para sus categorías, “OPTIMA”, “NORMAL”, “PESIMA”. En ambas tablas, las columnas son las siguientes:

1. **Inercia**: La inercia es una medida de la varianza explicada por cada fila o columna (categoría de alguno de los atributos) en el análisis de correspondencias. Valores más altos indican que la fila o columna contribuye más a la varianza total del análisis (Inercia Total).
2. **Coordenadas Dim. 1 y Dim. 2**: Coordenadas de las filas o columnas en las dos dimensiones del espacio de correspondencias. Indican la posición de cada fila o columna en el espacio bidimensional.
3. **cos2 Dim. 1 y Dim. 2 (Coseno Cuadrado)**: El coseno cuadrado de los ángulos entre las filas o columnas y las dimensiones. Mide la calidad de la representación de las filas o columnas en las dimensiones. Valores cercanos a 1 indican que la fila o columna está bien representada en esa dimensión. Valores bajos indican una mala representación.

# ANALISIS DE CORRESPONDENCIAS SIMPLE  
  
 library (FactoMineR)  
 aceolicas <- CA(X = tab.originales, graph = F)  
  
 SolucionCA <- list()  
   
 EigenCA <- as.data.frame(t(aceolicas$eig))  
 EigenCA$elemento <- c("Inercia Total", "% Inercia Principal", "Acumulada")  
 EigenCA <- data.frame(EigenCA, row.names = 3)  
   
   
 TableEigenCA <- EigenCA %>%  
 kable(format = knitr.table.format,  
 caption="Análisis de correspondencias: Dimensión Matrix vs Valoración.",  
 col.names = c("Dimensión/Eje 1", "Dimensión/Eje 2"),  
 digits = 3,  
 align= c("c", "c")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped", "bordered", "condensed",  
 position = "center",  
 font\_size = 12) %>%  
 row\_spec(0, bold= T, align = "c")  
   
 SolucionCA[[1]] <-TableEigenCA  
   
 # Extraer la información de las filas  
 rows\_data <- aceolicas$row$coord  
 rows\_cos2 <- aceolicas$row$cos2  
  
 # Crear un DataFrame para las filas  
 rows\_df <- data.frame(  
 Iner\_1000 = aceolicas$row$inertia,  
 Dim\_1 = rows\_data[, 1],  
 cos2\_1 = rows\_cos2[, 1],  
 Dim\_2 = rows\_data[, 2],  
 cos2\_2 = rows\_cos2[, 2]  
 )  
 rownames(rows\_df) <- rownames(rows\_data)  
   
 TableRows <- rows\_df %>%  
 kable(format = knitr.table.format,  
 caption= (paste0("Análisis de correspondencias: ", colnames(originales)[1], ".")),  
 col.names = c("Inercia", "Coordenadas Dim. 1", "Cos2 Dim. 1",  
 "Coordenadas Dim. 2", "Cos2 Dim. 2"),  
 digits = 3,  
 align= c("c", "c", "c", "c", "c")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped", "bordered", "condensed",  
 position = "center",  
 font\_size = 12) %>%  
 row\_spec(0, bold= T, align = "c")  
   
 SolucionCA[[2]] <-TableRows  
  
 # Extraer la información de las columnas  
 columns\_data <- aceolicas$col$coord  
 columns\_cos2 <- aceolicas$col$cos2  
  
 # Crear un DataFrame para las columnas  
 columns\_df <- data.frame(  
 Inercia = aceolicas$col$inertia,  
 Dim\_1 = columns\_data[, 1],  
 cos2\_1 = columns\_cos2[, 1],  
 Dim\_2 = columns\_data[, 2],  
 cos2\_2 = columns\_cos2[, 2]  
 )  
 rownames(columns\_df) <- rownames(columns\_data)  
  
 TableCols <- columns\_df %>%  
 kable(format = knitr.table.format,  
 caption= (paste0("Análisis de correspondencias: ", colnames(originales)[2], ".")),  
 col.names = c("Inercia", "Coordenadas Dim. 1", "Cos2 Dim. 1",  
 "Coordenadas Dim. 2", "Cos2 Dim. 2"),  
 digits = 3,  
 align= c("c", "c", "c", "c", "c")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped", "bordered", "condensed",  
 position = "center",  
 font\_size = 12) %>%  
 row\_spec(0, bold= T, align = "c")  
   
 SolucionCA[[3]] <-TableCols  
   
 SolucionCA[[1]]  
 SolucionCA[[2]]  
 SolucionCA[[3]]

(#tab:unnamed-chunk-306)(#tab:unnamed-chunk-306)Análisis de correspondencias: Dimensión Matrix vs Valoración.

Dimensión/Eje 1

Dimensión/Eje 2

Inercia Total

0.175

0.051

% Inercia Principal

77.514

22.486

Acumulada

77.514

100.000

(#tab:unnamed-chunk-306)(#tab:unnamed-chunk-306)Análisis de correspondencias: DIMENSION.

Inercia

Coordenadas Dim. 1

Cos2 Dim. 1

Coordenadas Dim. 2

Cos2 Dim. 2

GRANDE

0.102

0.516

0.970

0.091

0.030

MEDIA

0.083

-0.549

0.715

0.347

0.285

REDUCIDA

0.041

-0.196

0.408

-0.236

0.592

(#tab:unnamed-chunk-306)Análisis de correspondencias: VALORACION.

Inercia

Coordenadas Dim. 1

Cos2 Dim. 1

Coordenadas Dim. 2

Cos2 Dim. 2

NORMAL

0.129

0.710

0.996

-0.044

0.004

OPTIMA

0.053

-0.278

0.773

-0.150

0.227

PESIMA

0.044

-0.157

0.122

0.421

0.878

En nuestro ejemplo, la primera tabla nos informa, esencialmente, de que la primera dimensión o eje recoge el 77,5% de la inercia total (varianza o comportamiento) de las categorías de los dos atributos o factores, mientras que la segunda dimensión o eje asume algo menos del 22,5%. Recordemos que esta interpretación es similar a la que se hace cuando se determinan los porcentajes de “comunalidad” que recogen las componentes calculadas, en el análisis de componentes principales. Recordemos también que, en el caso del análisis de correspondencias simple (dos atributos o factores), las dos dimensiones recogen el 100% de la inercia total (puesto que no se “descarta” ninguna dimensión de las calculadas).

Es habitual que los porcentajes de inercia principal asumidos por ambas dimensiones o ejes puedan estar bastante desequilibrados a favor del primer eje; sobre todo cuando hay una fuerte asociación entre los atributos o factores. Algunas características que pueden ayudar a un mayor equilibrio son:

* Distribución de los Datos: Una distribución equilibrada de las frecuencias en la tabla de contingencia puede ayudar a que la inercia total se distribuya de manera más uniforme.
* Número de Categorías: Tener un número similar de categorías en ambos atributos puede favorecer una distribución más equitativa de la inercia entre los ejes.
* Relaciones Simétricas: Que las relaciones entre las categorías de los atributos sean simétricas y no estén dominadas por unas pocas asociaciones fuertes (ninguna valoración domina de un modo extraordinario en alguna dimensión de matriz).
* Tamaño de la Muestra: Un tamaño de muestra grande.
* Homogeneidad de las Categorías: Que las categorías dentro de cada atributo sean relativamente homogéneas respecto a sus asociaciones con las categorías del otro atributo (la estructura de valoraciones es similar entre dimensiones).

En cuanto a la segunda tabla, destinada al análisis de las categorías del atributo o factor DIMENSION, podemos concluir lo siguiente: la mayor varianza (inercia) de las tres categorías de dimensión de la compañía matriz de pertenencia corresponde a “GRANDE”, seguida de “MEDIA” y “REDUCIDA”. En cuanto a la calidad de la representación de estas categorías en las dos dimensiones o ejes; las categorías “GRANDE” y “MEDIA” están representadas principalmente por la dimensión o eje 1; solo la categoría “REDUCIDA” viene mejor representada (levemente) por la dimensión o eje 2.

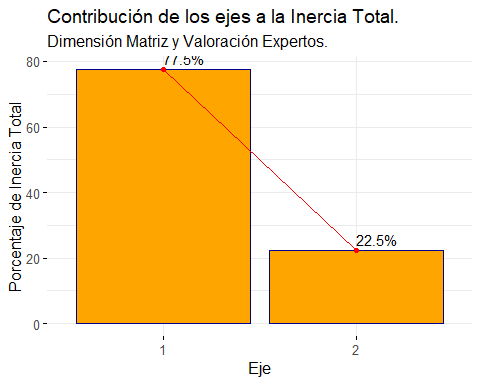
Por último, la tercera tabla recoge la representación de las categorías del atributo o factor VALORACION: “OPTIMA”, “NORMAL” y “PESIMA”. La mayor inercia o varianza es la de “NORMAL”, seguida de “OPTIMA” y, por último, “PESIMA”. En cuanto a la calidad de la representación en las dimensiones o ejes, “NORMAL” y “OPTIMA” vienen representadas, casi exclusivamente, por la dimensión o eje 1; mientras que ocurre lo contrario con la categoría “PESIMA”.

El principal output del análisis de correspondencias es el gráfico bidimensional donde se representan las categorías de los atributos o factores. En general, cuanto más cerca se localicen determinada categoría de un factor y determinada categoría del otro, mayor será la relación estadística (asociación) entre ambas categorías, lo que implica a su vez mayor asociación entre los atributos o factores.

Antes de presentar el gráfico bidimensional, vamos a construir un gráfico de barras con los porcentajes de la inercia total que asumen cada una de las dos dimenciones o ejes del gráfico. Esto es importante, ya que si la inercia recogida por la segunda dimensión o eje es muy pequeña, como a veces ocurre, habría que tener en cuenta, sobre todo, la localización de las categorías en la primera dimensión o eje, a la hora de establecer conclusiones en cuanto a la asociación o no entre categorías. También es necesario tener en cuenta la calidad con que cada eje representa a cada categoría de cada variable o factor, medida con el coseno cuadrado (cos2), como ya se ha comentado.

Para crear de un modo sencillo el gráfico de contribuciones de los ejes o dimensiones a la inercia total, puede recurrirse a la función fviz\_screeplot() del paquete {factoextra}:

# Gráfico de contribuciones de las dimensiones a la Inercia Total  
  
 library(factoextra)  
  
 gcontrib <- fviz\_screeplot(aceolicas,  
 addlabels= TRUE,  
 barcolor= "darkblue",  
 barfill= "orange",  
 linecolor= "red") +  
 labs(title= "Contribución de los ejes a la Inercia Total.",  
 subtitle = "Dimensión Matriz y Valoración Expertos.") +  
 ylab("Porcentaje de Inercia Total") +  
 xlab("Eje") +  
 theme(text = element\_text(size = 12))  
  
gcontrib



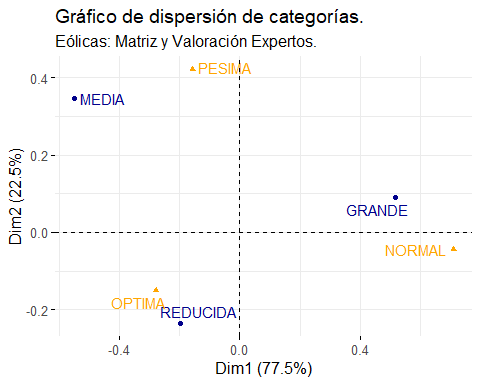
Puede observarse, como ya se recogió en la primera table de la solución, que la primera dimensión o eje asume el 77,5% de la inercia tital (varianza o comportamiento de las categorías), mientras que la segunda dimensión o eje asume el 22,5% restante.

En cuanto al gráfico bidimensional, puede obtenerse igualmente mediante la función fviz\_ca\_biplot() de {factoextra}. Este es el papel que juegan los diferentes argumentos de la función:

* **map = "symmetric"**: Representa tanto las filas como las columnas en el mismo espacio, utilizando la misma escala.
* **axes = c(1, 2)**: Selecciona las dos primeras dimensiones para el gráfico.
* **label = "all"**: Muestra las etiquetas de todas las categorías.
* **repel = TRUE**: Evita la superposición de etiquetas.
* **col.col y col.row**: Colores para las columnas y filas, respectivamente.
* **labs()**: Añade títulos y subtítulos al gráfico.
* **theme()**: Ajusta el tamaño del texto en el gráfico.

Es de destacar que el argumento map= controla cómo se representan las filas y columnas en el gráfico bidimensional del análisis de correspondencias. Este argumento tiene varias opciones que determinan la escala y la simetría de la representación. Para representar las posibles asociaciones entre las categorías de los dos factores implicados, la opción más apropiada es “symmetric”. Esta opción permite visualizar las filas y las columnas en el mismo espacio, facilitando la interpretación de las asociaciones entre las categorías de ambas variables.

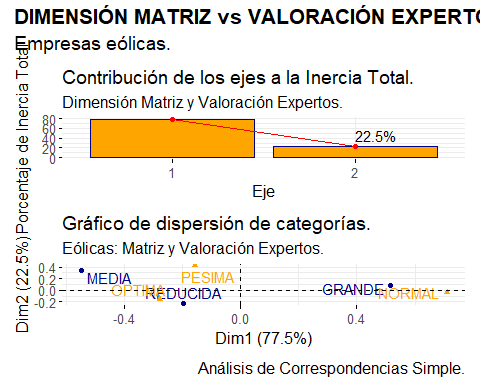
gbiplot <- fviz\_ca\_biplot (aceolicas,  
 axes= c(1,2),  
 label= "all",  
 repel = T,  
 col.col= "orange",  
 col.row= "darkblue",  
 map= "symmetric") +  
 labs(title= "Gráfico de dispersión de categorías.",  
 subtitle = "Eólicas: Matriz y Valoración Expertos.") +  
 theme(text = element\_text(size = 12))  
  
 gbiplot



En nuestro ejemplo, se aprecia cómo hay una intensa asociación entre la categoría de VALORACIÓN “OPTIMA”, y la categoria de DIMENSION (de la empresa matriz correspondiente) “REDUCIDA”. También se aprecia cierta asociación entre las categorías de DIMENSION “GRANDE” y VALORACION “NORMAL”. Por último la categoría de VALORACION “PESIMA”, y la categoría del factor o atributo DIMENSION “MEDIA” están bastante alejados de otras categorías, lo que implica que parece no estar muy asociadas con otras categorías específicas.

Finalmente, para una presentación más compacta de los dos últimos gráficos, puede recurrirse al paquete {patchwork}:

gcombinado <- gcontrib / gbiplot  
 gcombinado <- gcombinado +  
 plot\_annotation(title = "DIMENSIÓN MATRIZ vs VALORACIÓN EXPERTOS.",  
 subtitle = "Empresas eólicas.",  
 caption = "Análisis de Correspondencias Simple.",  
 theme = theme(plot.title = element\_text(size = 16, face = "bold"),  
 plot.subtitle = element\_text(size = 14),  
 plot.caption = element\_text(size = 12))  
 )  
  
 gcombinado



## 10.4 Modelos logaritmico-lineales aplicados a tablas de contingencia.

Volviendo a la verificación de la existencia de asociación entre factores, hemos de tener en cuenta la posibilidad de que los factores en estudio sean más de dos. En este caso multifactorial, no son aplicable las técnicas exploradas anteriormente, pensadas para el caso bifactorial (simple).

Una posibilidad que se nos ofrece es la aplicación de modelos logarítmico-lineales (log-lineales) a tablas de contingencia multifactoriales, que es lo que se tratará en este apartado.

Los modelos log-lineales aplicados a tablas de contingencia parten de la condición de independencia entre factores. Supongamos el caso simple, con una tabla de contingencia de dos factores o atributos, A y B. Bajo la hipótesis de independencia (absoluta o teórica) entre los factores, tendremos que cada frecuencia absoluta conjunta de la tabla se obtiene como:

con:

* ( ) es la frecuencia conjunta para la categoría o fila ( i ) del atributo o factor A, y la categoría o columna ( j ) del atributo o factor B).
* ( ) es el total de la fila ( i ) del atributo o factor A.
* ( ) es el total de la columna ( j ) del atributo o factor B.

Tomando logaritmos:

Y renombrando los términos:

Si no existe independencia entre ambas variables o factores, tendremos:

Los términos y se denominan efectos directos o principales, mientras que el término es el efecto conjunto o interacción entre los dos atributos o factores. Bajo la hipótesis de independencia entre los dos factores, ese efecto tomaría valor 0.

Si el modelo planteado solo tiene en la especificación los efectos directos, se dirá que es el **modelo de independencia**. Si se plantean todas las interacciones posibles entre los factores, se hablará del **modelo saturado**. El modelo *saturado* otorga un ajuste perfecto; pero es poco útil a la hora de extraer conclusiones relevantes. Se requiere un modelo que, aunque no ajuste al 100% las frecuencias, **recoja solo los efectos más importantes**.

Si algunos de los efectos más importantes (es decir, significativos en términos estadísticos) son conjuntos (interacciones), podremos concluir que existe asociación entre los factores del modelo (al menos, entre los que existan interacciones importantes, en el caso de más de dos factores). Si no es así, y el modelo que mejor representa la realidad es el de independencia, diremos que los factores son independientes unos de otros, y que no existe asociación (significativa) entre ellos.

Los modelos han de respetar siempre la regla de la jerarquía en su especificación: solo se podrá plantear en el modelo una interacción entre los atributos A y B si se han especificado los efectos directos de A y de B. O se podrá plantear una interacción conjunta entre los factores A, B y C si se han planteado también las interacciones entre A y B, B y C, y A y C.

Los modelos log-lineales se estiman por el método de máxima-verosimilitud.

La estrategia a seguir para estudiar la asociación entre factores será plantear diferentes especificaciones, y comprobar si son aptas para representar bien la realidad (tabla de contingencia). Eso se consigue mediante pruebas que evalúan la magnitud de los residuos, entendidos como la diferencia entre las frecuencias conjuntas realmente observadas, y las frecuencias estimadas por el modelo estimado. Las dos pruebas más comunes son la del *ratio de verosimilitud*, y la de *Pearson*.

SI existen varios modelos que, desde el punto de vista de las pruebas anteriores, son aptos para representarrazonablemente la realidad, se eligirá el mejor de ellos mediante algún criterio específico, como puede ser aquel que minimice el Criterio de Información de Akaike, que penaliza, para una capacidad de explicación semejante, al modelo con una estructura más compleja (más términos).

Los parámetros estimados finalmente, correspondientes a la mejor especificación, informarán si los efectos directos e interacciones o efectos conjuntos entre los distintos factores del modelo final tienen una influencia positiva o negativa sobre el valor de las diferentes frecuencias conjuntas de la tabla de contingencia, lo que informará no solo de si existe asociación entre factores o no; sino también de, en caso de existir, de cómo se materializa tal asociación.

Para ejemplificar la especificación, estimación e interpretación de los modelos log-lineales aplicados a tablas de contingencia, vamos a plantear un caso en el que entran en juego 3 factores o atributos, que caracterizan a un conjunto o muestra de 474 empresas eólicas. Estos factores o atributos son:

* **DIMENSION:** tamaño del grupo empresarial al que pertenece la empresa en cuestión. Tiene tres niveles: grande, media y reducida.
* **AUTOFINA:** capacidad de autofinanciación de la empresa a medio y largo plazo. Tiene tres niveles: alta, positiva y negativa.
* **FJUR**: forma jurídica. Tiene dos posibles categorías: Sociedad anónima o Sociedad limitada.

Los datos se encuentran alojados en la hoja “Datos” del archivo de Microsoft® Excel® “[eolica\_contingencia2.xlsx](https://docs.google.com/spreadsheets/d/1f0OQaxyzjKIIXrqhCFL1vIBNcLLGx6R4/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true)”. El código a desarrollar está disponible en el *script* “[loglineal\_eolica.R](https://drive.google.com/file/d/1xMmM2nrSHBCMU1EEDePb2BRuv_TbD-7r/view?usp=sharing)”. Puede desarrollarse el ejemplo creando para ello un proyecto de RStudio, por ejemplo, el proyecto “loglineal”.

Comenzando a ejecutar el código presente en el *script*, la primera parte se dedica, como es habitual, a la gestión de los datos: borrado previo de memoria, importación de los datos alojados en el archivo de Excel y volcado a un *data frame*, corrección del nombre de las filas de este, y selección de las variables categóricas del estudio:

# Analisis de Asociacion y modelos log-lineales de eolicas  
# Disculpen por la falta de tildes!  
  
rm(list = ls())  
  
# DATOS  
  
 # Importando datos  
  
 library (readxl)  
 eolicas <- read\_excel("eolica\_contingencia2.xlsx", sheet ="Datos")  
 eolicas <- data.frame(eolicas, row.names = 1)  
 summary (eolicas)  
  
 # Seleccionando factores/atributos para el analisis  
  
 library(dplyr)  
 originales2 <- eolicas %>%  
 select(DIMENSION, AUTOFINA, FJUR)  
 summary (originales2)

# Analisis de Asociacion y modelos log-lineales de eolicas  
# Disculpen por la falta de tildes!  
  
rm(list = ls())  
  
# DATOS  
  
 # Importando datos  
  
 library (readxl)  
 eolicas <- read\_excel("eolica\_contingencia2.xlsx", sheet ="Datos")  
 eolicas <- data.frame(eolicas, row.names = 1)  
 summary (eolicas)  
  
 # Seleccionando factores/atributos para el analisis  
  
 library(dplyr)  
 originales2 <- eolicas %>%  
 select(DIMENSION, AUTOFINA, FJUR)  
 summary (originales2)

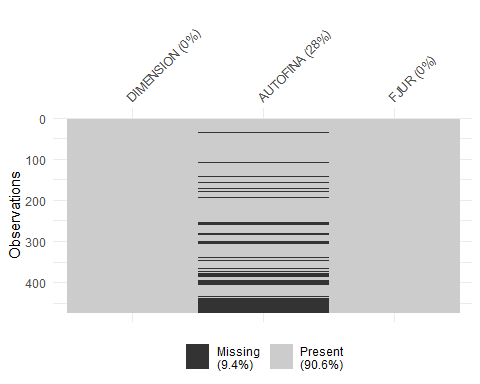
## MARGEN SOLVENCIA COM   
## Min. : -4124.22 Min. :-1162.21 Length:474   
## 1st Qu.: 14.09 1st Qu.: 13.95 Class :character   
## Median : 42.64 Median : 39.36 Mode :character   
## Mean : 794.48 Mean : 38.19   
## 3rd Qu.: 64.82 3rd Qu.: 72.69   
## Max. :298700.00 Max. : 100.00   
## NA's :87   
##   
## FJUR ING NCOMP   
## Length:474 Min. : 0.1 Min. : 0   
## Class :character 1st Qu.: 392.9 1st Qu.: 3   
## Mode :character Median : 3221.0 Median : 78   
## Mean : 8128.6 Mean : 1193   
## 3rd Qu.: 7709.2 3rd Qu.: 392   
## Max. :364989.0 Max. :72434   
## NA's :87 NA's :1   
##   
## RES ACTIVO FPIOS   
## Min. :-16107.21 Min. : 1.0 Min. : -51817.4   
## 1st Qu.: 3.09 1st Qu.: 470.2 1st Qu.: 67.1   
## Median : 382.71 Median : 8103.0 Median : 1578.0   
## Mean : 2164.98 Mean : 38994.2 Mean : 15932.6   
## 3rd Qu.: 2413.00 3rd Qu.: 31147.1 3rd Qu.: 8503.9   
## Max. : 78290.00 Max. :2429299.0 Max. :1382020.0   
## NA's :52   
##   
## RENECO RENFIN LIQUIDEZ   
## Min. : -269.05 Min. : -687.42 Min. : 0.000   
## 1st Qu.: 0.00 1st Qu.: 0.00 1st Qu.: 0.621   
## Median : 6.01 Median : 16.29 Median : 1.768   
## Mean : 151.08 Mean : 194.65 Mean : 23.474   
## 3rd Qu.: 17.51 3rd Qu.: 46.66 3rd Qu.: 3.947   
## Max. :66538.10 Max. :67943.49 Max. :1622.359   
## NA's :14   
##   
## APALANCA DIMENSION AUTOFINA   
## Min. : -7016.77 Length:474 Length:474   
## 1st Qu.: 0.00 Class :character Class :character   
## Median : 22.64 Mode :character Mode :character   
## Mean : 769.68   
## 3rd Qu.: 201.63   
## Max. :177381.90   
##   
## Length Class Mode   
## 474 character character

El *data frame* “originales2” albergará los tres factores o atributos del análisis:

## DIMENSION AUTOFINA FJUR   
## Length:474 Length:474 Length:474   
## Class :character Class :character Class :character   
## Mode :character Mode :character Mode :character

Posteriormente se desarrolla la localización y tratamiento de *missing data*, ya que para realizar el análisis es necesario que todos los casos posean dato en todas las variables. Para tener una idea general, se puede recurrir a la función vis\_miss() del paquete {visdat}, que localizará gráficamente los *missing values* de las diferentes variables, y calculará el porcentaje de casos que supone, con respecto al total de observaciones. En el ejemplo, los casos con missing data se concentran en el atributo o factor AUTOFINA, con 133 casos. El nombre concreto de los casos afectados se visualiza con un filtro construido a partir de la función filter() del paquete {dplyr}:

# Identificando missing values.  
  
 library(visdat)  
 vis\_miss(originales2)



originales2 %>% filter(is.na(DIMENSION) |  
 is.na(AUTOFINA)|  
 is.na(FJUR)) %>%  
 select(DIMENSION, AUTOFINA, FJUR)

## DIMENSION AUTOFINA FJUR  
## Sierra de Selva SL GRANDE <NA> Sociedad limitada  
## Eolica de Rubio SL GRANDE <NA> Sociedad limitada  
## Fuerzas Energeticas DEL SUR de Europa II SL. GRANDE <NA> Sociedad limitada  
## Parque Eolico de LA Bobia Y SAN Isidro Sociedad Limitada REDUCIDA <NA> Sociedad limitada  
## Engasa Eolica SA MEDIA <NA> Sociedad anonima  
## Parque Eolico de Ameixenda-Filgueira SL REDUCIDA <NA> Sociedad limitada  
## Compañia Eolica Granadina SA REDUCIDA <NA> Sociedad anonima  
## Suresa Retama S.L. GRANDE <NA> Sociedad limitada  
## Parque Eolico de Adraño SL REDUCIDA <NA> Sociedad limitada  
## M Torres Desarrollos Energeticos SL REDUCIDA <NA> Sociedad limitada  
## Corporacion Eolica de Valdivia SL GRANDE <NA> Sociedad limitada  
## Parque Eolico de A Ruña SL REDUCIDA <NA> Sociedad limitada  
## Molinos DEL Moncayo SL. REDUCIDA <NA> Sociedad limitada  
## Eolico Alijar SA GRANDE <NA> Sociedad anonima  
## Parque Eolico de Vicedo SL REDUCIDA <NA> Sociedad limitada  
## Eolica de Villanueva SL GRANDE <NA> Sociedad limitada  
## Parque Eolico de Virxe DO Monte SL REDUCIDA <NA> Sociedad limitada  
## Estructuras Y Revestimiento de Galicia SL REDUCIDA <NA> Sociedad limitada  
## Energias Renovables EL Abra SL GRANDE <NA> Sociedad limitada  
## Proyectos Eolicos Aragoneses SL REDUCIDA <NA> Sociedad limitada  
## Soslaires Canarias SL REDUCIDA <NA> Sociedad limitada  
## Eolica Cantabria SA REDUCIDA <NA> Sociedad anonima  
## Señorio de Bariain SA REDUCIDA <NA> Sociedad anonima  
## Energetica DEL Montalt SL REDUCIDA <NA> Sociedad limitada  
## Eolica Lodosa SL REDUCIDA <NA> Sociedad limitada  
## Parque Eolico LA Union S.L. REDUCIDA <NA> Sociedad limitada  
## Eolica Pueyo SL REDUCIDA <NA> Sociedad limitada  
## Parsona Corporacion SL. REDUCIDA <NA> Sociedad limitada  
## Infraestructuras Electricas LA Mudarra SL. GRANDE <NA> Sociedad limitada  
## Eolica Unzue SL REDUCIDA <NA> Sociedad limitada  
## Technical Services Wind SL. REDUCIDA <NA> Sociedad limitada  
## Saltos DEL Mundo SL REDUCIDA <NA> Sociedad limitada  
## Sistemas Energeticos LA Plana SA GRANDE <NA> Sociedad anonima  
## Renovalia Reserve SL. MEDIA <NA> Sociedad limitada  
## Eolpop SL. REDUCIDA <NA> Sociedad limitada  
## Emprendimientos Y Desarrollo de Iniciativas Energeticas SL REDUCIDA <NA> Sociedad limitada  
## Megaturbinas Arinaga SA REDUCIDA <NA> Sociedad anonima  
## Sunterra XXI Sociedad Limitada. REDUCIDA <NA> Sociedad limitada  
## Aizdegi SL REDUCIDA <NA> Sociedad limitada  
## Parque Eolico LA Sargilla Sociedad Anonima. REDUCIDA <NA> Sociedad anonima  
## Intercon SA REDUCIDA <NA> Sociedad anonima  
## Altosalvo SL REDUCIDA <NA> Sociedad limitada  
## Bluefloat Energy International SL. REDUCIDA <NA> Sociedad limitada  
## Corolla Power 1 SL REDUCIDA <NA> Sociedad limitada  
## Corolla Power 3 SL REDUCIDA <NA> Sociedad limitada  
## Parque Energetico de G C SL REDUCIDA <NA> Sociedad limitada  
## Dapasa Servicios E Inversiones SL REDUCIDA <NA> Sociedad limitada  
## EL Guijorral SL REDUCIDA <NA> Sociedad limitada  
## Minicentrales Bouza Vella SL REDUCIDA <NA> Sociedad limitada  
## Locus Mentis SL REDUCIDA <NA> Sociedad limitada  
## Vento Laracha SL. MEDIA <NA> Sociedad limitada  
## Huerto Solar EL Tronco SL REDUCIDA <NA> Sociedad limitada  
## Fotovoltaica LA Solana SL REDUCIDA <NA> Sociedad limitada  
## Fargo MAS 3 SL REDUCIDA <NA> Sociedad limitada  
## Naduele SL REDUCIDA <NA> Sociedad limitada  
## Maririas Energy SL REDUCIDA <NA> Sociedad limitada  
## Helios Almaden Sociedad Limitada. REDUCIDA <NA> Sociedad limitada  
## Explotaciones MI Cobijo SL. REDUCIDA <NA> Sociedad limitada  
## Catral Renovables SL REDUCIDA <NA> Sociedad limitada  
## Energia Solar Turolense SL REDUCIDA <NA> Sociedad limitada  
## AV Serra de Liñares SL. MEDIA <NA> Sociedad limitada  
## Parque Eolico Donado SL GRANDE <NA> Sociedad limitada  
## AV Paxareiras SL. MEDIA <NA> Sociedad limitada  
## Terranova Energy Corporation SA GRANDE <NA> Sociedad anonima  
## AV Serra DO Farelo SL. MEDIA <NA> Sociedad limitada  
## Eolica de Cordales BIS SL. MEDIA <NA> Sociedad limitada  
## AV Cernego SL. MEDIA <NA> Sociedad limitada  
## AV Outeiro Rubio SL. MEDIA <NA> Sociedad limitada  
## Airosa Vento SL MEDIA <NA> Sociedad limitada  
## Eolica de Cordales SL. MEDIA <NA> Sociedad limitada  
## Enerfin Renovables IV SL. GRANDE <NA> Sociedad limitada  
## Sistemas Energeticos Jupiter Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Sistemas Energeticos Oberon Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Sistemas Energeticos Pluton Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Sistemas Energeticos Saturno Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Sistemas Energeticos Titan Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Sistemas Energeticos Urano Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Sistemas Energeticos Venus Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Puerto Rosario Solar 3 Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## IM2 Energia Solar Proyecto 24 SL. MEDIA <NA> Sociedad limitada  
## Parque Eolico Punta Langosteira SL. REDUCIDA <NA> Sociedad limitada  
## Desarrollos Fotovoltaicos Fuentes SL. GRANDE <NA> Sociedad limitada  
## Guadalaviar Consorcio Eolico SA. GRANDE <NA> Sociedad anonima  
## Desarrollo Eolico LAS Majas Xxxi SL. GRANDE <NA> Sociedad limitada  
## Puerto Rosario Solar 2 Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Energias Renovables de Hidra SL. GRANDE <NA> Sociedad limitada  
## Energias Renovables de Cilene SL. GRANDE <NA> Sociedad limitada  
## Desarrollo Eolico LAS Majas XV SL. GRANDE <NA> Sociedad limitada  
## Renovacyl SA GRANDE <NA> Sociedad anonima  
## Sistemas Energeticos DEL SUR SA GRANDE <NA> Sociedad anonima  
## Eolica Santa Teresa Sociedad Limitada. REDUCIDA <NA> Sociedad limitada  
## Lan2030 Toroña S.L. MEDIA <NA> Sociedad limitada  
## Gerr Grupo Energetico XXI SA GRANDE <NA> Sociedad anonima  
## Belidia Energy SL. MEDIA <NA> Sociedad limitada  
## Parc Tramuntana SL. REDUCIDA <NA> Sociedad limitada  
## Aerogeneracion Galicia SL REDUCIDA <NA> Sociedad limitada  
## Greenalia Wind Power A Marabilla, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Alto DO Rodicio II, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power AS Lagoas, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Campos Vellos, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Cardon, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Cedeira, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Cervo, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Cordobelas, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Coto DOS Chaos, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Dunas, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Esteiro, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Guanche, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Huracan, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Lamas II, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Mojo, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Montoxo, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power O Barral, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Piñeiro, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Punta Candieira, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Regoa, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power SAN Isidro, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power SAN Roman, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Suime, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Teixido, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Tormenta, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Vaqueira, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Vilas, S.L. GRANDE <NA> Sociedad limitada  
## Greenalia Wind Power Xesteiron, S.L. GRANDE <NA> Sociedad limitada  
## Infraestructuras Para EL Desarrollo de Energias Renovables SL MEDIA <NA> Sociedad limitada  
## Renovables DEL Cantabrico Sociedad Limitada. REDUCIDA <NA> Sociedad limitada  
## Sistemas Energeticos Erbania 1, Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Sistemas Energeticos Erbania 2, Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Sistemas Energeticos Marte Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Sistemas Energeticos Mercurio Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Sistemas Energeticos Neptuno Sociedad Limitada. GRANDE <NA> Sociedad limitada  
## Wind Premier Monte Redondo, S.L. GRANDE <NA> Sociedad limitada  
## Wind Premier Serra Pequena, S.L. GRANDE <NA> Sociedad limitada

Se decide eliminar los casos que carecen de valor en el atributo AUTOFINA, lo que se realiza mediante otro filtro:

originales2 <- originales2 %>%  
 filter(! is.na(DIMENSION) &  
 ! is.na(AUTOFINA) &  
 ! is.na(FJUR))

El siguiente bloque se ocupa de crear el objeto “table” que recoge la tabla de contingencia formada al cruzar la distribución de las frecuencias o casos entre las diferentes categorías de los distintos factores. La función para convertir el *data frame* en una estructura de almacenamiento de datos especial llamada table (que es la **tabla de contingencia**) es precisamente table(). La tabla de contingencia construida, de nombre “tab.originales2”, es posteriormente presentada como una tabla diseñada a partir de la función kable() del paquete {knitr}, y otras funciones incluidas en el paquete {kableExtra}:

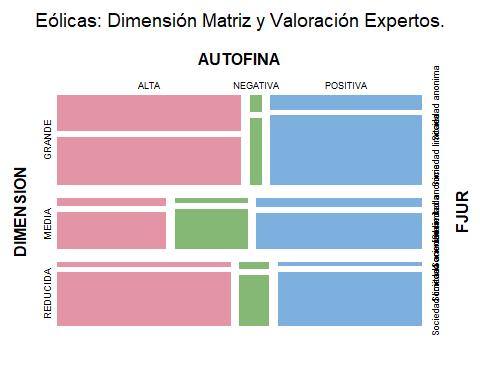
# TABLA DE CONTINGENCIA  
  
 tab.originales2 <- table(originales2)  
  
 library(knitr)  
 library(kableExtra)  
 knitr.table.format = "html"  
  
 tab.originales2 %>%  
 kable(format = knitr.table.format,  
 caption="Empresas eólicas") %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = "striped", "bordered", "condensed",  
 position = "center",  
 font\_size = 12) %>%  
 row\_spec(0, bold= T, align = "c")

Table 10.5: Empresas eólicas

| DIMENSION | AUTOFINA | FJUR | Freq |
| --- | --- | --- | --- |
| GRANDE | ALTA | Sociedad anonima | 34 |
| MEDIA | ALTA | Sociedad anonima | 5 |
| REDUCIDA | ALTA | Sociedad anonima | 4 |
| GRANDE | NEGATIVA | Sociedad anonima | 1 |
| MEDIA | NEGATIVA | Sociedad anonima | 2 |
| REDUCIDA | NEGATIVA | Sociedad anonima | 1 |
| GRANDE | POSITIVA | Sociedad anonima | 11 |
| MEDIA | POSITIVA | Sociedad anonima | 8 |
| REDUCIDA | POSITIVA | Sociedad anonima | 3 |
| GRANDE | ALTA | Sociedad limitada | 45 |
| MEDIA | ALTA | Sociedad limitada | 22 |
| REDUCIDA | ALTA | Sociedad limitada | 49 |
| GRANDE | NEGATIVA | Sociedad limitada | 4 |
| MEDIA | NEGATIVA | Sociedad limitada | 16 |
| REDUCIDA | NEGATIVA | Sociedad limitada | 8 |
| GRANDE | POSITIVA | Sociedad limitada | 54 |
| MEDIA | POSITIVA | Sociedad limitada | 33 |
| REDUCIDA | POSITIVA | Sociedad limitada | 41 |

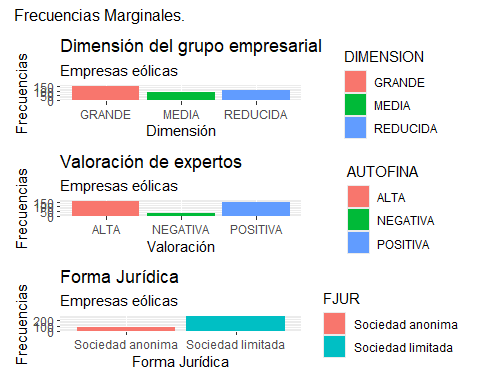
Un modo visual de obtener una primera idea de las relaciones que se incluyen en la tabla es construir un gráfico de mosaico, con la función mosaic() del paquete {vcd}, en la que el área de los diferentes rectángulos sea proporcional a la frecuencia conjunta correspondiente:

# Representación gráfica de la tabla con mosaico  
  
 library (vcd)  
 mosaic(tab.originales2,  
 main = "Eólicas: Dimensión Matriz y Valoración Expertos.",  
 shade = T,  
 gp = shading\_Marimekko(tab.originales2),  
 main\_gp = gpar(fontsize = 14),  
 sub\_gp = gpar(fontsize = 12),  
 labeling\_args = list(gp\_labels = gpar(fontsize = 7)))



En cuanto a las frecuencias marginales de cada nivel o categoría de los tres atributos o factores, pueden representarse estas mediante gráficos de barras, generados mediante las funciones del paquete {ggplot2}, y reunidos en una sola imagen mediante el paquete {patchwork}. El código es el siguiente:

# Representando frecuencias de categorias en factores  
  
 library (ggplot2)  
 library (patchwork)  
  
 g1b <- ggplot(originales2, map= aes(x= DIMENSION,  
 fill = DIMENSION)) +  
 geom\_bar() +  
 ggtitle("Dimensión del grupo empresarial",  
 subtitle = "Empresas eólicas") +   
 ylab("Frecuencias") +  
 xlab("Dimensión")   
  
 g2b <- ggplot(originales2, map= aes(x= AUTOFINA,  
 fill = AUTOFINA)) +  
 geom\_bar() +  
 ggtitle("Valoración de expertos",  
 subtitle = "Empresas eólicas") +   
 ylab("Frecuencias") +  
 xlab("Valoración")   
  
 g3b <- ggplot(originales2, map= aes(x= FJUR,  
 fill = FJUR)) +  
 geom\_bar() +  
 ggtitle("Forma Jurídica",  
 subtitle = "Empresas eólicas") +   
 ylab("Frecuencias") +  
 xlab("Forma Jurídica")   
  
 (g1b / g2b / g3b) + plot\_annotation(title = "Frecuencias Marginales.",  
 theme = theme(plot.title = element\_text(size = 12)))



Los modelos log-lineales se especifican y estiman mediante la función loglm() de la librería {MASS} (el paquete MASS lo hemos activado junto a la propia función loglm() ya que, de hacerlo con la función library(), se crea un conflicto con la función select() del paquete {dplyr}).

Cuando se estiman los modelos, los diferentes resultados se almacenan en una lista. Entre estos elementos se encuentran los parámetros estimados, que, dentro de la lista de resultados, se almacenan, a su vez, en una estructura que, según la especificación que se elija del modelo, puede llegar a ser compleja. para facilitar el trabajo de extracción de los parámetros y almacenamiento en un data frame, se ha desarrollado una función denominada extraer\_coeficientes(). Debido a la complejidad del código, y antes de proceder a estimar los modelos, exponemos el código de la función, que recibe como input o argumento el nombre de un modelo log-lineal estimado, y devuelve como output un data frame con los coeficientes o parámetros estimados:

# MODELOS LOG-LINEALES  
  
# Función para extraer coeficientes y sus nombres de un modelo ##########  
extraer\_coeficientes <- function(modelo) {  
 # Extraer los parámetros del modelo  
 parametros <- modelo$param  
   
 # Crear listas para almacenar nombres y valores de coeficientes  
 coef\_names <- c()  
 coef\_values <- c()  
   
 # Función para generar nombres de coeficientes  
 generate\_coef\_name <- function(levels) {  
 return(paste(levels, collapse = ":"))  
 }  
   
 # Recorrer los coeficientes y extraer los nombres y valores  
 for (term in names(parametros)) {  
 if (term == "(Intercept)") {  
 coef\_names <- c(coef\_names, "T. Independiente")  
 coef\_values <- c(coef\_values, parametros[[term]])  
 } else if (is.matrix(parametros[[term]])) {  
 # Si es una matriz, recorrer filas y columnas  
 for (i in 1:nrow(parametros[[term]])) {  
 for (j in 1:ncol(parametros[[term]])) {  
 coef\_names <- c(coef\_names,  
 paste(rownames(parametros[[term]])[i],  
 colnames(parametros[[term]])[j],  
 sep = ":"))  
 coef\_values <- c(coef\_values, parametros[[term]][i, j])  
 }  
 }  
 } else if (is.array(parametros[[term]])) {  
 # Si es un array de más de dos dimensiones  
 dims <- dim(parametros[[term]])  
 dimnames\_list <- dimnames(parametros[[term]])  
 for (i in seq\_len(dims[1])) {  
 for (j in seq\_len(dims[2])) {  
 for (k in seq\_len(dims[3])) {  
 coef\_name <- paste(dimnames\_list[[1]][i],  
 dimnames\_list[[2]][j],  
 dimnames\_list[[3]][k],  
 sep = ":")  
 coef\_names <- c(coef\_names, coef\_name)  
 coef\_values <- c(coef\_values,  
 parametros[[term]][i, j, k])  
 }  
 }  
 }  
 } else {  
 levels <- names(parametros[[term]])  
 for (level in levels) {  
 coef\_names <- c(coef\_names,  
 generate\_coef\_name(c(term, level)))  
 coef\_values <- c(coef\_values,  
 parametros[[term]][[level]])  
 }  
 }  
 }  
   
 # Verificar la longitud de los vectores antes de crear el data frame  
 if (length(coef\_names) == length(coef\_values)) {  
 tabla\_coeficientes <- data.frame(Coefficient = coef\_names,  
 Value = coef\_values,  
 stringsAsFactors = FALSE)  
 return(tabla\_coeficientes)  
 } else {  
 stop("Error: Las longitudes de coef\_names y coef\_values no coinciden.")  
 }  
}  
############################################################################

Del mismo modo, se ha creado otra función para facilitar la pesentación de la información más importante que se genera con la estimación del modelo loglineal. Es la función generar\_solucion(). Esta función recibe como argumento o *intput* el nombre del modelo estimado, y devuelve tres elementos: dos tablas diseñadas con kable(), que se almacenan en una lista, y un gráfico de *mosaico* que recoge los **residuos del modelo** (diferencias entre las frecuencias conjuntas reales u observadas de la tabla de contingencia, y sus estimaciones por parte del modelo). La primera de las tablas recoge las pruebas de validez de los modelos del *ratio de verosimilitud* (deviance), y de *Pearson*: nombre de la prueba, estadístico del contraste, grados de libertad y *p-valor*. La segunda tabla recoge los coeficientes o parámetros estimados, para lo cual la función llama, a su vez, a la función extraer\_coeficientes(), toma el data frame que construya tal función, y la usa como base para diseñar la segunda tabla.

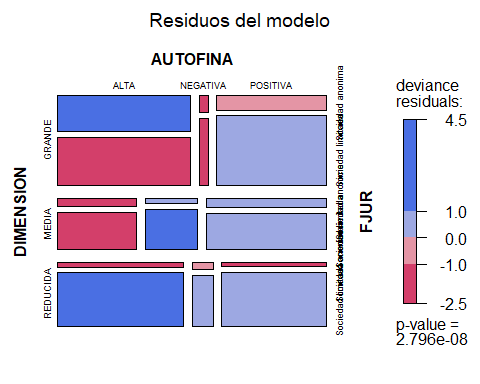
El código de la función es el siguiente:

############################################################################  
# Función generar tablas y gráfico a partir de modelo log-lineal  
  
generar\_solucion <- function(modelo) {  
 # Extraer la información del modelo  
 summary\_modelo <- summary(modelo)  
   
 # Crear una tabla con la información relevante de las pruebas de validez  
 tabla\_informacion <- data.frame(  
 Statistic = c("Likelihood Ratio", "Pearson"),  
 X2 = c(summary\_modelo$tests[1, "X^2"], summary\_modelo$tests[2, "X^2"]),  
 df = c(summary\_modelo$tests[1, "df"], summary\_modelo$tests[2, "df"]),  
 P\_value = c(summary\_modelo$tests[1, "P(> X^2)"], summary\_modelo$tests[2, "P(> X^2)"]),  
 stringsAsFactors = FALSE  
 )  
   
 # Formatear la tabla usando kable  
 independencia\_valida\_tab <- tabla\_informacion %>%  
 kable(caption ="Validación del modelo",  
 format = "html",  
 col.names = c("Prueba", "Estadístico", "Grados Libertad", "P-valor")) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = c("striped", "bordered", "condensed"),  
 position = "center",  
 font\_size = 12) %>%  
 row\_spec(0, bold= T, align = "c")  
   
 # Extraer los coeficientes del modelo  
 independencia\_df <- extraer\_coeficientes(modelo)  
   
 # Formatear la tabla de coeficientes usando kable  
 independencia\_coef\_tab <- independencia\_df %>%  
 kable(format = "html",  
 caption ="Coeficientes del modelo",  
 digits = 3) %>%  
 kable\_styling(full\_width = F,  
 bootstrap\_options = c("striped", "bordered", "condensed"),  
 position = "center",  
 font\_size = 12) %>%  
 row\_spec(0, bold= T, align = "c")  
   
 # Crear el gráfico de mosaico con los residuos y mostrarlo en pantalla directamente  
 plot(modelo, panel = mosaic,  
 main="Residuos del modelo",  
 residuals\_type = c("deviance"),  
 gp = shading\_hcl,  
 gp\_args = list(interpolate = c(0, 1)),  
 main\_gp = gpar(fontsize = 14),  
 sub\_gp = gpar(fontsize = 9),  
 labeling\_args = list(gp\_labels = gpar(fontsize = 7)))  
   
 # Guardar las tablas en una lista  
 solucion\_nombre <- paste0("solucion\_", deparse(substitute(modelo)))  
 solucion\_lista <- list(  
 Informacion = independencia\_valida\_tab,  
 Coeficientes = independencia\_coef\_tab  
 )  
   
 assign(solucion\_nombre, solucion\_lista, envir = .GlobalEnv)  
}  
##########################################################################

Con el par de funciones anteriores, es fácil estimar un modelo log-lineal aplicado a tablas de contingencia y recopilar la información más relevante.

Comenzaremos por el **modelo de independencia**, que plantea que solo existen efectos directos en la determinación de las frecuencias conjuntas de la tabla (no-asociación entre factores). La especificación y estimación del modelo es la siguiente:

# Modelo Independencia.  
  
 modelo\_indep <- MASS::loglm(~ DIMENSION + AUTOFINA + FJUR,  
 data= tab.originales2)  
  
 generar\_solucion(modelo\_indep)



Puede comprobarse que se ha creado la lista “solucion\_modelo\_indep”, que guarda dos elementos ($Informacion y $Coeficientes), y se ha generado el gráfico de mosaico de los residuos. Cuanto más intensos son los colores, mayores serán los residuos (en valor absoluto) y, por lo tanto, peor será el ajuste obtenido, lo que se deberá a que se ha considerado (erróneamente) que no hay interacción entre los factores (no asociación o independencia). En cuanto a los tonos de color:

* **Rectángulos azulados**: Indican que la frecuencia observada es mayor que la frecuencia estimada por el modelo. Es decir, el modelo subestima la frecuencia observada.
* **Rectángulos rojizos**: Indican que la frecuencia observada es menor que la frecuencia estimada por el modelo. Es decir, el modelo sobreestima la frecuencia observada.

En nuestro caso, existen frecuencias en los que los residuos toman colores bastante intensos, lo que hace pensar en que no se ha producido un buen ajuste. Como este modelo planteaba un escenario en el que no hay asociación entre los atributos; el gráfico parece apoyar la hipótesis de que sí existe asociación, al menos entre algunos de los factores o atributos.

Vamos a interpretar ahora el primer elemento de la lista “solucion\_modelo\_indep”, que es la tabla donde se disponen las pruebas de validez del modelo. Para obtener la tabla ejecutaremos:

solucion\_modelo\_indep$Informacion

Table 10.6: Validación del modelo

Prueba

Estadístico

Grados Libertad

P-valor

Likelihood Ratio

59.48973

12

0

Pearson

66.55540

12

0

En ambas pruebas (ratio de verosimilitud y Pearson) se obtiene un p-valor de 0. Teniendo en cuenta que la hipótesis nula de ambas pruebas es que existe un buen ajuste; la conclusión es que, según los resultados, **el modelo de independencia no es capaz de representar la realidad con suficiente precisión** (de ahí la magnitud de los residuos). Esto se puede interpretar, a su vez, como que **se rechaza la hipótesis de independencia entre los factores** y se admite que existe asociación entre, al menos, algunos de ellos.

Por último, vamos a mostrar el segundo elemento de la lista “solucion\_modelo\_indep”, que es la tabla donde se disponen los parámetros estimados del modelo, que en este caso corresponden solo a efectos directos o principales. Para obtener la tabla ejecutaremos:

solucion\_modelo\_indep$Coeficientes

Table 10.6: Coeficientes del modelo

Coefficient

Value

T. Independiente

2.479

DIMENSION:GRANDE

0.297

DIMENSION:MEDIA

-0.253

DIMENSION:REDUCIDA

-0.044

AUTOFINA:ALTA

0.554

AUTOFINA:NEGATIVA

-1.049

AUTOFINA:POSITIVA

0.496

FJUR:Sociedad anonima

-0.686

FJUR:Sociedad limitada

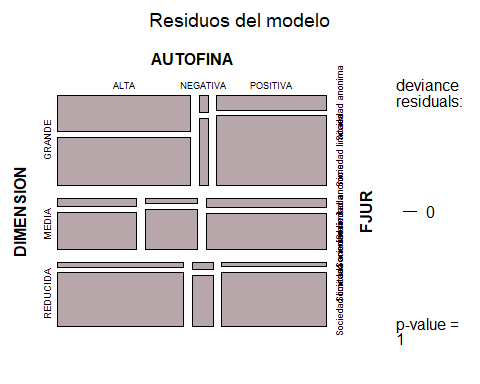
0.686

ssage=FALSE, warning=FALSE}

Puede destacarse el hecho de que, que la capacidad de autofinanciación sea negativa, tiende a provocar una reducción del número de casos en las frecuencias conjuntas implicadas (signo negativo) También el hecho de que la forma jurídica adoptada sea la de *Sociedad Anónima*. En el extremo opuesto, destacan los signos positivos de la forma jurídica de *Sociedad Limitada* y la capacidad de autofinanciación alta, lo que implica que estas categorías tienden a acumular más frecuencias.

Pasamos ahora al **modelo saturado**, en el cuál se especifican todos los efectos directos o principales de los factores, y todas las interacciones posibles entre dichos factores (interacciones dos a dos, y la interacción entre los tres factores de modo simultáneo). Este modelo, en la práctica, no es relevante porque, aunque explica al 100% las frecuencias observadas, no indica cuál de los niveles de los factores o atributos y sus interacciones son los más relevantes (modelo redundante). Para su estimación, simplemente se sustituyen los signos “+” del modelo anterior por los signos “\*”. El modelo se denominará “modelo\_sat”:

# Modelo Saturado.  
  
 modelo\_sat <- MASS::loglm(~ DIMENSION \* AUTOFINA \* FJUR,  
 data= tab.originales2)  
  
 generar\_solucion(modelo\_sat)



El gráfico de mosaico muestra que, obviamente, todos los residuos son 0, ya que coinciden las frecuencias observadas y las estimadas por el modelo. En cuanto a la tabla de validación:

solucion\_modelo\_sat$Informacion

Table 10.7: Validación del modelo

Prueba

Estadístico

Grados Libertad

P-valor

Likelihood Ratio

0

0

1

Pearson

0

0

1

En ambas pruebas el p-valor es 1, dado que no se rechaza la hipótesis de que el modelo representa adecuadamente la realidad (de hecho, la representa perfectamente). Pero, como hemos dicho, desde el punto de vista del análisis estructural el modelo saturado no es muy útil, ya que no distingue entre los efectos e interacciones relevantes y los que son poco importantes. Por último, mostraremos la tabla con los coeficientes estimados, correspondientes tanto a los efectos principales o directos como a las interacciones:

solucion\_modelo\_sat$Coeficientes

Table 10.7: Coeficientes del modelo

Coefficient

Value

T. Independiente

2.279

DIMENSION:GRANDE

0.239

DIMENSION:MEDIA

0.012

DIMENSION:REDUCIDA

-0.250

AUTOFINA:ALTA

0.606

AUTOFINA:NEGATIVA

-1.124

AUTOFINA:POSITIVA

0.517

FJUR:Sociedad anonima

-0.858

FJUR:Sociedad limitada

0.858

GRANDE:ALTA

0.542

GRANDE:NEGATIVA

-0.701

GRANDE:POSITIVA

0.159

MEDIA:ALTA

-0.547

MEDIA:NEGATIVA

0.566

MEDIA:POSITIVA

-0.019

REDUCIDA:ALTA

0.004

REDUCIDA:NEGATIVA

0.135

REDUCIDA:POSITIVA

-0.139

GRANDE:Sociedad anonima

0.315

GRANDE:Sociedad limitada

-0.315

MEDIA:Sociedad anonima

0.028

MEDIA:Sociedad limitada

-0.028

REDUCIDA:Sociedad anonima

-0.342

REDUCIDA:Sociedad limitada

0.342

ALTA:Sociedad anonima

0.146

ALTA:Sociedad limitada

-0.146

NEGATIVA:Sociedad anonima

-0.067

NEGATIVA:Sociedad limitada

0.067

POSITIVA:Sociedad anonima

-0.080

POSITIVA:Sociedad limitada

0.080

GRANDE:ALTA:Sociedad anonima

0.256

GRANDE:ALTA:Sociedad limitada

-0.256

GRANDE:NEGATIVA:Sociedad anonima

-0.084

GRANDE:NEGATIVA:Sociedad limitada

0.084

GRANDE:POSITIVA:Sociedad anonima

-0.173

GRANDE:POSITIVA:Sociedad limitada

0.173

MEDIA:ALTA:Sociedad anonima

-0.057

MEDIA:ALTA:Sociedad limitada

0.057

MEDIA:NEGATIVA:Sociedad anonima

-0.143

MEDIA:NEGATIVA:Sociedad limitada

0.143

MEDIA:POSITIVA:Sociedad anonima

0.201

MEDIA:POSITIVA:Sociedad limitada

-0.201

REDUCIDA:ALTA:Sociedad anonima

-0.199

REDUCIDA:ALTA:Sociedad limitada

0.199

REDUCIDA:NEGATIVA:Sociedad anonima

0.227

REDUCIDA:NEGATIVA:Sociedad limitada

-0.227

REDUCIDA:POSITIVA:Sociedad anonima

-0.028

REDUCIDA:POSITIVA:Sociedad limitada

0.028

La estimación del modelo saturado permite aplicar algún algoritmo para la obtención de la especificación de un modelo que, sin llegar a ser el saturado, **asegure una representación valida de la realidad obviando los efectos e interacciones que no sean estadísticamente relevantes**. Por ejemplo, un método es el *step / backward*, que, en función del *Criterio de Información de Akaike* (AIC), irá probando a estimar especificaciones más simples que disminuyan el AIC (lo que implica una mejor especificación). En nuestro caso, se aplicará con el código:

# Elección del modelo final.  
  
 modelo\_def <- step(modelo\_sat, scale = 0,  
 direction = c("backward"),  
 trace = 1, steps = 1000)

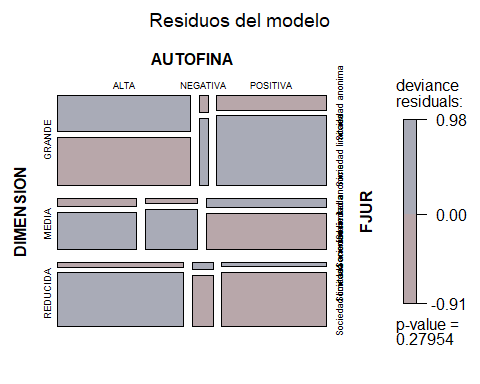
## Start: AIC=36  
## ~DIMENSION \* AUTOFINA \* FJUR  
##   
## Df AIC  
## - DIMENSION:AUTOFINA:FJUR 4 33.077  
## <none> 36.000  
##   
## Step: AIC=33.08  
## ~DIMENSION + AUTOFINA + FJUR + DIMENSION:AUTOFINA + DIMENSION:FJUR +   
## AUTOFINA:FJUR  
##   
## Df AIC  
## <none> 33.077  
## - AUTOFINA:FJUR 2 36.975  
## - DIMENSION:AUTOFINA 4 47.839  
## - DIMENSION:FJUR 2 51.252

En el código anterior, se genera el modelo óptimo siguiendo el proceso de eliminar los términos del modelo que más contribuyan a la reducción del valor de AIC, mediante la función step(). El modelo así obtenido se guarda como “modelo\_def”.

En nuestro ejemplo, la especificación del modelo definitivo incluye los tres efectos directos o principales de los factores o atributos, y todas las interacciones entre pares de factores; es decir, se diferencia del modelo saturado en la no inclusión de la interacción entre los tres factores o atributos de modo simultáneo.

Luego, se pasa el modelo a la función generar\_solución() para obtener los resultados (el gráfico de mosaico y las dos tablas). El gráfico obtenido es:

generar\_solucion(modelo\_def)



Los tonos grisáceos y pardos indican que existen leves diferencias entre las frecuencias conjuntas observadas, y las estimadas por el modelo. En cuanto a la tabla de validación:

solucion\_modelo\_def$Informacion

Table 10.8: Validación del modelo

Prueba

Estadístico

Grados Libertad

P-valor

Likelihood Ratio

5.076509

4

0.2795369

Pearson

5.277486

4

0.2599919

Las dos pruebas muestran un *p-valor* superior a 0,05, lo que implica, para ese nivel de significación, el **no rechazo de la hipótesis nula de validez o idoneidad del modelo** para representar la realidad adecuadamente. En cuanto a los coeficientes estimados del modelo:

solucion\_modelo\_def$Coeficientes

Table 10.8: Coeficientes del modelo

Coefficient

Value

T. Independiente

2.255

DIMENSION:GRANDE

0.303

DIMENSION:MEDIA

0.056

DIMENSION:REDUCIDA

-0.359

AUTOFINA:ALTA

0.711

AUTOFINA:NEGATIVA

-1.183

AUTOFINA:POSITIVA

0.472

FJUR:Sociedad anonima

-0.880

FJUR:Sociedad limitada

0.880

GRANDE:ALTA

0.381

GRANDE:NEGATIVA

-0.644

GRANDE:POSITIVA

0.263

MEDIA:ALTA

-0.551

MEDIA:NEGATIVA

0.693

MEDIA:POSITIVA

-0.142

REDUCIDA:ALTA

0.170

REDUCIDA:NEGATIVA

-0.049

REDUCIDA:POSITIVA

-0.121

GRANDE:Sociedad anonima

0.384

GRANDE:Sociedad limitada

-0.384

MEDIA:Sociedad anonima

0.088

MEDIA:Sociedad limitada

-0.088

REDUCIDA:Sociedad anonima

-0.472

REDUCIDA:Sociedad limitada

0.472

ALTA:Sociedad anonima

0.267

ALTA:Sociedad limitada

-0.267

NEGATIVA:Sociedad anonima

-0.131

NEGATIVA:Sociedad limitada

0.131

POSITIVA:Sociedad anonima

-0.135

POSITIVA:Sociedad limitada

0.135

En la tabla anterior detaca, dentro de los efectos directos o principales, el coeficiente negativo de la capacidad de autofinanciación negativa y el de forma jurídica de *Sociedad Anónima* como principales categorías que influyen en la disminución de las frecuencias conjuntas implicadas. En el extremo opuesto se encuentra el coeficiente correspondiente a la forma jurídica de *Sociedad Limitada*. En cuanto a las interacciones entre factores, se comprueba que la simultaneidad entre una dimensión de la matriz grande y una capacidad de autofinanciación negativa parece influir en una reducción significativa de las frecuencias conjuntas implicadas. Lo mismo ocurre con la simultaneidad entre una dimensión de la empresa matriz media y una capacidad de autofinanciación alta, y entre una dimensión de la empresa matriz reducida y la forma jurídica de *Sociedad Anónima*. En el extremo opuesto, hay ciertas interacciones que parecen inluir en que las frecuencias conjuntas implicadas aumenten, como por ejemplo una dimensión de la matriz reducida y una capacidad de autofinanciación negativa, o de nuevo una dimensión de la matriz empresarial reducida con una forma jurídica de *Sociedad Limitada*.

## 10.5 Materiales para realizar las prácticas del capítulo.

En esta sección se muestran los links de acceso a los diferentes materiales (*scripts*, datos…) necesarios para llevar a cabo los contenidos prácticos del capítulo.

**Datos (en formato Microsoft (R) Excel (R)):**

* eolica\_contingencia.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1N5QIRKbm6OKAOaN8DB2YFDptpZ_0pbPc/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true))
* eolica\_contingencia2.xlsx ([obtener aquí](https://docs.google.com/spreadsheets/d/1f0OQaxyzjKIIXrqhCFL1vIBNcLLGx6R4/edit?usp=sharing&ouid=115375878280465826079&rtpof=true&sd=true))
* **Scripts:**
* correspondencias\_eolica.R [(](https://drive.google.com/file/d/1p4p17OudRprwkpF7wvvF1gwHZG82uv89/view?usp=sharing)[obtener aquí](https://drive.google.com/file/d/1u8gJa3jQH2cwuQ8KsISQ9esHKrFAO_au/view?usp=sharing))
* loglineal\_eolica.R ([obtener aquí](https://drive.google.com/file/d/1xMmM2nrSHBCMU1EEDePb2BRuv_TbD-7r/view?usp=sharing))

Hernández-Alonso, J. 2000. *Economía Cuantitativa*. Síntesis.

Intriligator, R. G.; Hsiao, M. D.; Bodkin. 1996. *Econometric Models, Techniques and Applications*. Prentice-Hall.

Martín-Pliego, F. J. 2004. *Introducción a La Estadística Económica y Empresarial*. Thomson/Paraninfo.

Mood, F. A., A. M.; Graybill. 1963. *Introduction to the Theory of Statistics (2nd Edition)*. Mc Graw-Hill.

Peña, D. 1983. “La Influencia de Las Teorías de Newton y Darwin En El Nacimiento de La Estadística Matemática.” *Estadística Española* 99: 103–4.

R Core Team. 2024. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. <https://www.R-project.org/>.

Samuelson, W. D., P. A.; Nordhaus. 2006. *Microeconomía (18ª Edición)*. Mc Graw Hill.

Wickham, Hadley. 2021. *Ggplot2: Elegant Graphics for Data Analysis, 3 Ed.* Springer-Verlag New York. <https://ggplot2-book.org/>.

Wickham, Hadley, and Garrett Grolemund. 2017. *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data*. 1st ed. Paperback; O’Reilly Media. <http://r4ds.had.co.nz/>.