

Métodos de Data Science aplicados a la Economía y a la Administración y Dirección de Empresas

Juan Antonio Vicente Vírseda

Julio González Arias

Francisco Javier Parra Rodríguez

Mauricio Beltrán Pascual



Métodos de Data Science aplicados a la Economía y a la Dirección y Administración de Empresas

Autores:

Juan Antonio Vicente Vírseda
Julio González Arias
Francisco Javier Parra Rodríguez
Mauricio Beltrán Pascual

**MÉTODOS DE DATA SCIENCE APLICADOS A LA ECONOMÍA
Y A LA DIRECCIÓN Y ADMINISTRACIÓN DE EMPRESAS**

Quedan rigurosamente prohibidas, sin la autorización escrita de los titulares del Copyright, bajo las sanciones establecidas en las leyes, la reproducción total o parcial de esta obra por cualquier medio o procedimiento, comprendidos la reprografía y el tratamiento informático, y la distribución de ejemplares de ella mediante alquiler o préstamos públicos.

© Universidad Nacional de Educación a Distancia
Madrid 2019

www.uned.es/publicaciones

© Juan Antonio Vicente Vírseda, Julio González Arias, Francisco Javier Parra Rodríguez
Mauricio Beltrán Pascual

*Todas nuestras publicaciones han sido sometidas
a un sistema de evaluación antes de ser editadas.*

ISBN electrónico: 978-84-362-7601-5

Edición digital: octubre de 2019

ÍNDICE

Primer Bloque. Contextualización y Consideraciones Iniciales

Tema 1: De la Estadística a Data Science y Big Data

1. Métodos Estadísticos

- 1.1. Definición y clasificación de la Estadística
- 1.2. Conceptos estadísticos fundamentales
- 1.3. La Estadística oficial en España y Europa

2. Data Science y Big Data. La Nueva Realidad

- 2.1. Conceptos clave
- 2.2. Minería de Datos o Data Mining
- 2.3. Modelos SEMMA y CRISP-DM
 - 2.3.1. Modelo SEMMA
 - 2.3.2. Modelo CRISP-DM
 - 2.3.3. Diferencias entre SEMMA y CRISP-DM
- 2.4. Principales métodos y algoritmos en la Minería de Datos

3. Big Data

- 3.1. Desafíos
 - 3.1.1. Open Data
 - 3.1.2. Small Data y Smart Data
 - 3.1.3. No es oro todo lo que reluce
 - 3.1.4. Consideraciones legales básicas
- 3.2. Aplicaciones
- 3.3. Principales herramientas
 - 3.3.1. Hadoop y MapReduce
 - 3.3.2. Spark

4. Programas de Software más Utilizados

- 4.1. R y RStudio
 - 4.1.1. Consola R-Studio
 - 4.1.2. R Markdown
 - 4.1.3. Librerías en el programa estadístico R
- 4.2. Weka
- 4.3. Introducción al lenguaje de programación Python
- 4.4. El programa IBM SPSS Statistics: conexión con R y Python
- 4.5. Otros softwares: Julia y Scala
 - 4.5.1. Julia
 - 4.5.2. Scala

Tema 2: Introducción al Lenguaje R

1. Introducción a R y Ayuda en Línea

2. Objetos en R

- 2.1. Vectores
 - 2.1.1. Crear
 - 2.1.2. Seleccionar elementos
 - 2.1.3. Trabajar con vectores

- 2.2. Matrices
 - 2.2.1. Crear
 - 2.2.2. Seleccionar elementos
 - 2.2.3. Trabajar con matrices
- 2.3. Listas
 - 2.3.1. Crear
 - 2.3.2. Seleccionar elementos
 - 2.3.3. Manipular elementos
 - 2.3.4. Unir listas
 - 2.3.5. Convertir lista en vector
- 2.4. Factores
- 2.5. Dataframes
 - 2.5.1. Crear
 - 2.5.2. Seleccionar elementos
 - 2.5.3. Manipular dataframes
 - 2.5.4. Importar/Exportar datos externos

3. Realizar Consultas

- 3.1. Condiciones simples
- 3.2. Condiciones múltiples
- 3.3. Filtrar dataframes

4. Funciones

- 4.1. Definir una función
- 4.2. Funciones incorporadas
- 4.3. Funciones definidas por el usuario

5. Estructuras de Control

- 5.1. Sentencias condicionales
 - 5.1.1. Sentencia if
 - 5.1.2. Sentencia if...else
 - 5.1.3. Sentencias anidadas
- 5.2. Bucles
 - 5.2.1. Bucle For
 - 5.2.2. Bucle While

6. La Familia de Funciones Apply

- 6.1. Las funciones apply vs. bucles

7. Gráficos

Segundo Bloque. Métodos Estadísticos Multivariantes

Tema 3: Modelo Lineal General y Modelo Lineal Generalizado

1. Modelo Lineal General

- 1.1. Modelo de Regresión Lineal
 - 1.1.1. Introducción
 - 1.1.2. Modelo de Regresión Lineal Simple
 - 1.1.3. Modelo de Regresión Lineal Múltiple
 - 1.1.4. Propiedades estadísticas del estimador MCO
 - 1.1.5. Coeficiente de determinación
 - 1.1.6. Inferencia acerca de los estimadores

- 1.1.7. Predicción
- 1.1.8. Estimación del modelo de regresión con R
- 1.2. Extensiones al Modelo de Regresión Lineal
 - 1.2.1. Introducción
 - 1.2.2. Heterocedasticidad
 - 1.2.3. Autocorrelación
 - 1.2.4. Deficiencias muestrales
 - 1.2.5. Errores de especificación
 - 1.2.6. Métodos de selección de variables en el modelo lineal general
- 1.3. Modelos con variables cualitativas explicativas
 - 1.3.1. Introducción
 - 1.3.2. Modelos ANOVA: efectos fijos
 - 1.3.3. Modelos de componentes de la varianza: efectos aleatorios
 - 1.3.4. Modelos anidados o jerárquicos
 - 1.3.5. Modelos ANCOVA
- 1.4. Modelos con variable dependiente multivariante: MANOVA y MANCOVA
 - 1.4.1. Definición del contraste
 - 1.4.2. Supuestos para su aplicación
 - 1.4.3. Estadísticos
 - 1.4.4. Interpretación del test
 - 1.4.5. Cálculo en R
- 1.5. Estimación por Máxima Verosimilitud Restringida (REML) en modelos mixtos
- 1.6. Ajuste de modelos mixtos con R
 - 1.6.1. Función lme() del paquete nlme
 - 1.6.2. Función lmer() del paquete lme4
 - 1.6.3. Ejemplos de modelos con R

2. Modelo Lineal Generalizado

- 2.1. Formulación general
- 2.2. Modelos con variables cualitativas endógenas
 - 2.2.1. Modelo probabilístico lineal
 - 2.2.2. Modelo Logit
 - 2.2.3. Modelo Probit
 - 2.2.4. Modelo Logit vs Modelo Probit
- 2.3. Modelo Tobit

3. Evaluación de Modelos

- 3.1. Devianza. Estadístico G₂ de Wilks de razón de verosimilitudes
- 3.2. Estadístico χ^2 de Pearson
- 3.3. Criterio de Información de Akaike (AIC) y Criterio de Información Bayesiano (BIC)
- 3.4. Prueba de Hosmer-Lemeshaw
- 3.5. Medidas tipo R₂
 - 3.5.1. Pseudo R² de McFadden (McFadden, 1974)
 - 3.5.2. Pseudo R² de Cox-Snell (Cox & Snell, 1989)
 - 3.5.3. Pseudo R² de Nagelkerke (Nagelkerke, 1991)
- 3.6. Métodos específicos para modelos de clasificación
 - 3.6.1. Métodos basados en métricas
 - 3.6.2. Métodos basados en la curva ROC
 - 3.6.3. Métodos basados en una matriz de costes

Tema 4: Métodos Estadísticos de Reducción de Dimensiones

1. Análisis Factorial y Componentes Principales

- 1.1. Introducción
- 1.2. Análisis Factorial vs Componentes Principales
- 1.3. Análisis de Componentes Principales
- 1.4. Análisis Factorial
 - 1.4.1. Planteamiento
 - 1.4.2. Hipótesis en el Modelo Factorial
 - 1.4.3. Comunalidad y especificidad (unicidad)
 - 1.4.4. Diseño del análisis
 - 1.4.5. Extracción de los factores
 - 1.4.6. La matriz factorial o de componentes
 - 1.4.7. Autovalores o valores propios
 - 1.4.8. Número de factores a conservar
 - 1.4.9. Rotación de los factores
 - 1.4.10. Puntuaciones factoriales
 - 1.4.11. Interpretación de los factores
 - 1.4.12. Casos de Heywood y otras anomalías sobre estimaciones de comunalidad
 - 1.4.13. Ejemplos con R

2. Análisis de Correspondencias

- 2.1. Introducción
- 2.2. Objetivo
- 2.3. Análisis de Correspondencias Simple
 - 2.3.1. Planteamiento
 - 2.3.2. Definición de perfiles
 - 2.3.3. Medida de distancia utilizada
 - 2.3.4. Extracción de las dimensiones o espacios factoriales
 - 2.3.5. Método de normalización
 - 2.3.6. Interpretación de resultados
- 2.4. Análisis de Correspondencias Múltiple
 - 2.4.1. Planteamiento
 - 2.4.2. Nube de puntos, perfiles
 - 2.4.3. Inercia
 - 2.4.4. Solución del Análisis de Correspondencias
 - 2.4.5. Interpretación de los resultados
- 2.5. Ejemplo de Análisis de Correspondencias Simple con el software R
 - 2.5.1. Análisis exploratorio
 - 2.5.2. Estimación del modelo
 - 2.5.3. Valores propios
 - 2.5.4. Biplot simétrico
 - 2.5.5. Análisis de perfiles fila
 - 2.5.6. Análisis de perfiles columna
 - 2.5.7. Biplots asimétricos
 - 2.5.8. Biplot de contribución
 - 2.5.9. Descripción de la dimensión
- 2.6. Ejemplo de Análisis de Correspondencias Múltiple en R
 - 2.6.1. Análisis exploratorio
 - 2.6.2. Estimación del modelo
 - 2.6.3. Valores propios
 - 2.6.4. Biplot simétrico

- 2.6.5. Análisis de las variables
- 2.6.6. Análisis de los individuos
- 2.6.7. Coloreando individuos por grupos
- 2.6.8. Descripción de la dimensión
- 2.6.9. Individuos y variables suplementarias
- 2.6.10. Filtrado de resultados

Tema 5: Medidas de Distancias y Agrupamiento

1. Medidas de distancia/proximidad

- 1.1. Medidas de distancia o disimilaridad
 - 1.1.1. Escala de intervalo
 - 1.1.2. Frecuencias
 - 1.1.3. Datos binarios
- 1.2. Medidas de proximidad o similaridad
 - 1.2.1. Escala de intervalo
 - 1.2.2. Datos binarios
- 1.3. Distancia de Mahalanobis
 - 1.3.1. Distancia euclídea normalizada
 - 1.3.2. Definición y propiedades de la distancia de Mahalanobis
 - 1.3.3. Distancias singulares

2. Agrupamiento de la Información

- 2.1. Análisis Discriminante
 - 2.1.1. Clasificación con dos grupos
 - 2.1.2. Clasificación con más de dos grupos
 - 2.1.3. Ejemplos con el software R
- 2.2. Análisis Clúster
 - 2.2.1. Introducción
 - 2.2.2. Etapas a seguir en el desarrollo del Análisis Clúster
 - 2.2.3. Modelos jerárquicos
 - 2.2.4. Modelos no jerárquicos
- 2.3. Escalamiento Multidimensional
 - 2.3.1. Modelo general o método clásico
 - 2.3.2. Otros modelos de escalamiento
 - 2.3.3. Relación con otras técnicas multivariantes
- 2.4. Análisis de Correlación Canónica
 - 2.4.1. Introducción
 - 2.4.2. Modelo
 - 2.4.3. Interpretación de resultados
 - 2.4.4. Ejemplo en R

Tercer Bloque. Introducción al Machine Learning

Tema 6: Regresión y Clasificación: Árboles de Decisión y Redes Neuronales

1. Uso de Muestras para el Entrenamiento, Validación y Test

- 1.1. Muestras de entrenamiento, validación y test
- 1.2. Validación cruzada

2. Árboles de Decisión y Clasificación

- 2.1. Introducción
- 2.2. Aplicabilidad de los árboles de decisión para clasificación
- 2.3. Características de los algoritmos de clasificación
 - 2.3.1. Particiones posibles y criterios de selección
 - 2.3.2. Ganancia de información
 - 2.3.3. El criterio de proporción de ganancia
 - 2.3.4. Índice de diversidad de Gini
 - 2.3.5. Otros criterios de selección
 - 2.3.6. Poda en Árboles de clasificación
- 2.4. Árbol CHAID (Chi-square Automatic Interaction Detection) y CHAID exhaustivo
- 2.5. Árbol CRT (Classification and Regression Trees)
- 2.6. Árbol QUEST (Quick, Unbiased, Efficient Statistical Tree)
- 2.7. Árbol C5.0
- 2.8. Otros algoritmos de clasificación
 - 2.8.1. Algoritmo de construcción de árboles consolidados
 - 2.8.2. Random Forest
 - 2.8.3. Decision Stump
- 2.9. Árboles de decisión con R
 - 2.9.1. Conditional Inference Tree
 - 2.9.2. Recursive Partitioning and Regression Trees
 - 2.9.3. CHAID (CHi-square Automatic Interaction Detection)
 - 2.9.4. Árbol C5.0 de Quinlan
 - 2.9.5. Random Forest
 - 2.9.6. Árboles con caret (ejemplo de Random Forest)
 - 2.9.7. Árboles con Rweka

3. Redes Neuronales Artificiales

- 3.1. Introducción
- 3.2. Tipos de modelos de redes neuronales
- 3.3. Unidades de procesamiento de la información
- 3.4. Propiedades de los sistemas neuronales
- 3.5. Perceptrón multicapa
 - 3.5.1. Etapa de funcionamiento
 - 3.5.2. Etapa de aprendizaje
 - 3.5.3. Metodología de aplicación de un perceptrón multicapa
 - 3.5.4. Evaluación del rendimiento del modelo
- 3.6. Funciones de base radial
- 3.7. Comparación entre las Funciones de Base Radial y el Perceptrón Multicapa
- 3.8. Análisis de sensibilidad e interpretación de los pesos de la red
 - 3.8.1. Análisis basado en la magnitud de los pesos de la red
 - 3.8.2. Análisis de sensibilidad
- 3.9. Redes neuronales y modelos estadísticos clásicos
- 3.10. Otras arquitecturas de redes neuronales
- 3.11. Librería R Weka con redes neuronales
 - 3.11.1. Análisis con base de datos German Credit
 - 3.11.2. Análisis con base de datos Boston Housing

Tema 7: Exploración y Preprocesado de los Datos

1. Introducción: Fases Metodológicas de un Proceso de Data Science

2. Imputación de Datos Ausentes

3. Filtrado y Eliminación de Valores Extremos u Outlier

4. Transformación de la Base de Datos

4.1. Discretización de variables

5. Balanceo de las Clases

6. Reducción de Variables o de la Dimensionalidad

6.1. Aproximación indirecta o filter

6.2. Aproximación directa o wrapper (envoltura)

6.3. Selección de variables con la librería caret de R

6.4. Selección de variables con el programa WEKA

Bibliografía

Aquí podrá encontrar información adicional y actualizada de esta publicación

PRIMER BLOQUE. CONTEXTUALIZACIÓN Y CONSIDERACIONES INICIALES

TEMA 1: DE LA ESTADÍSTICA A DATA SCIENCE Y BIG DATA

ÍNDICE

1. Métodos Estadísticos

- 1.1. Definición y clasificación de la Estadística
- 1.2. Conceptos estadísticos fundamentales
- 1.3. La Estadística oficial en España y Europa

2. Data Science y Big Data. La Nueva Realidad

- 2.1. Conceptos clave
- 2.2. Minería de Datos o Data Mining
- 2.3. Modelos SEMMA y CRISP-DM
- 2.4. Principales métodos y algoritmos en la Minería de Datos

3. Big Data

- 3.1. Desafíos
- 3.2. Aplicaciones
- 3.3. Principales herramientas

4. Programas de Software más Utilizados

- 4.1. R y RStudio
- 4.2. Weka
- 4.3. Introducción al lenguaje de programación Python
- 4.4. El programa IBM SPSS Statistics: conexión con R y Python
- 4.5. Otros softwares: Julia y Scala

OBJETIVOS	PALABRAS CLAVE
<ul style="list-style-type: none">· Introducción a los conceptos básicos sobre la estadística· Introducción a los conceptos básicos sobre Big Data y Data Science· Presentar las herramientas más importantes en el ámbito del Data Science y Big Data	<ul style="list-style-type: none">· Estadística· Data Mining· Big Data· Data Science

1. MÉTODOS ESTADÍSTICOS

1.1. DEFINICIÓN Y CLASIFICACIÓN DE LA ESTADÍSTICA

La Estadística suele definirse como la ciencia que tiene por objeto recoger de forma agrupada la información que se produce de fenómenos repetitivos o no ocasionales. En su origen, la Estadística se desarrolló en la Economía y en la Política, de hecho hasta bien entrado el siglo XIX la "Estadística" se utilizaba para hacer mención a informaciones de tipo socioeconómico sobre la realidad de un Estado (establecimiento de registros de población, nacimientos, defunciones, etc.; censos de edificios y de elementos de riqueza, etc.); etimológicamente la palabra alude precisamente a esta acepción de "Ciencia de los Estados".

Hoy, la Estadística no queda reservada al estudio del Estado. Su gran desarrollo durante el siglo XX la ha configurado como base fundamental para la investigación en múltiples ciencias y áreas del conocimiento humano.

Actualmente, en su definición más común se le reconoce como "*la ciencia que se ocupa de la obtención de información y que proporciona instrumentos para la toma de decisiones cuando prevalecen condiciones de incertidumbre*" o "*la rama del método científico que se ocupa de los datos obtenidos contando o midiendo las propiedades de determinados colectivos*"; a estos colectivos, se les denomina en Estadística "Poblaciones".

El concepto de incertidumbre o de falta de certeza proviene de la existencia en la naturaleza de fenómenos aleatorios; en este sentido se diferencia entre fenómenos deterministas y fenómenos aleatorios; un fenómeno es determinista cuando al experimentarlo en las mismas condiciones se obtienen siempre los mismos resultados (las leyes físicas y químicas clásicas suelen tener este tipo de comportamientos, al dejar caer un objeto desde un árbol siempre cae al suelo a una determinada velocidad derivada de la ley de la gravedad, la unión de 2 moléculas de hidrógeno y una de oxígeno en determinadas condiciones siempre proporciona agua, etc..); un fenómeno es, por el contrario, aleatorio cuando al experimentarlo en las mismas condiciones produce un resultado variable que no puede predecirse a priori con exactitud (el tiempo de vida de una lámpara, el lanzamiento de una moneda, el resultado de un partido de fútbol o de un examen, etc.).

Los fenómenos aleatorios son los más comunes en la naturaleza y en el comportamiento humano y social; bien por su propia naturaleza, bien por su complejidad, bien por la falta de información o de modelos explicativos adecuados, tanto en la vida cotidiana como en la toma de decisiones empresariales nos enfrentamos a factores de "aleatoriedad"; el resultado de una contienda electoral, la cuantía de una subvención pedida a la administración, la adjudicación de un concurso público, la duración del viaje en coche al lugar de vacaciones, el tiempo que nos hará el fin de semana, el número de visitantes que tendremos en una exposición, etc., son fenómenos aleatorios, a cuyo análisis y comprensión ayudan las técnicas estadísticas.

La Estadística suele dividirse en dos grandes apartados: la **Estadística Descriptiva**, que recoge un conjunto de técnicas y procedimientos para organizar, resumir y tratar sistemáticamente datos disponibles de sucesos ya acaecidos y la **Estadística Inferencial** o **Inferencia Estadística** que, basada en la teoría matemática de la Probabilidad, estudia los métodos empleados para inferir algo acerca de una población basándose en la información aportada por una parte del colectivo (muestra).

Dado el contexto en el que se desarrolla, donde el coste fundamental es el destinado a la obtención de los datos, y las limitaciones existentes a nivel de proceso, la inferencia estadística se orienta fundamentalmente a la eficiencia, es decir, a obtener las mejores conclusiones posibles con el menor número de datos. Para ello, se opta por modelos simples (fundamentalmente lineales) y se hacen supuestos simplificadores, que abarcan desde la distribución a priori de las frecuencias relativas de las variables, a la ausencia de correlación entre variables independientes en los modelos, aleatoriedad, etc.

1.2. CONCEPTOS ESTADÍSTICOS FUNDAMENTALES

Los principales conceptos básicos que se utilizan en Estadística son los siguientes:

Población

Se denomina población al conjunto de elementos que cumplen ciertas propiedades y entre los que se desea estudiar el fenómeno en cuestión; este conjunto de elementos es de distinta naturaleza: personas, hogares, empresas, edificios, tornillos, caras de una moneda o de un dado, cartas de una baraja, etc.

Si estudiamos las caras de una moneda, la población estará compuesta por dos elementos (cara y cruz), si consideramos un dado, por 6 elementos, etc.

Muestra

Muestra es cualquier subconjunto de individuos pertenecientes a una población determinada.

El interés de las muestras es que, en determinadas condiciones, las técnicas estadísticas permiten que los resultados que se obtengan del análisis de una muestra puedan ser extendidos (“inferidos”) al conjunto de la población a la que pertenece; estas técnicas nos evitan el enorme trabajo que puede suponer el estudio de toda una población de grandes dimensiones.

Para ello, las muestras deben ser representativas de la población, tener un tamaño suficientemente grande y cumplir una serie de condiciones, fundamentalmente hipótesis sobre su distribución de frecuencias.

Individuo

Individuo o Unidad de Investigación es cada uno de los elementos de una muestra o de una población.

Un ejemplo de población serían el conjunto de visitantes al Museo del Prado en el mes de junio del año 2017. Cada uno de estos visitantes es un individuo o un elemento de dicha población; una muestra elegida aleatoriamente de esta población podría ser la selección de los 10 primeros visitantes que entraron cada día a partir de las 11 y de las 13 horas de la mañana (horas, ambas, elegidas al azar).

Si tenemos 30 días de apertura del museo y realizamos una encuesta a 20 individuos cada día, tendremos una muestra de 600 individuos elegidos al azar; esta muestra es suficientemente grande para extraer determinadas conclusiones estadísticas generales, de forma que, si en la muestra seleccionada un 30% de los individuos son extranjeros y el 70% nacionales, podemos inferir ambas proporciones al colectivo y afirmar que el 30% de los visitantes al Museo del Prado durante el mes de referencia eran extranjeros, evitando con

ello el arduo trabajo de tener que entrevistar a todos y cada uno de los visitantes para conocer su nacionalidad.

La teoría de la inferencia estadística nos permitirá extraer estas conclusiones con un determinado error y un cierto nivel de confianza o probabilidad de equivocarnos; de esta forma será habitual indicar que el dato anterior se ha obtenido, por ejemplo, con un 5% de error y con un nivel de confianza del 95%; estas afirmaciones quieren decir que la teoría estadística asegura que si hiciésemos el experimento (repitiésemos la encuesta) 100 veces en idénticas condiciones, 95 veces obtendríamos un resultado que estaría comprendido entre el 28,5 (un 5% menos del 30% obtenido) y el 31,5% (un 5% más del 30% obtenido); las otras 5 veces puede darnos cualquier cosa diferente.

Hasta ahí y solo hasta ahí llega la inferencia estadística, lo que no es poco como ayuda en la toma de decisiones empresariales; saber, con un 95% de certeza, el país de procedencia de los visitantes a una ciudad, el grado de satisfacción de los clientes con los servicios de una empresa, el motivo por el que se elige un determinado producto, etc., puede ser fundamental para planificar una acción de marketing o para tomar decisiones de cambio en la organización de una determinada empresa.

Las investigaciones estadísticas pueden ser censales y muestrales; baste recordar aquí, que según los casos, determinadas investigaciones solo pueden hacerse con carácter muestral (solo puede analizarse una muestra de sangre de un individuo o una muestra de agua del mar, ya que una investigación censal sería en ambos casos imposible), mientras que en otros, son necesarias investigaciones censales (un catastro, que se haga con el fin de grabar los edificios con un impuesto de bienes inmuebles, debe tener carácter censal porque debe disponerse de información detallada de todos y cada uno de los individuos a fin de fijar la cuota impositiva que le corresponda).

Parámetros

Son las características poblacionales que deseamos investigar y que suelen ser desconocidas a priori. Por ejemplo, la edad de los viajeros de una compañía aérea, el precio medio en el mercado de un determinado producto, la opinión de los clientes sobre los productos fabricados por una empresa, etc.

Variables

Cuando estas características o parámetros son numéricas, es decir, cuando se pueden medir, se denominan Variables (años de edad, renta anual en euros, etc.).

Las variables pueden ser discretas o continuas, según tomen un número infinito no numerable (variables continuas) o un número finito o infinito numerable de valores (discretas) en un intervalo.

Ejemplos de variables discretas son el número de hijos de una familia, el número de coches de un país, el número de turistas que visitan una ciudad, etc., ejemplos de variables continuas son la temperatura, la edad, el peso o la altura de las personas, la medición de la distancia entre dos puntos, etc.

La mayor parte de las variables continuas pueden tratarse como discretas; así, por ejemplo, si valoramos la edad de las personas en años, despreciando las unidades de tiempo menores (días, minutos, segundos, milésimas de segundo, etc.), una variable continua se convierte en

discreta; lo mismo podríamos pensar para el peso en kilogramos o para la distancia entre dos puntos medida en kilómetros, etc.

También, y como veremos más adelante, aunque se pierda información sobre la distribución, es bastante habitual agrupar los valores de la variable en intervalos de amplitud constante o variable.

Otra característica de las variables es su referencia o no a un determinado período de tiempo.

Tendremos en este sentido dos tipos de datos:

- Variables temporales o históricas; son las referidas a distintos momentos del tiempo y adoptan en general la forma de serie; por ejemplo, la serie mensual de parados inscritos en el INEM.
- Variables atemporales, también llamadas de corte transversal; están referidas a un momento o período concreto y más o menos largo; por ejemplo, las personas que visitaron Toledo el mes de febrero de 2010 o la cifra de negocio de una empresa durante el año 2017.

Atributos o factores

Cuando los parámetros o características de la población no son susceptibles de medirse numéricamente reciben el nombre de Atributos (el color del pelo, el sexo, la profesión, el estado civil, el grado de satisfacción del cliente con un servicio, etc.).

Los atributos, a diferencia de las variables, presentan Modalidades o Categorías (el atributo sexo puede adoptar las modalidades de varón o mujer, el atributo intención de voto se concreta en los nombres de los partidos políticos que se presenten a las elecciones sondeadas o el de opinión sobre la satisfacción de un cliente con el servicio recibido en una gestión bancaria, puede adoptar las modalidades que se determinen: excelente, bueno, malo, regular, etc.).

Los atributos pueden clasificarse como ordenables, que son los que sugieren una ordenación, por ejemplo, el grado de satisfacción con el trato recibido (excelente, bueno, regular, malo), el nivel de estudios (alto, medio, bajo), y no ordenables, que son los que solo admiten una ordenación alfabética o casual, por ejemplo, el estado civil, el color del pelo, el país de procedencia o la nacionalidad de un turista, etc.

El atributo más simple es el que solo presenta dos modalidades: presencia/ausencia; favorable/desfavorable, etc.

1.3. LA ESTADÍSTICA OFICIAL EN ESPAÑA Y EUROPA

La estadística oficial en España comienza su andadura con la creación de la Comisión de Estadística del Reino. El 3 de noviembre de 1856, el general Narváez, presidente del Consejo de Ministros de Isabel II, firma un Decreto por el que se crea una Comisión, compuesta por personas de reconocida capacidad, para la formación de la Estadística General del Reino. Unos meses más tarde, el 21 de abril de 1857, la Comisión pasa a denominarse Junta de Estadística. Su primer trabajo es el Censo de Población, con fecha de referencia del 21 de mayo del mismo año.

La Ley de Instrucción Pública de 9 de septiembre de 1857 establece que la Estadística será una disciplina académica, pasando a impartirse en la Universidad. Un Decreto del 12 de septiembre de 1870, durante el gobierno provisional del general Serrano, crea el Instituto Geográfico. Tres años más tarde, 19 de junio de 1873, pasa a denominarse Instituto Geográfico y Estadístico, asumiendo todas las tareas de recogida de información numérica para el Estado.

En 1877, el Instituto Geográfico y Estadístico aprueba su Reglamento. Las estadísticas pasan a depender del Ministerio de Fomento en el año 1890. Un Decreto de 1 de octubre de 1901 establece la formación de las estadísticas oficiales y la publicación de las mismas. El Instituto Geográfico y Estadístico se transforma en Dirección General y se crean departamentos en los Ministerios para completar su labor.

En 1924, el Consejo del Servicio Estadístico, creado en 1921 es reformado, cuatro años antes de que pase a depender del Ministerio de Trabajo y Previsión. En 1931, la adscripción se hace al Ministerio de la Presidencia.

Durante la Guerra Civil (1936-1939) comienza a funcionar el Servicio Sindical de Estadística en coordinación con los Servicios de Estadística del Estado, dentro de la llamada zona nacional.

La Ley de 31 de diciembre de 1945, publicada en el BOE del 3 de enero de 1946, crea el Instituto Nacional de Estadística (INE, www.ine.es), que tiene como misión la elaboración y perfeccionamiento de las estadísticas demográficas, económicas y sociales ya existentes, la creación de otras nuevas y la coordinación con los servicios estadísticos de las áreas provinciales y municipales.

Además de regular la coordinación entre otros servicios estadísticos como el Servicio Sindical de Estadística, la Ley crea el Consejo Superior de Estadística. El INE se organiza en Servicios Centrales, Delegaciones provinciales y Delegaciones en los Ministerios. El 9 de mayo de 1989 se promulga la Ley de la Función Estadística Pública que hace del INE un organismo autónomo potenciando las nuevas tecnologías estadísticas, la coordinación con las Comunidades Autónomas, la elaboración del Plan Estadístico Nacional y las relaciones con la Unión Europea en materia estadística.

El actual Estatuto del INE, aprobado por Real Decreto 508/2001 de 11 de mayo (BOE 12-05-2001), le asigna las funciones de coordinación general de los servicios estadísticos de la Administración General del Estado, la vigilancia, control y supervisión de las competencias de carácter técnico de los servicios estadísticos estatales, y las demás previstas en la Ley 12/1989, de 9 de mayo, de la Función Estadística Pública.

Con la creación del Estado de las autonomías, a raíz de la Constitución de 1978, los gobiernos regionales han ido asumiendo de forma paulatina la competencia exclusiva en materia estadística para fines no estatales y dentro de sus ámbitos geográficos, creando distintos organismos de estadística regionales; destacan el Instituto Vasco de Estadística (EUROSTAT), el Instituto de Estadística de Cataluña (IDESCAT) y el Instituto Gallego de Estadística (IGE), Instituto de Estadística de la Comunidad de Madrid, etc.

Por su parte los Ayuntamientos son los encargados de la creación, mantenimiento, revisión y custodia del Padrón Municipal de Habitantes, correspondiendo en todo caso al INE la coordinación de todos los municipios y la realización de las comprobaciones oportunas. La resolución de las posibles discrepancias entre los Ayuntamientos con el INE, corresponde al Consejo de Empadronamiento, órgano colegiado con representantes de la Administración

General del Estado (INE, Oficina del Censo Electoral y Ministerio de AAPP). Asimismo, cabe destacar la importante labor estadística que comienzan a realizar los ayuntamientos correspondientes a los grandes municipios, como Madrid, Barcelona o Sevilla, como consecuencia de la creciente demanda de información estadística de ámbito municipal e inferior.

Dada la abundante producción estadística que realizan las oficinas públicas, resulta imprescindible una armonización metodológica de las estadísticas que permita su comparabilidad; a ello han contribuido decisivamente organismos internacionales como la Oficina de Estadísticas Europea (EUROSTAT), el Banco Central Europeo, la Organización Mundial del Turismo (OMT) o la Organización para la Cooperación y el Desarrollo Económico (OCDE).

El EUROSTAT, con sede en Luxemburgo, es la oficina estadística de la Comisión Europea, que produce datos sobre la Unión Europea y promueve la armonización de los métodos estadísticos de los estados miembros. Se creó en 1953 para satisfacer las demandas de la Comunidad del Carbón y el Acero. A lo largo de los años su tarea se ha ampliado; cuando se fundó la Comunidad Económica Europea en 1958 se convirtió en una Dirección General (DG) de la Comisión Europea. Dos de sus papeles particularmente importantes son:

- La producción de datos macroeconómicos que apoyan las decisiones del Banco Central Europeo en su política monetaria para el euro.
- La producción y organización de datos regionales y su clasificación por zonas (NUTS) para orientar las políticas estructurales de la Unión Europea.

Puede decirse que el principal papel de EUROSTAT es proporcionar estadísticas al resto de Direcciones Generales de la Comisión Europea y otras instituciones Europeas para que puedan definir, analizar y mejorar las políticas comunitarias; también se ocupa de desarrollar sistemas estadísticos en los países candidatos a entrar en la Unión.

EUROSTAT no genera datos. Son las autoridades estadísticas de los Estados Miembros las que generan, verifican y analizan los datos nacionales y los envían a EUROSTAT, cuyo papel es consolidar esta información y asegurarse de que se utiliza una metodología homogénea que asegura su comparabilidad.

2. DATA SCIENCE Y BIG DATA. LA NUEVA REALIDAD

La evolución de las tecnologías de la información y de la comunicación llevada a cabo en estos últimos años ha supuesto una auténtica revolución tecnológica. Las sociedades modernas, en general, se encuentran en un nuevo contexto que les brinda la oportunidad de disponer de instrumentos novedosos que incrementan la posibilidad de relacionarse y, al mismo tiempo, las empresas tienen a su alcance herramientas capaces de analizar las nuevas aplicaciones disponibles en internet para optimizar el proceso de toma de decisiones.

A modo de ejemplo, existen bases de datos en la actualidad que contienen microdatos de censo de población, registros médicos, impuestos, a los que podríamos incorporar datos sobre la realización de transacciones financieras en línea o, mediante dispositivos móviles, información geográfica que incluya las coordenadas GPS, etc., es decir, la incorporación de

todas aquellas actividades que de manera habitual realizamos con nuestros dispositivos informáticos, y que generan, en la actualidad, cantidades ingentes de información.

Además de la información directa generada por los seres humanos, la tecnología actual origina el denominado **Internet de las Cosas (Internet of Things -IoT-)**, con un volumen importante en la creación de grandes cantidades de datos. Los datos generados por máquinas provienen principalmente de sensores que recopilan información sobre el clima, señales GPS, imágenes procedentes de satélites, registros de transacciones de compra, imágenes digitales y vídeos, información que deriva de la tecnología que apliquemos. Los dispositivos y sensores instalados en la industria de la automoción, transporte, comercio, servicios, por ejemplo, se espera que crezcan a una tasa anual de entorno al 30% (Bankinter, 2011).

A los recursos humanos y productivos que han formado parte tradicionalmente de los activos relevantes de nuestra economía y sociedad², debemos incorporar, por tanto, un nuevo activo fundamental en las economías del siglo XXI: **los datos**. La aparición de nuevas tecnologías que llevan asociadas la generación de una ingente cantidad de información como sistemas estadísticos o geográficos, datos como los meteorológicos, geolocalización, transporte, salud, consumo de energía o simplemente datos de investigación, genera la necesidad de racionalizar este conjunto masivo de datos, lo que implica, en primer lugar, el desarrollo de nuevas habilidades y, en segundo lugar, de nuevas herramientas.

El **Big Data**, también denominado grandes datos, datos masivos o macrodatos, se refiere a un gran conjunto de datos o información generados por un gran número de diversas fuentes y de forma muy rápida. Esta generación de datos puede provenir directamente de las personas o, por el contrario y de manera indirecta, generados por máquinas. Abarca, por lo tanto, a todos los sectores de la actividad económica, desde el transporte o la energía hasta el cuidado de la salud.

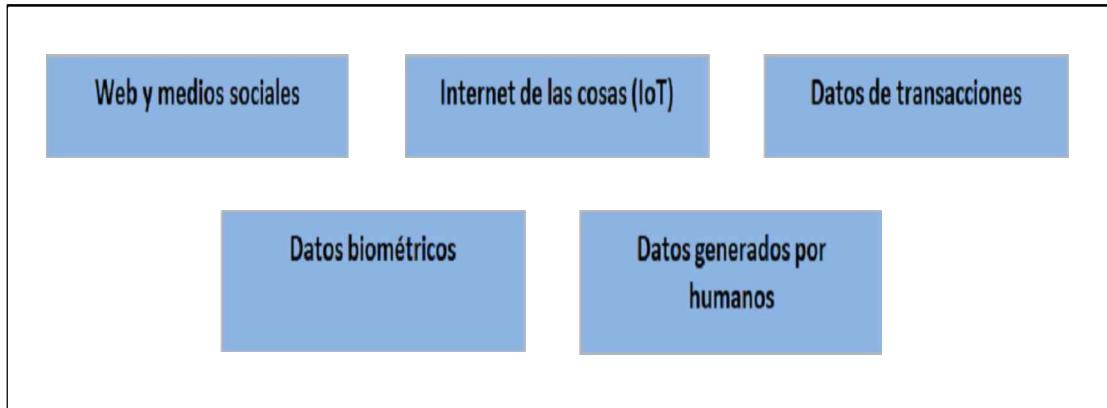
En cuanto a los diferentes tipos de datos a examinar, no debemos olvidar que la primera pregunta que debemos plantearnos es acerca de cuál es el problema que pretendemos solucionar en nuestra empresa o institución. Partiendo de la hipótesis de trabajo y de la heterogeneidad de los tipos de datos de los que disponemos y que pretendemos analizar, se pueden clasificar los tipos datos de *big data* de la siguiente manera:

1. *Datos procedentes de la web y de las redes sociales*: incluye contenidos procedentes de páginas web, así como información obtenida de redes sociales (Twitter, Facebook, LinkedIn), blogs, entre otros.
2. *Internet de las Cosas*: en el ámbito de aquellas tecnologías que nos posibilitan la conexión mediante internet de objetos cotidianos. Este concepto, incluye la tecnología máquina a máquina que se refiere a la transmisión de datos entre dispositivos a través de redes (alámbricas, inalámbricas -wireless-, o híbridas).
3. *Datos procedentes de transacciones*: entre los que se incluyen registros de facturación o registros de llamadas. Este tipo de datos es susceptible de adoptar formatos de datos estructurados, no estructurados e híbridos.

¹ La instalación de sensores en los contenedores nos proporcionan su geolocalización en tiempo real, siendo esta información remitida a las compañías de transporte. En el ámbito del sector energético, los sensores posibilitan determinar el consumo de energía y, por lo tanto, la demanda real de nuestros clientes.

² <https://ec.europa.eu/digital-single-market>

4. *Datos biométricos*: como pueden ser huellas digitales, escaneo de la retina, reconocimiento facial, genética, etc.
5. *Datos generados por los humanos*: entre los que se encuentran los correos electrónicos, mensajes, documentos electrónicos, análisis médicos, notas de voz o llamadas telefónicas.

FIGURA 1. TIPOS DE DATOS DE *BIG DATA*

Fuente: Soares (2013)

En resumen, y según se desprende de la figura anterior, la naturaleza y características de la información generada en estos últimos años es sustancialmente diferente a la información producida en el pasado. Este hecho se debe a la existencia de nuevos dispositivos y tecnologías que forman parte de nuestra vida cotidiana como cámaras, micrófonos, sensores, escáneres médicos, GPS, entre otros. Los datos generados a partir de estos dispositivos conformarán en el futuro la principal fuente de información disponible.

Si los datos ya están en el mercado, la siguiente fase o etapa será la creación de valor en la "industria de los datos" como marco base de la futura economía del conocimiento. Por consiguiente, se trata de incorporar el uso eficiente de los datos a la totalidad de la actividad económica incluyendo desde los sectores más novedosos relacionados con las nuevas tecnologías, hasta los sectores más tradicionales como la manufactura, el transporte o la salud, para que, en última instancia, se generen nuevas oportunidades de negocio.

Una vez obtenidos los datos masivos, la mejora del procesamiento y el posterior análisis de los datos permitirán:

- Aumentar la eficacia y la eficiencia tanto en el sector privado como en el sector público.
- Con carácter general, incrementar la productividad de todos los sectores de la economía mediante la implementación de la inteligencia empresarial.
- Lograr la reducción de costes, derivado de la incorporación de servicios más personalizados.
- Transformar las industrias de servicios mediante la generación de una amplia gama de productos y servicios de información innovadores.
- Mejorar la investigación y acelerar la innovación.
- En suma, afrontar de manera eficiente el conjunto de desafíos que debe superar la sociedad del siglo XXI.

Derivado de lo anterior, aparece un nuevo concepto: la **analítica digital**, que definimos como el protocolo por el cual los diferentes operadores son capaces de recoger las interacciones que realizan los usuarios con el conjunto de aplicaciones digitales, entre las que se encuentran el acceso a páginas web, redes sociales o las *app*³.

La finalidad consiste en cuantificar los datos obtenidos de los usuarios, tratarlos convenientemente a fin de extraer conclusiones que aporten un nuevo valor añadido.

Los datos obtenidos de la analítica digital, "**aplicados a la economía y a la administración y dirección de empresas**", ofrecen notables beneficios, entre otros:

1. Accesibilidad y uso de las páginas web.
2. Valoración de los ratios financieros de nuestra compañía.
3. Conocimiento de los hábitos de consumo de los usuarios.
4. Conocimiento del público objetivo.
5. Facilitar e incrementar el proceso de compra.
6. Optimizar el posicionamiento de la página web en el ranking de los diferentes motores de búsqueda⁴.
7. Optimizar nuestro marketing empresarial⁵ o publicidad en los motores de búsqueda.
8. Interés de los usuarios por la marca.
9. Descubrir nuevos nichos de negocio.

En una primera clasificación de las aplicaciones podemos dividirlas en dos grandes bloques:

- *Aplicaciones destinadas al análisis de páginas web*: como Google Analytics, Webtrends, SEMrush, Alexa, entre otras.
- *Aplicaciones destinadas al análisis de redes sociales*⁶: como Twitter Analytics, Facebook Insights, Twitonomy, TweetStats, TwitReach, Audiense, LikeAlizer, Iconosquare, Stigram, SumAll, Pinterest Analytics, entre otras.

El proceso finaliza mediante la presentación y la interpretación de los datos obtenidos. El resultado final deberá ser entendible, visual y comparable en función de los diferentes períodos temporales analizados.

Es en este contexto cuando se empieza a hablar de **Data Science**, disciplina que engloba tanto las técnicas estadísticas tradicionales como las derivadas del nacimiento del Big Data, y que ha de permitirnos abordar el análisis de cualquier tipo de datos independientemente de su volumen o heterogeneidad.

2.1. CONCEPTOS CLAVE

Antes de entrar en mayor profundidad, conviene tener claros una serie de conceptos que nos permitirán acercarnos al tema desde una perspectiva más global.

³ App, acrónimo de "application", es decir, aplicación de software que se instala en dispositivos móviles.

⁴ Técnica que responde al acrónimo SEO, *Search Engine Optimization*.

⁵ Técnica que responde al acrónimo SEM, *Search Engine Marketing*.

⁶ Redes sociales como Twitter, Facebook, Instagram o Pinterest.

- ***Big Data***

Como ya indicamos previamente, *Big Data*⁷, macrodatos, datos masivos o inteligencia de datos, corresponde a una terminología que se emplea en el sector de las tecnologías de la información y de la comunicación para aludir a un conjunto de datos que, por su volumen, variedad y por la velocidad a la que necesitan ser procesados, supera las capacidades de los sistemas informáticos habituales. Si bien, podemos ampliar el término *big data* como la capacidad de gestionar un enorme volumen de datos heterogéneos, procesados a una velocidad adecuada y dentro del marco de tiempo razonable, para permitir el análisis y la reacción en tiempo real.

Las características principales de *big data* son las siguientes:

1. Volumen: ¿cuántos datos?
2. Velocidad: ¿con qué rapidez esos datos se procesan?
3. Variedad: ¿los diferentes tipos de datos o su heterogeneidad?
4. Veracidad: ¿fiabilidad de los datos utilizados en el proceso de toma de decisiones?
5. Valor: ¿obtener información de los grandes datos de manera rentable?

FIGURA 2. LAS 5V EN *BIG DATA*



Fuente: adaptado de Russom (2011)

En un principio, el *big data* se caracterizaba por las denominadas 3V (volumen, velocidad y variedad) pero con posterioridad, se han añadido nuevas características como la veracidad y el valor. Por este motivo, aparecen las 5V en big data.

- ***Data Science (Ciencia de los Datos)***

Como hemos visto, el *Big Data* ha traído consigo el nacimiento de una nueva disciplina relacionada con el tratamiento y posterior análisis de grandes volúmenes de datos. Como decíamos, esta disciplina se desarrolla fundamentalmente para manejar la gran cantidad de datos que se genera en la actualidad a nivel mundial, fruto del proceso de globalización.

El *Data Science* precisa la interacción de diversas disciplinas como la estadística, matemática, ingeniería de datos, reconocimiento de patrones y finalmente su visualización. Asimismo, otras disciplinas que contribuyen al *data science* son la inteligencia competitiva, la analítica de negocios o la biología, entre otras. A todo esto es preciso incorporar, la creatividad de los científicos de datos cuya competencia requerirá de las siguientes habilidades:

⁷ Fundeu (2017)

- Habilidad para recopilar, procesar y extraer valor del conjunto de los diversos datos.
- Imaginación para comprender, visualizar y comunicar sus hallazgos al resto de la población no especializada en esta nueva disciplina.
- Habilidad para crear soluciones basadas en datos que aumenten los beneficios, reduzcan los costes y tiendan a mejorar la calidad de vida de los ciudadanos.

En resumen, conviene no solo estudiar los diferentes modelos y técnicas a nivel teórico, sino que se requiere de coherencia y sentido común en su aplicación práctica.

- **Data Mining (Minería de Datos)**

Sin lugar a duda, el *Data Mining* o Minería de Datos⁸ constituye una de las técnicas más utilizadas en el análisis de los grandes datos. Definimos el *data mining* como el análisis masivo de la información contenida en una base de datos o *data warehouse* para extraer relaciones, patrones de comportamiento, tendencias, ciclos estacionales, anomalías, etc., sin tener en cuenta el significado preciso de los datos que se analizan.

El gran reto actual es el análisis de los datos procedentes de las redes sociales, blogs, wikis, etc. y su estudio ha dado origen a nuevas categorías encuadradas en la minería de datos conocida como minería social y minería de opinión. Asimismo, hoy se considera también una nueva categoría propiciada por el intercambio de datos entre objetos, principalmente sensores, conocida como analítica M2M (máquina a máquina).

Por lo tanto, dentro del concepto más general de *Data Mining*, podemos incluir otros con ámbitos más concretos, como el *Web Mining*, el *Text Mining* o la *Minería o Análisis de Sentimientos*.

- **Web Mining**

El *web mining* constituye una metodología para la obtención de información que emplea algoritmos y técnicas de la minería de datos (y por tanto de la que forma parte), con la finalidad de extraer información de las webs, de los enlaces de las mismas y de los registros de navegación.

En la actualidad (Joyanes, 2014) la web es la fuente de *Big Data* más utilizada debido a que podemos almacenar los contenidos de las páginas web y capturar las transacciones generadas en cada visita a las páginas web. Las técnicas actuales nos posibilitan procesar, almacenar y analizar cantidades ingentes de información que una vez procesada nos permitirá conocer las características de nuestros clientes, pero no solo de aquellas transacciones que se han llevado a cabo sino las que en un futuro se realizarán. Por tanto, el conocimiento del comportamiento de los clientes en la web es básico en el proceso actual de toma de decisiones empresarial.

- **Text Mining**

También conocida como *Minería de Texto* (Luong, 2012), se refiere fundamentalmente al proceso de extraer patrones o conocimientos relevantes, utilizando documentos de texto estructurados e incluso no estructurados. Es esta una tarea sumamente compleja debido a que implica tratar con datos de texto que son inherentemente no estructurados y difusos. La minería de textos es una materia multidisciplinar, que incluye la recuperación de

⁸ Joyanes (2014:102, 386)

información, análisis de texto, lingüística computacional, extracción de información, así como la agrupación, categorización y visualización de las bases de datos.

- ***Análisis de Sentimientos***

También conocido como *Opinion Mining* o *Minería de Opinión* (Baláhur, 2011) tiene por finalidad la detección automática de sentimientos expresados en textos y su posterior clasificación. El sentimiento proviene de una fuente, que puede ser una persona, un evento, un producto o una organización, que expresa su parecer sobre un objeto. Una vez conocida la opinión, se procede a la clasificación en función de la tendencia manifestada como, por ejemplo, positiva, neutra o negativa.

Esta aparente simplicidad conceptual esconde una metodología enormemente compleja que precisa de un análisis desde diferentes puntos de vista, en función de los factores que estemos investigando.

- ***Business Intelligence***

El *Business Intelligence (BI)* o *Inteligencia de Negocios* (IBM, 2009), es un concepto originario de 1960, y que en la actualidad se refiere al proceso de toma de decisiones empresariales fundamentadas en el análisis y gestión del *big data* y en la automatización de procesos. Se trata de un proceso interactivo para explorar y analizar grandes bases de datos, estructuradas y no estructuradas, con la finalidad de descubrir tendencias o patrones a partir de los cuales derivar ideas y extraer conclusiones. Además, el proceso BI incorpora la comunicación de los descubrimientos y realizar, si fuera procedente, los cambios oportunos en la empresa. El proceso, por tanto, afecta a clientes, proveedores, competidores, productos y servicios.

Analizando los datos existentes en una organización, generamos conocimiento sobre el medio, cuantitativo y cualitativo, capaz de generar nuevas estrategias, planteamientos y soluciones empresariales. Se trata, en definitiva, de diferenciar datos, información y conocimiento, en función del nivel de precisión alcanzado, con la finalidad de establecer, una vez determinadas las fortalezas y debilidades, la estrategia empresarial.

- ***Machine Learning***

El *Machine Learning, Aprendizaje de Máquinas* o *Aprendizaje Automático*, es una materia derivada de las ciencias de la computación y de la inteligencia artificial que intentan establecer técnicas que, en última instancia, posibiliten a los dispositivos aprender. Se trata de un método de análisis de datos¹⁰ que automatiza la construcción de modelos analíticos, mediante el uso de algoritmos que interactivamente aprenden de los datos, encontrando parámetros ocultos sin estar explícitamente programado el lugar de la búsqueda.

El *Machine Learning* tiene, actualmente, una multitud de aplicaciones como, entre otras, motores de búsqueda, diagnósticos médicos, detección de fraude, análisis de los mercados financieros, clasificación de secuencias de ADN, reconocimiento del habla y del lenguaje escrito, juegos, robótica, etc.

⁹ IBM (2009).

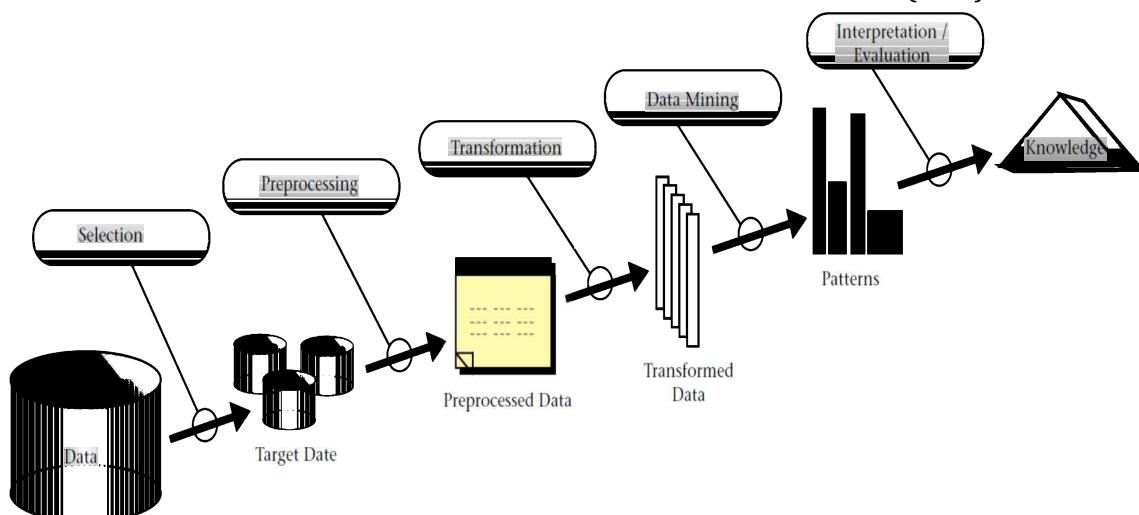
¹⁰ https://www.sas.com/en_gb/insights/analytics/machine-learning.html

2.2. MINERÍA DE DATOS O DATA MINING

Como ya hemos indicado en el epígrafe anterior, el *Data Mining* o la Minería de Datos hace referencia a la obtención, preparación y gestión de un volumen elevado de datos para, por medio de la aplicación de herramientas de naturaleza matemática y estadística, explotarlos y analizarlos (de forma automática o semiautomática) al objeto de identificar patrones de comportamiento, tipologías, correlaciones y predicciones.

Este concepto no es nuevo. Fue en la década de los ochenta cuando se empezó a consolidar los términos *Data Mining* y Extracción de Conocimiento en Bases de Datos (*Knowledge Discovery in Databases*, KDD), si bien el modelo como tal se adaptó en 1996. De hecho, el KDD está compuesto por cinco etapas: 1) Selección de Datos; 2) Procesamiento; 3) Transformación; 4) *Data Mining* y 5) Interpretación y Evaluación. La minería de datos solo se correspondería con la cuarta etapa del modelo KDD, no obstante, se ha generalizado su definición y se ha ampliado a todo el proceso, como señalan Virseda y Román (2006).

FIGURA 3. FASES DEL PROCESO DE EXTRACCIÓN DEL CONOCIMIENTO (KDD)



Fuente: Fayyad *et al.* (1996)

El KDD fue el primero de los modelos de aplicación en proyectos de *Data Mining*, si bien a lo largo de los años han aparecido otros, que son, en esencia, procesos normalizados de actuación que fijan unas fases o procesos de actuación en la minería de datos.

Los más conocidos y utilizados en la actualidad son el modelo SEMMA (que es el acrónimo de las diferentes fases del tratamiento de la información: *Sample, Explore, Modify, Model, Assess*) y el CRISP-DM (*CRoss Industry Standard Process for Data Mining*). Sin perjuicio de la existencia de otras metodologías como la propia KDD o *Catalyst* (también conocida como *P3TQ*), o, incluso, metodologías propias de algunas empresas, que son más marginales en cuanto a uso. Por ello, no desarrollaremos estos modelos, dando la posibilidad al estudiante que profundice en la materia si estuviera interesado.

2.3. MODELOS SEMMA Y CRISP-DM

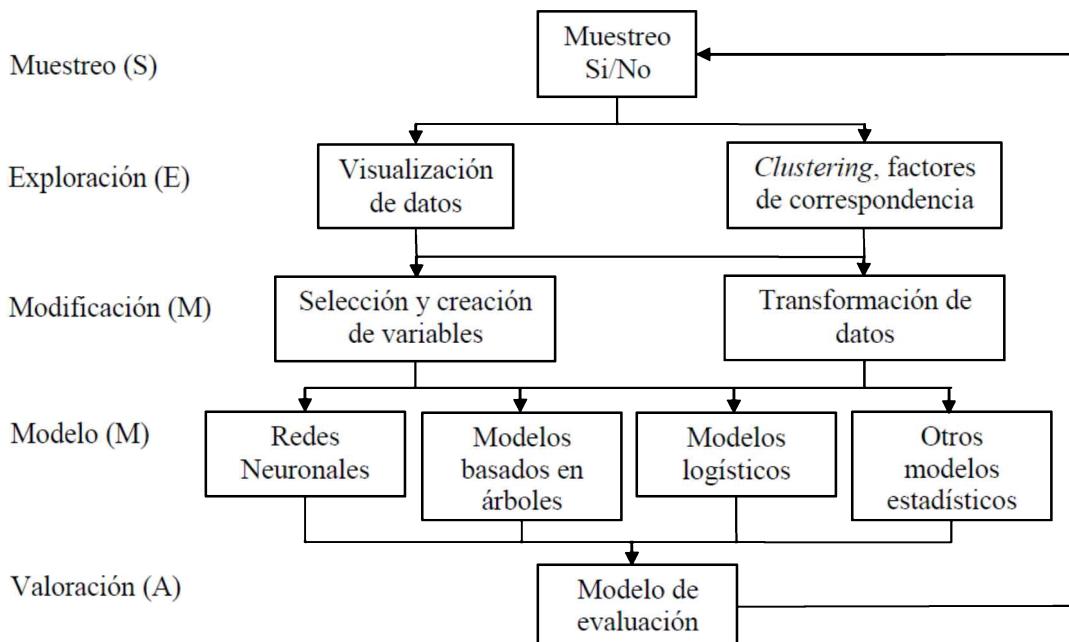
Es habitual en el ámbito empresarial que cada compañía se decante e implante una de las metodologías concretas, la cual aplicará con carácter general a todos los proyectos de *Data Mining*. Si bien, las diferentes fases que las definen, así como sus características particulares,

pueden ser más adecuadas según el proyecto concreto que se quiera acometer (Rodríguez *et al.*, 2003).

2.3.1. Modelo SEMMA

El modelo SEMMA fue desarrollado por SAS *Institute* y se define como el proceso de selección, exploración y modelado de grandes volúmenes de datos para descubrir patrones de negocio desconocidos (SAS, 1998). Se compone de 5 fases o etapas concatenadas, aunque en sí mismo es un modelo cíclico cuyas actividades internas se pueden realizar iterativamente según el proyecto y los resultados del análisis.

FIGURA 4. FASES DEL PROCESO SEMMA



Fuente: SAS (1998) y Britos *et al.* (2008)

A continuación, describimos brevemente cada una de las fases (Vanrell, 2011):

- *Fase I - Muestreo (Sample):* En esta primera etapa se extrae la población muestral representativa sobre la cual se aplicará el análisis, la cual deberá ser representativa para obtener unos resultados fiables. El método más habitual es el "muestreo aleatorio simple", seleccionando de manera aleatoria los individuos que compondrán la muestra.
- *Fase II - Exploración (Explore):* El siguiente paso es realizar una exploración de la información para simplificar el problema y así optimizar la eficiencia del modelo. Para lograrlo se propone el uso de herramientas de visualización o técnicas estadísticas que ayuden a poner en manifestó las relaciones entre las variables. Con esto se pretende determinar cuáles son las variables explicativas que se utilizarán como entradas del modelo.
- *Fase III - Modificación (Modify):* En esta fase se modifican y manipulan los datos, en función de la exploración realizada, de forma que sean correctamente definidos y con el formato adecuado para introducirlos en el modelo.
- *Fase IV - Modelado (Model):* Es esta fase se identificarán las relaciones entre las diferentes variables a fin de obtener conclusiones válidas que aporten valor a los

datos estudiados. Para ello, se utilizarán técnicas estadísticas muy variadas, que van desde los modelos estáticos tradicionales hasta los más avanzados como árboles de decisión, redes neuronales, etc.

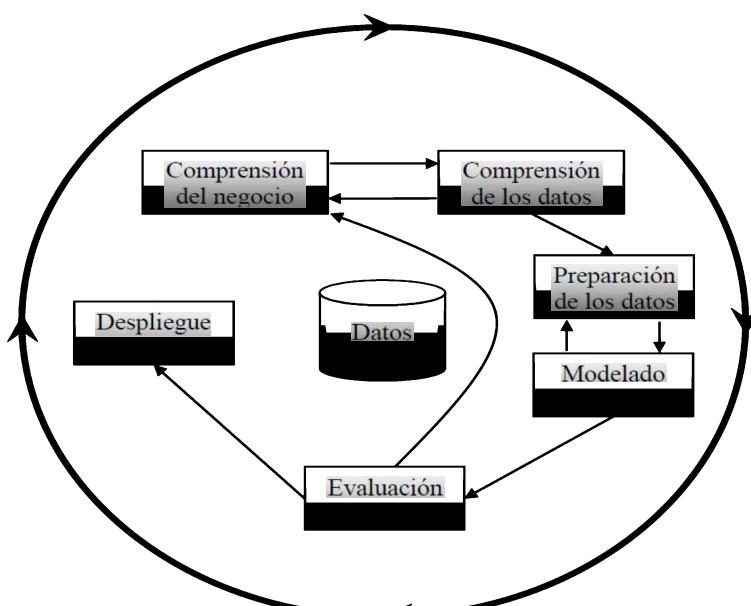
- *Fase V - Valoración (Assess)*: Finalmente se procederá a determinar la bondad del modelo, contrastando los datos obtenidos con otros métodos estadísticos o con otras poblaciones muestrales.

La metodología SEMMA está enfocada en los aspectos técnicos de los proyectos de *Data Mining*. Al ser desarrollada por SAS está diseñada para trabajar con sus herramientas (a las que denomina "nodos"), ofreciendo software específico para la ejecución de cada una de las fases.

2.3.2. Modelo CRISP-DM

En el año 2000, las empresas SPSS, NCR y Daimler Chrysler desarrollaron la metodología CRIPS-DM, que es actualmente la más utilizada en los proyectos de minería de datos. El modelo está organizado en torno a una consecución jerárquica de procesos a diferentes niveles, que inicialmente y de manera general se estructura en seis etapas: comprensión del negocio (o análisis del problema), comprensión de los datos, preparación de los datos, modelado, evaluación e implementación.

FIGURA 5. FASES DE LA METODOLOGÍA CRISP-DM



Fuente: Chapman *et al.* (2000)

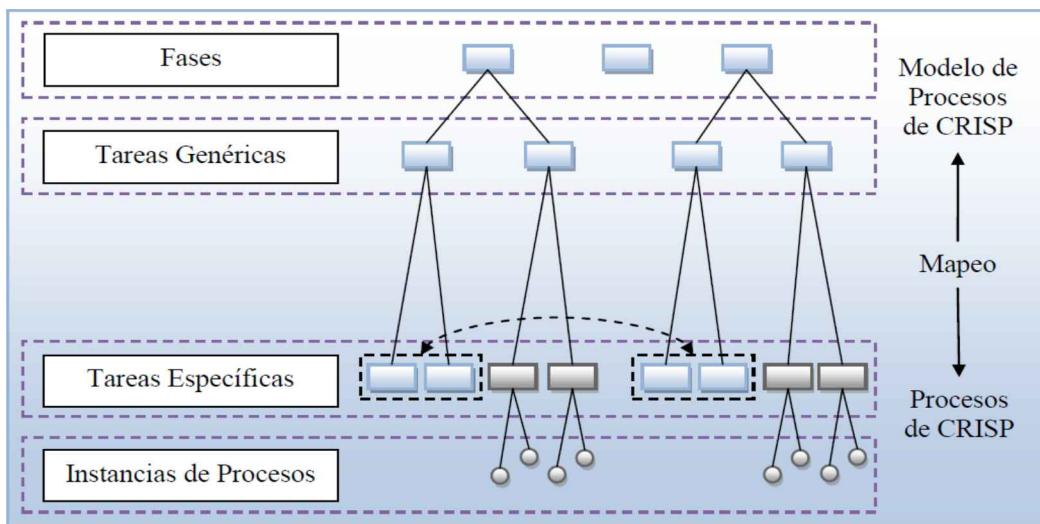
Las cuatro fases intermedias (comprensión de los datos, preparación de los datos, modelado, evaluación) coinciden en términos generales con lo definido para la metodología SEMMA, pero presenta dos fases nuevas:

- *Fase I - Compresión del negocio*: Desde una perspectiva general que permita conocer el negocio y las necesidades del mismo, podremos definir o delimitar un problema que debe ser resuelto a través del procesamiento de los datos y, por tanto, establecer unos objetivos para el proyecto de *Data Mining*.

- *Fase VI - Implementación:* Es la última fase del proceso y consiste en la explotación práctica de los resultados, que puede ser algo tan sencillo con la redacción de un informe o tan complejo como el desarrollo de un nuevo modelo de gestión.

No obstante, como indicábamos a comienzo del subepígrafe, la metodología CRISP-DM se plantea en varios niveles. El primer nivel son las fases previamente definidas, pero por debajo de estas se establecen listados de tareas: 1) tareas genéricas; 2) tareas específicas y 3) instancias de proceso, lo que supone hasta un cuarto nivel de desglose. A medida que descendemos por los niveles, el nivel de concreción es mayor, ofreciendo la metodología un listado de procesos muy concreto. No obstante, el modelo muestra la flexibilidad necesaria para adaptarse a cualquier proyecto de *Data Mining*, no siendo imprescindible la realización de todas y cada una de las tareas.

FIGURA 6. NIVELES DE LA METODOLOGÍA CRISP-DM



Fuente: Chapman *et al.* (2000)

Planteemos un ejemplo para entenderlo mejor: Una de las fases de la metodología, en el primer nivel, es la "preparación de los datos". Por debajo de ella, encontraremos una tarea general (segundo nivel) que será "limpieza de datos". En el tercer nivel, se propondrán como tareas específicas la "limpieza de datos numéricos" o la "limpieza de datos categóricos". Finalmente, dentro de las instancias de proceso se establecerán las acciones necesarias para llevar a cabo la actividad.

Los usuarios disponen de un manual de CRISP-DM y de una guía, en los cuales se explican los procesos, tareas, recomendaciones y orientaciones para desarrollar un proyecto, si bien no especifica detalladamente el "cómo". Esa será una tarea del propio usuario (Chapman *et al.*, 2000; IBM, 2012).

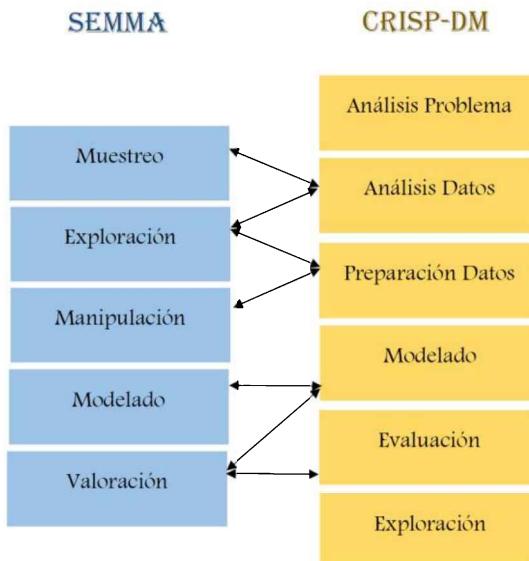
2.3.3. Diferencias entre SEMMA y CRISP-DM

Existen importantes diferencias entre ambas metodologías, la más notable, como señalan algunos autores, responde a la definición propia de cada modelo. Según sostiene Pressman (2005), un modelo de proceso, que se correspondería con el KDD o el SEMMA, es un conjunto de tareas para realizar un proyecto, definiéndose también las entradas y salidas. Por su parte, una metodología, no solo define las actividades las tareas y las entradas y salidas, sino que también especifica las actividades de ejecución, como pasa en el SCRISP-DM o en el

Catalyst. Por lo tanto, mientras que el *SEMMA* es un modelo de proceso, el *CRISP-DM* se consideraría una metodología.

Por otra parte, el modelo *SEMMA* se centra en las características técnicas del proyecto de minería de datos, mientras que la metodología *CRISP-DM* mantiene una perspectiva más amplia respecto a los objetivos empresariales del proyecto, cuestión que se justifica por la inclusión de una etapa inicial y final, que ya hemos comentado.

FIGURA 7. COMPARATIVA ENTRE LAS FASES DEL *SEMMA* Y DEL *CRISP-DM*



Fuente: Rodríguez *et al.* (2003)

De manera resumida en la siguiente tabla, comparamos las principales características entre las dos metodologías.

TABLA 1. PRINCIPALES CARACTERÍSTICAS DE LAS METODOLOGÍAS *SEMMA* Y *CRISP-DM*

	SEMMA	CRISP-DM
Permite elección libre de las herramientas	NO	SI
Cantidad de fases	5	6
Todas las fases pueden relacionarse	NO	SI
Considera los motivos del proyecto	NO	NO
Considera la naturaleza del interés de las partes	NO	NO
Considera otros aspectos no técnicos	NO	SI
Identifica claramente las variables sobre las cuales el proyecto tiene impacto	NO	NO
Está detallada paso a paso cada etapa del método	NO	NO
Identifica problemas de inteligencia de negocio (PIN)	NO	SI
Identifica una caracterización abstracta de PIN	Parcialmente	NO
Identifica técnicas de Explotación de Información (TEI) utilizables	SI	SI
Identifica relaciones entre las TEI y los PIN	Parcialmente	NO
Identifica procesos de explotación de información (procesos PINxTEI)	Parcialmente	NO

Fuente: Méndez y Rodríguez (2009) y Britos (2008)

Finalmente, también presentan divergencias en cuanto a la propiedad y uso. *SEMMA* es una licencia privada de *SAS*, mientras que *CRISP-DM* es un software libre.

2.4. PRINCIPALES MÉTODOS Y ALGORITMOS EN LA MINERÍA DE DATOS

Aunque los principales métodos de análisis de datos serán vistos de manera conceptual y práctica, fundamentalmente a través de **R**, en este epígrafe se ofrece una visión general, clasificando los algoritmos con el objeto de contextualizar de manera más clara las diferentes técnicas.

Existen diversas clasificaciones en relación con las técnicas de *Data Mining*, según el criterio de agruparlas. Con carácter general, parece lógico que la primera clasificación tenga en consideración el objeto del análisis. Así, podemos diferenciar entre:

- **Métodos descriptivos:** Aquellos que pretenden definir, clasificar, catalogar o caracterizar el objeto de estudio. Es un análisis *ex post-facto* (después del hecho).
- **Métodos predictivos:** Aquellos métodos que mediante el estudio de los datos intentan adelantarse a ciertos eventos o pronosticar aspectos de comportamiento futuro. Es un tipo de análisis *ex ante*.

TABLA 2. PRINCIPALES ALGORITMOS DE MACHINE LEARNING

REGRESIÓN	ENSAMBLADO (ENSAMBLE)
Mínimos Cuadráticos Ordinarios	Bosques Aleatorios (Random Forest)
Regresión Stepwise	Potenciación del Gradiente (Gradient Boosting Machine)
Modelo Lineal Generalizado	Boosting
Mínimos Cuadrados en Dos Fases	Bootstrapped Aggregation (Bagging)
Regresión Adaptativa Multivariante	AdaBoost
Regresión Polinómica Ponderada	Gradiente de Árboles Boosting (GBRT)
Regresión Logística	Decorate
Regresión Ridge	Stacking
Regresión Robusta	
Regresión Lasso	
Regresión Elastic Net	
Regresión del Ángulo Mínimo (LARS)	
Cubist	
Stochastic Gradient Boosting	
Máquinas de Vectores Soporte	
REDES NEURONALES Y DEEP LEARNING	SISTEMA DE REGLAS
Modelo Conductual Desarrollativo (DBM)	One Rule (OneR)
Red de Creencia Profunda (DBN)	Zero Rule (ZeroR)
Redes Neuronales Convolucionales (CNN)	Repeater Incremental Pruning to Produce Error Reduction (RIPPER)
Auto-codificadores Acumulativos	Decision Table
Función de Base Radial (RBFN)	MSRules
Red Neuronal Perceptrón Multicapa	PART
Retropropagación	
Red Neuronal de Hopfield	
REDUCCIÓN DE DIMENSIONES	MÉTODOS BAYESIANOS
Análisis de Componentes Principales (PCA)	Naive Bayes
Regresión de Mínimos Cuadráticos Parciales (PLSR)	Averaged One-Dependence Estimators (AODE1 y AODE2)
Mapeo de Sammon	Modelo Lineal Generalizado Bayesiano
Escalamiento Multidimensional (MDS)	Bayesian Belief Network (BBN) / Redes Bayesianas
Búsqueda de Proyecciones (Projection Pursuit)	Naive Bayes Gaussiana
Regresión sobre Componentes Principales	Naive Bayes Multinomial
Ánálisis Discriminante del Mínimo Cuadrático Parcial	Redes Bayesianas (TAN, K2...)
Ánálisis Discriminante Múltiple (MDA)	
Ánálisis Discriminante Cuadrático (QDA)	
Ánálisis Discriminante Regularizado (RDA)	
Ánálisis Discriminante Flexible (FDA)	
Ánálisis Discriminante Lineal (LDA)	
MÉTODOS DE INSTANCIA	ÁRBOLES DE DECISIÓN
Métodos de los K Vecinos (KNN)	Árbol de Regresión y Clasificación (CART)
Cuantificación Vectorial (LVQ)	Iterative Dichotomiser 3 (ID3)
Mapas Auto-Organizados (SOM)	C4.5
Aprendizaje Localmente Ponderado (LWL)	C5.0
KStar	Chi-Squared Automatic Interaction Detection (CHAID)
REGLAS DE ASOCIACIÓN	Árbol de Decisión de un Nivel (Decision Stump)
A Priori	Conditional Decision Trees
FPGrow	MSP
Fibered Associator	
CLUSTERING	
	K-Medias
	K-Medianas
	Esperanza-Maximización (EM)
	Clasificación Jerárquica
	CobWeb
	Canopy
	Cluster Bietápico
	Clústeres Basados en Densidad
SERIES TEMPORALES	
	Métodos de suavizado
	Splines
	Modelos ARIMA

Fuente: Elaboración propia

Otra clasificación, con más detalle, atendiendo a las características y tipo de análisis realizado por cada método es la recogida en la Tabla 2.

Como se puede observar en la tabla anterior, existen muchas técnicas estadísticas, tanto paramétricas como no paramétricas: árboles de decisión, redes neuronales, algoritmos genéticos, máquinas de vectores soporte, modelos logit, probit y tobit, análisis discriminante, redes bayesianas, multiclasificadores, etc. Algunas de las principales se describen a continuación de forma somera.

Los **árboles de decisión** son particiones secuenciales de un conjunto de datos que maximizan las diferencias de la variable dependiente. Nos ofrecen una forma concisa de definir grupos que son consistentes en sus atributos pero que varían en términos de la variable dependiente. Esta herramienta puede emplearse tanto para la resolución de problemas de clasificación como de regresión: árboles de clasificación y árboles de regresión. Mediante esta técnica se representan de forma gráfica un conjunto de reglas sobre las decisiones que se deben de tener en cuenta para asignar un determinado elemento a una clase (valor de salida).

Las **redes neuronales** tratan de emular el comportamiento cerebral. Una red neuronal puede describirse mediante cuatro conceptos: el tipo de modelo de red neuronal; las unidades de procesamiento que recogen información, la procesan y arrojan un valor; la organización del sistema de nodos para transmitir las señales desde los nodos de entrada a los nodos de salida y, por último, la función de aprendizaje a través de la cual el sistema se retroalimenta.

Se considera una red neuronal como la ordenación secuencial de tres tipos básicos de nodos o capas: nodos de entrada, nodos de salida y nodos intermedios (capa oculta o escondida).

Los nodos de entrada se encargan de recibir los valores iniciales de los datos de cada caso para transmitirlos a la red. Los nodos de salida reciben entradas y calculan el valor de salida (no van a otro nodo). En casi todas las redes existe una tercera capa denominada oculta. Este conjunto de nodos utilizados por la red neuronal, junto con la función de activación posibilita a las redes neuronales representar fácilmente las relaciones no lineales, que son muy problemáticas para las técnicas multivariantes.

Cuando se presenta un patrón de entrada $X_p : x_{p1}, \dots, x_{pi}, \dots, x_{pN}$ se transmite a la red a través de los pesos w_{ji} desde la capa de entrada a la capa oculta. Las neuronas de esta capa transforman las señales a través de la función de activación proporcionando un valor de salida. Este valor se transmite a su vez a través de los pesos v_{kj} a la capa de salida donde aplicando de nuevo la función de activación obtenemos un valor de salida.

Los fundamentos teóricos de las **máquinas de vectores soporte (Support Vector Machines, SVM)** fueron presentados en el año 1992 en la conferencia COLT (Computacional Learning Theory) por Boser, Guyon y Vapnik (1992) y descritos posteriormente en diversos artículos por Cortes y Vapnik [Cortes y Vapnik (1995)]; Vapnik [(1998) y (2000)] a partir de los trabajos sobre la teoría del aprendizaje estadístico.

Las máquinas de vectores soporte pertenecen a la familia de los clasificadores lineales dado que inducen hiperplanos o separadores lineales de muy alta dimensionalidad introducidos por funciones núcleo o kernel. Es decir, el enfoque de las SVM adopta un punto de vista no habitual, en vez de reducir la dimensión buscan una dimensión mayor en la cual los puntos puedan separarse linealmente.

Los **algoritmos genéticos** propuestos por Holland (1975) suponen uno de los enfoques más originales en la minería de datos, se inspiran en el comportamiento natural de la evolución, para ello se codifica cada uno de los casos de prueba como una cadena binaria (que se

asemejaría a un gen). Esta cadena se replica o se inhibe en función de su importancia, determinada por una función denominada de ajuste o fitness.

Los algoritmos genéticos son adecuados para obtener buenas aproximaciones en problemas de búsqueda, aprendizaje y optimización, Marczik, (2004).

De forma esquemática un algoritmo genético es una función matemática que tomando como entrada unos individuos iniciales (población origen) selecciona aquellos ejemplares (también llamados genes) que recombinándose por algún método generarán como resultado la siguiente generación. Esta función se aplicará de forma iterativa hasta verificar alguna condición de parada, bien pueda ser un número máximo de iteraciones o bien la obtención de un individuo que cumpla unas restricciones iniciales.

Las **redes bayesianas**, que también se conocen en la literatura con otros nombres: redes causales o redes causales probabilísticas, redes de creencia, sistemas probabilísticos, sistemas expertos bayesianos o también como diagramas de influencia, son métodos estadísticos que representan la incertidumbre a través de las relaciones de independencia condicional que se establecen entre ellas, Edwards (1998). Este tipo de redes codifica la incertidumbre asociada a cada variable por medio de probabilidades. Kadie *et al.* (2001) afirman que una red bayesiana es un conjunto de variables, una estructura gráfica conectada a estas variables y un conjunto de distribuciones de probabilidad.

Estas redes probabilísticas automatizan el proceso de modelización probabilístico utilizando toda la expresividad de los grafos para representar las dependencias y la teoría de la probabilidad para cuantificar esas relaciones. En esta unión se plasma de forma eficiente tanto el aprendizaje automático como la inferencia con los datos y la información disponible.

Las redes bayesianas tienen la habilidad de codificar la causalidad entre las variables, por lo que han sido muy utilizadas en el modelado o en la búsqueda automática de estructuras causales, López *et al.* (2006). La potencia de las redes bayesianas está en su capacidad de codificar las dependencias/independencias relevantes considerando no solo las dependencias marginales sino también las dependencias condicionales entre conjuntos de variables.

La técnica estadística clásica más conocida y utilizada en clasificación es el **modelo de regresión logística**. Es un tipo de análisis de regresión utilizado para predecir el resultado de una variable categórica (una variable que puede adoptar un número limitado de categorías) en función de las variables independientes o predictoras. Es útil para modelar la probabilidad de un evento ocurriendo como función de otros factores. El análisis de regresión logística se enmarca en el conjunto de Modelos Lineales Generalizados (GLM), usando como función de enlace la función logit, es decir, las probabilidades que describen el posible resultado de un único ensayo se modelan, como una función de variables explicativas, utilizando una función logística.

La regresión logística es utilizada extensamente en las ciencias médicas y sociales. Otros nombres usados en varias áreas de aplicación serían: modelo logístico, modelo logit y clasificador de máxima entropía.

Los **Multiclasificadores** como combinación de modelos representan una excelente forma de conseguir una mayor precisión en las predicciones de nuestros modelos. La combinación de las hipótesis de los multiclasificadores es una manera de integrar la información de diferentes fuentes. Esta combinación de dos o más clasificadores, en general, proporciona

estimaciones más robustas y eficientes que cuando se utiliza un único clasificador. También se utilizan porque resuelven el problema de sobreadaptación (overfitting) y es posible obtener buenos resultados con pocos datos. Son múltiples los estudios que se han realizado con los métodos multiclasicadores, así que podemos conocerlos en la literatura existente con muchos nombres: métodos de ensamble, modelos múltiples, sistemas de múltiples clasificadores, combinación de clasificadores, integración de clasificadores, mezcla de expertos, comité de decisión o fusión de clasificadores de aprendizaje multimodelo.

3. BIG DATA

Big data, traducido como grandes datos o datos masivos, es la consecuencia lógica a la que hemos derivado por el imparable crecimiento de las comunicaciones y la tecnología, unido al desarrollo de aplicaciones integradas y relacionadas que permiten amplificar el efecto de manera exponencial. Esta circunstancia ha sido secundada, o al menos debería haberlo sido, por un incremento en el conocimiento y la sabiduría. Como se apunta en el informe de la Fundación Innovación Bankinter (Paniagua, 2015), en los últimos cinco años se ha generado más información científica que en toda la historia de la humanidad. Ahora bien, la pregunta inmediata es la siguiente: ¿tenemos capacidad, no de acceso, sino para leer, procesar y asimilar toda esa información? La respuesta es, evidentemente, no.

Son muchos, y cada vez más, los que consideran al *Big Data* la nueva ciencia, definiéndola como el estudio, tratamiento y análisis de grandes volúmenes de datos de los que obtener conclusiones concretas. Ahora bien, sería necesario completar lo anterior con una cuestión básica: *¿qué es lo que se busca?* Tener un gran volumen de información y las técnicas para tratarlo puede ser inútil si no se sabe qué es lo que se quiere obtener.

3.1. DESAFÍOS

El nuevo escenario definido por la transformación digital está condicionando de manera evidente la forma de comunicarnos, relacionarnos y la toma de decisiones en todos los ámbitos. El proceso de cambio está siendo tan rápido que apenas somos capaces de asimilar su globalidad. Pero a su vez, los que antes se adapten contarán con una ventaja comparativa significativa, como consecuencia de las grandes oportunidades que se abren.

Si retrocedemos unos pocos años en el tiempo, encontramos un hito fundamental: en 1990 aparece la *World Wide Web*, y mientras Internet se comercializaba y se extendía su uso entre la población mundial, se creaban nuevas formas de comunicación que transformaban nuestros hábitos y, paralelamente, se generaban grandes volúmenes de información. Así, el término *Big Data* se emplea por primera vez en 1997 para referirse a aquella cantidad de datos que no podían ser contenidos en un disco duro (Cox y Ellsworth, 1997). Paralelamente, durante esos años se acuñó una clasificación de los datos según su naturaleza: aquellos conocidos como "*born analog*", que proceden del mundo físico aunque pueden ser transformados en digitales; en contraposición de los "*born digital*" creados en su uso digital, que son los más numerosos. La verdadera revolución la situamos a partir de 2010 con el desarrollo de técnicas y herramientas que permiten el tratamiento y análisis de los grandes volúmenes de datos.

En poco más de 20 años se ha producido la transformación del entorno como jamás se había imaginado, bajo unas nuevas reglas del juego que muchos identifican con los datos. No

obstante, para evitar que el propio proceso de transformación nos engulla conviene tener una perspectiva global de la situación, contextualizando adecuadamente los conceptos y delimitando convenientemente los ámbitos de actuación y utilidad.

3.1.1. Open Data

Aunque estamos en la era de los datos y cada individuo genera cientos o miles de datos al día (pago con tarjetas de crédito, redes sociales, datos profesionales, etc.) no significa que se tenga acceso a todos ellos de manera general. Muchas de las fuentes de datos son inaccesibles para el público en general y otras tienen coste. Luego, aunque se hable de *Big Data* con mucha laxitud, es conveniente delimitar el acceso a las fuentes, pues en muchos casos presentan restricciones.

En contraposición a esto, cada vez se populariza más el uso de "*Open Data*" o dato abierto, a los que tienen acceso cualquier público en general. Si bien, incluso este concepto debe ser matizado, pues no toda base de datos que se ofrezca puede ser considerada *Open Data*, pues debería poseer unas características:

- *Primarios*: Deben provenir de la fuente inicial que los creó.
- *Accesibles*: Deben estar disponibles para cualquier usuario que quiera hacer uso de ellos.
- *Completos*: Reflejarán el total de la información y variables que contienen, evitando los datos nulos o perdidos.
- *Procesables*: Que estén en formato adecuado para ser tratados convenientemente.
- *Sin propiedad*: Que ninguna entidad tenga el control exclusivo de ellos.
- *Actualizados*: Deben contener los últimos datos disponibles para que el tratamiento de los datos sea lo más óptimo posible.
- *Bajo licencia abierta*: Debe contar con una licencia oficial de uso abierto que así los defina.

Este tipo de fuentes tienen su origen inicialmente en los organismos oficiales y administraciones públicas (bancos centrales, institutos de estadística y similares, entes públicos como ayuntamientos, comunidades, etc.), que bajo un criterio de transparencia, y con la motivación de tener informados a la población, ofrece de forma abierta los datos para que los usuarios puedan realizar análisis y valoraciones de los mismos.

No obstante, este tipo de iniciativas se ha ido haciendo extensible a otros muchos ámbitos de la sociedad: sector social, económico, etc. Cada vez son más las iniciativas encaminadas a crear bases de datos abiertas que tienen una doble intencionalidad, por un lado ofrecer el uso de esos datos a quienes pudieran estar interesados, y por otra parte, favorecer el desarrollo de estudios e investigaciones en materias y sectores concretos que, evidentemente, también trasladarán valor añadido a los aportantes de la información.

3.1.2. Small Data y Smart Data

El desarrollo y evolución del *Big Data* ha traído asociado la aparición de otros conceptos relacionados como pueden ser el *Small Data* o el *Smart Data*.

Por *Small Data* entendemos, a diferencia del *Big Data* o datos masivos, aquellos pequeños volúmenes de datos que se generan en nuestro entorno, en un ámbito más local, a los que tenemos más fácil acceso. Podría entenderse que la unión de los diferentes *Small Data* generados por los diferentes agentes daría como resultado un *Big Data*.

La afirmación anterior es más que obvia, si bien el verdadero valor del *Small Data* reside en la capacidad de dar respuesta a la pregunta que antes planteábamos: *¿qué buscar?* Si los *Small Data* son aquellos que se encuentran en nuestro entorno y, por ello, aquellas variables que nos afectan directamente, se puede extrapolar dicho estudio a un conjunto de datos mayor (*Big Data*).

Adicionalmente, y sin mayores pretensiones que poder desentrañar lo que ocurre en el ámbito local, los *Small Data* pueden ser tratados de igual forma que los amplios volúmenes de datos, puesto que se crean de manera continuada y en tiempo real, si bien el acceso a los mismos es más fácil y su coste, si se establece un proceso coherente de obtención, es más bajo.

Por su parte, el *Smart Data* es una acotación lógica del concepto de *Big Data*, al que en muchos casos se ha dado amplitud tal vez excesiva, generalizándolo a todo proceso de gestión y tratamiento de datos. No debemos olvidar que el objeto del tratamiento de los datos es encontrar soluciones a problemas planteados. En ocasiones, para resolver estos problemas no es necesario un gran volumen de datos, sino solamente los datos necesarios, los que algunos han llamado "datos inteligentes" o, dicho de otro modo, solo aquellas variables que realmente aportan algo a mi problema.

3.1.3. No es oro todo lo que reluce

Aunque al *Big Data*, en su concepción más genérica como la punta de lanza de una nueva realidad, se le atribuyen importantes ventajas en todos los ámbitos, su crecimiento exponencial ha ocultado algunos aspectos que deberían tenerse en consideración. Ya hemos señalado algunas limitaciones que muchas veces obviamos al referirnos a este concepto, tales como *¿qué buscar?* o la limitación y coste de acceso a esa ingente cantidad de datos que se generan diariamente.

Algunos autores detectan una consecuencia de esta realidad, que definen como *la tiranía de los datos*. Es indudable las oportunidades que ofrece el estudio de grandes datos aplicando técnicas de *Data Mining* para obtener importantes y significativas conclusiones, pero autores como Mayer-Schönberger y Cukier (2013) han hablado de ciertos riesgos asociados al fetichismo de los datos.

Disponer de datos y conocer las técnicas de tratamiento no asegura alcanzar conclusiones aceptables, sobre todo si no se aplica el sentido común a las herramientas de *Data Mining* de que se dispone. Un mal uso de éstas generaría un efecto contrario. Por otra parte, aun cuando los análisis fueran precisos, no debemos olvidar que eso solo sería una parte del total de la información con la que contamos, ya que debemos considerar otras fuentes más cualitativas o humanísticas. Y, finalmente, la información solo es un pilar sobre el que se debe asentar la toma de decisiones. Por lo tanto, los datos y su tratamiento es únicamente una parte del conjunto y, en ningún caso, se pueden convertir en el centro del todo.

Por otra parte, también debemos considerar al aplicar técnicas de carácter predictivo que el estudio de los datos para determinar comportamientos o patrones de conducta a partir del conjunto de datos excluiría la componente humana, lo que condicionaría el resultado.

Existen otras consideraciones sobre los datos y el origen de éstos, que caerían en el ámbito de la ética y la moral. Muchos de los datos que cada uno de nosotros generamos diariamente se vinculan al plano personal, lo que plantearía algunos dilemas sobre su uso interesado. Por ejemplo, las pulseras que registran pulsaciones, actividad deportiva, descanso, etc., en el momento que se trasladan a una aplicación pueden ser utilizadas por empresas de

seguros para determinar o cuantificar primas. O bien, por medio del móvil es fácil saber dónde ha estado un individuo en cada momento y qué lugares frecuenta. ¿Hasta qué punto la captación de datos puede franquear las líneas de privacidad de las personas?

La legislación se ha modificado para limitar algunos aspectos de la privacidad y uso de los datos. Las agencias de protección de datos deberán ser los garantes del correcto uso de determinada información, si bien siempre quedarán aspectos que generen cierta controversia.

3.1.4. Consideraciones legales básicas

Hoy en día nadie duda que los datos sean el negocio de los negocios, puede afirmarse que la información es el motor de la economía actual, y por supuesto se ha llegado a esta aseveración gracias al desarrollo tecnológico que se ha producido en los últimos años.

Actualmente se generan muchos datos, en los epígrafes anteriores se ha citado la amplia variedad de fuentes que están capturando datos de forma casi permanente, pero lo más significativo es que en la mayoría de las ocasiones esos datos están vinculados a "personas físicas". Por ello, afirmar que los datos personales son el petróleo del siglo XXI ya no sorprende.

La evolución tecnológica, y concretamente el Big Data, ha permitido a las empresas rentabilizar los datos personales acumulados en sus sistemas informáticos durante años, que completados con la información que obtienen a través de otras fuentes como las redes sociales, les abren nuevas posibilidades de negocio. Otra realidad cierta es que los clientes cada vez son más exigentes y quieren ser tratados de forma totalmente personalizada, por lo que las empresas tienen como una de sus prioridades detectar los gustos de sus clientes y lanzar campañas publicitarias personalizadas, óptimas y eficientes; también el Big Data es el sistema idóneo para este tipo de tratamientos.

Dicho lo anterior, no se puede obviar que determinados usos de Big Data tienen una incidencia directa en las personas y pueden entrañar riesgos, sobre todo cuando la información resultante se utiliza para discriminar a determinados sectores de la población. Es por ello por lo que cualquier empresa que opte por utilizar Big Data, tiene que conocer aspectos tan importantes como son la calidad de los datos personales utilizados, la transparencia de la información que debe facilitar a las personas cuyos datos personales va a tratar o el nivel de privacidad que sus clientes están dispuestos a aceptar.

Los legisladores cuando redactaron la Constitución Española en 1978 ya vislumbraron las implicaciones que podría tener el desarrollo tecnológico sobre la privacidad de las personas, y por ello recogieron en su artículo 18 el derecho a la intimidad e inviolabilidad del domicilio entre los derechos y deberes fundamentales. Textualmente el apartado 4 tiene la siguiente redacción "La Ley limitará el uso de la informática para garantizar el honor y la intimidad personal y familiar de los ciudadanos y el pleno ejercicio de sus derechos". Por tanto, es importante recordar que el derecho a la intimidad es un derecho fundamental que debe tener en cuenta cualquier empresa que trate datos de carácter personal, con independencia de la tecnología que utilice para el tratamiento.

En 1992, siguiendo el mandato legal de la Constitución Española, se aprueba la Ley 5/1992 de Regulación del Tratamiento Automatizado de Datos de carácter personal, derogada en 1999 por la LOPD (Ley Orgánica de Protección de Datos de carácter personal). El objeto de esta legislación es garantizar y proteger, en lo que concierne al tratamiento de los datos

personales, las libertades públicas y los derechos fundamentales de las personas físicas, y especialmente de su honor e intimidad personal y familiar.

Finalmente, a partir de mayo de 2018, todas las empresas deben estar en disposición de demostrar que cumplen con el Reglamento General de Protección de Datos (Reglamento del Parlamento Europeo y del Consejo relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de éstos). El objeto del Reglamento es similar al de la anterior legislación y lo que pretende es crear un marco legal de protección de datos armonizado en toda la Unión Europea y uno de los objetivos más importantes es devolver el control al ciudadano sobre sus datos personales. A partir de ese momento quedará derogada la LOPD.

Entre las principales implicaciones de los tratamientos big data en la privacidad podemos señalar de forma muy resumida:

- El origen de los datos, es decir las fuentes de las cuales se nutre la empresa para obtener datos de carácter personal.
- Transparencia en la información, que hace referencia al derecho de información que cualquier titular de datos debe tener sobre el tratamiento, finalidad y destino que se le va a dar a sus datos personales y la obligación de la empresa como responsable del tratamiento a facilitársela.
- Consentimiento del interesado como manifestación de voluntad libre, específica, informada e inequívoca por la que el interesado acepta el tratamiento de sus datos personales.
- Calidad de los datos, aspecto importante que está muy relacionado con el origen de los mismos ya que, dependiendo del nivel de confiabilidad que ofrezcan los diferentes orígenes de datos, la calidad de los datos personales puede verse comprometida.
- Derechos de los interesados, incluyendo acceso, rectificación, oposición y decisiones individuales automatizadas, supresión, limitación del tratamiento, y portabilidad de los datos.

3.2. APLICACIONES

Las consideraciones sobre un mal entendimiento del *Big Data* no ocultan la transformación de la realidad y de los procesos que abren nuevas puertas al conocimiento y a la actuación en multitud de campos. Tal es así, que el *Big Data* se ha colado en todos los ámbitos: sociales, económicos y personales.

Su aplicación en el **sector empresarial**, sin pretender ser exhaustivos, se extiende en todos los sectores empresariales. En temas de *finanzas y seguros*, para determinar patrones de compra y clientes, ingresos medios, gestión de riesgos, entre otros; en *marketing* tiene enormes aplicaciones en estudios de mercado, determinación de necesidades, fijación de precios, clientización, entre otros; igualmente en el *turismo* y las *smart cities*, para determinar las preferencias de los consumidores y aquellos aspectos que facilitan la vida diaria en las urbes. También se aplica en la gestión del *transporte*, bien de mercancías o personas, para optimizar estos procesos y generar mayores beneficios. En general, tiene reflejo en todas las áreas de la gestión empresarial, *planificación, organización y recursos humanos*.

Aunque hemos puesto el acento en la utilidad del análisis de grandes datos en el ámbito empresarial, sus aplicaciones exceden mucho más allá, ofreciendo importantes oportunidades y ventajas en otros campos. A modo de ejemplo, recogemos a continuación algunas de las posibilidades.

Es habitual encontrar trabajos e investigaciones del uso del *Big Data* en el área de la **salud**. El cuerpo humano es una gran fuente de datos y el estudio de millones de sujetos, así como de las diferentes pruebas que se realizan (tratamiento de enfermedades, análisis de sangre, radiografías, etc.), juega un papel muy importante en la evolución del conocimiento médico (en 2015 un hospital médico estadounidense gestionaba al año 665 terabyte frente a los 168 de 2010). Las iniciativas en este ámbito son tanto de origen privado (muchas de ellas a través de aplicaciones móviles, como el primer glucómetro de *Telcare*, o la Web de *Quantified Self*) como público, pues también permite ahorrar costes al sistema sanitario como puso de manifiesto el informe de McKinsey Global Institute de 2011. Pero además, puede ser de gran ayuda para el seguimiento de epidemias como demostró la aplicación de Google Flu Trends en 2009, encontrando una relación entre las personas que realizaban búsquedas de síntomas de gripe con aquellas que realmente los tenían.

El *Big Data* también tiene una amplia utilidad en el campo de las ciencias sociales, como en la **sociología y psicología**, estableciendo patrones de comportamiento y conducta que permitan establecer teorías y modelos terapéuticos.

En el **ámbito social y de desarrollo** también se están realizando interesantes proyectos, como por ejemplo se ha hecho en África, donde el 50% de la población tiene móvil y entre un 10% y un 20% no lo poseen, pero tienen acceso al dispositivo pudiendo de esta manera determinar patrones de movimiento de personas y modelizar la propagación de enfermedades. Esta práctica se realizó concretamente en Kenia localizando ubicaciones donde la probabilidad de propagación de la malaria era mayor.

Su uso no se limita a cuestiones profesionales o técnicas, en la **educación e investigación** las posibilidades que ofrece son notables. En lo referente a la educación, cada vez es más habitual el uso de TIC en el proceso docente que se sustentan en plataformas educativas (como aLF en la UNED o Moodle), por medio de los cuales se obtienen grandes volúmenes de datos sobre los estudiantes, su evolución, progreso, conocimientos adquiridos y evaluación final. El tratamiento y análisis de estos datos, al objeto de predecir el éxito o fracaso de un estudiante, ha dado pie a lo que se conoce como *Analítica del Aprendizaje* (Ferguson, 2014). Por su parte, en lo que a la *investigación* se refiere, el tratamiento masivo de amplios volúmenes de datos hospedados en diferentes servidores, permite ampliar las muestras de estudio, realizando con ello trabajos de investigación más amplios y profundos, mejorando el «Estado del Arte» (por medio de *Text Mining* o *Web Mining* y su derivada del análisis de sentimientos y nuevas tesis basadas en modelos más robustos y sólidos).

Un ámbito en el que cada vez está tomando más importancia la gestión y tratamiento de múltiples bases de datos es en la **Administración y seguridad**. Existen numerosos proyectos en entidades, instituciones y administraciones de carácter público en las que se aplica *Big Data* al objeto de ofrecer un valor a la sociedad. A modo de ejemplo, podemos señalar al INE (Instituto Nacional de Estadística) que está desarrollando proyectos a través del móvil para estimar variables del Censo de Población y en relación al turismo los datos de precios de webs de turoperadores para trasladarlos al IPC. De igual forma, en temas de seguridad y *Law Enforcement* (fuerzas del orden), se han desarrollado diversos proyectos: en 2002 el FBI, en su batalla contra el terrorismo, anunció que comenzarían a introducirse en bases de datos comerciales lo que permitiría acceder a multitud de información de cada

persona (si es fumador, talla, hábitos de compra, localización, etc.), y de manera paralela está desarrollando el programa FAST (*Future Attribute Screening Technology*), que permita identificar posibles terroristas monitorizando sus constantes vitales, lenguaje corporal y otros patrones fisiológicos; o, el análisis realizado en la ciudad de Los Ángeles para detectar qué calles y qué grupos o individuos son más propensos a delinquir.

El *Big Data* permite conocer mejor el tráfico de las autopistas, el uso de la energía doméstica, los fenómenos meteorológicos, el comportamiento de las personas y, en general, el universo que nos rodea.

3.3. PRINCIPALES HERRAMIENTAS

Big Data requiere de un conjunto de herramientas y software que permita almacenar, gestionar, tratar y analizar volúmenes enormes de datos, por medio de multitud de tareas que no pueden realizarse únicamente en un equipo.

En este epígrafe se muestra una perspectiva general, simplificando su presentación para facilitar su comprensión a todos aquellos estudiantes que desconocen estos conceptos.

De manera sencilla, podemos señalar que para tratar datos masivos se requiere del uso de un conjunto de servidores al mismo tiempo, diferenciando así varios niveles de herramientas. En un primer nivel estarían aquellas aplicaciones que nos permitan trabajar con varios servidores a la vez. Por debajo de ellas, se han desarrollado softwares específicos y un ecosistema de aplicaciones para gestionar las tareas de trabajo, cuyo resultado final será el análisis de los datos. En ese primer nivel, los dos *framework*¹¹ más conocidos son **Hadoop** y **Spark**. En segundo y posteriores niveles, dispondremos de otras aplicaciones y programas que se combinan con los dos *framework* para adaptarnos al proyecto concreto que se esté realizando.

3.3.1. *Hadoop* y *MapReduce*

Hadoop es un proyecto de **Apache** de código libre que permite trabajar en un entorno distribuido. En otras palabras, trabaja a través de un clúster de servidores cada uno de los cuales se encarga de un proceso y, además, en ellos se almacena la información.

Es una aplicación especialmente útil para aquellos que no poseen experiencia en entornos distribuidos. El usuario manejará un único ordenador, aunque en esencia se esté trabajando con un clúster de servidores, y será el propio **Hadoop** el que gestione aspectos técnicos tales como la penalización de tareas, la administración de procesos, el balanceo de carga y la tolerancia a fallos.

Hadoop se desarrolló sobre dos módulos:

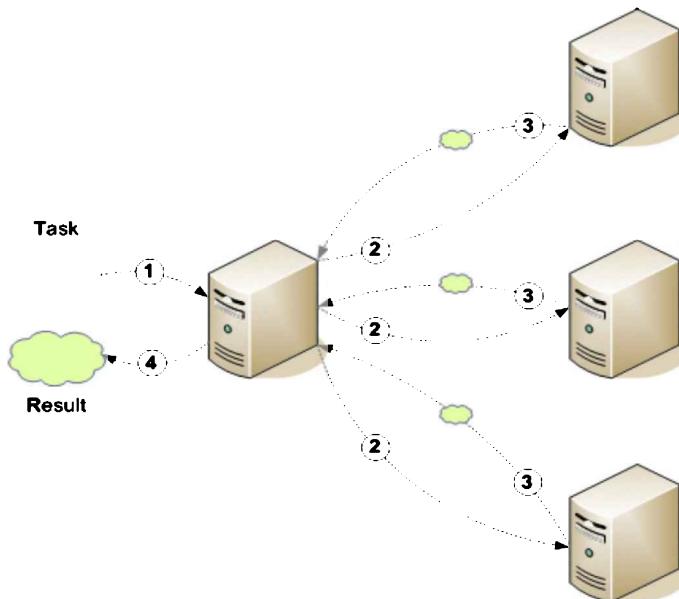
- **Hadoop Distributed File System (HDFS)**: el sistema de ficheros sobre el que se ejecutan la mayoría de las herramientas que conforman el ecosistema *Hadoop*. El HDFS tienen dos elementos fundamentales en la arquitectura: el *NameNode* y los *DataNode*. El *NameNode* se encuentra en el servidor y ordenador maestro encargándose de ubicar entre los diferentes servidores esclavos los datos almacenados. Por su parte, los *DataNode* se encuentran en los servidores esclavos y

¹¹ Son aplicaciones genéricas o estructuras de soporte incompletas a las que se les añade otro software para el desarrollo de un proyecto, como programas de soporte, librerías, software específico para desarrollar o unir componentes, etc.

se encargan de almacenar los datos. *Hadoop* se caracteriza por la duplicidad de los datos, es decir, estos se almacenan en más de un nodo, evitando que el mal funcionamiento de un servidor en particular provoque una pérdida de información.

- **MapReduce:** Es un *framework* de programación para el desarrollo de aplicaciones y algoritmos. Trabaja en paralelo bajo el mismo sistema de maestro-esclavos para encontrar respuesta a los análisis realizados sobre los datos. El motor *MapReduce* consiste en un planificador de trabajos denominado *JobTracker*, el cual gestiona, monitoriza y distribuye la carga de trabajo en los nodos. Por otra parte, el *TaskTracker*, que se ejecuta en los nodos del clúster, realiza el proceso sobre el bloque de trabajo que contenga.

FIGURA 8. DIAGRAMA DE HADOOP



Fuente: <https://hadoop.apache.org/>

Al ser *Hadoop* un *framework* de código abierto y licencia libre, se han desarrollado proyectos y herramientas que han permitido crear todo un *ecosistema* (incluye más de 150 proyectos) en el cual podemos encontrar diferentes softwares, de los cuales, los más conocidos, se agrupan a continuación según su utilidad:

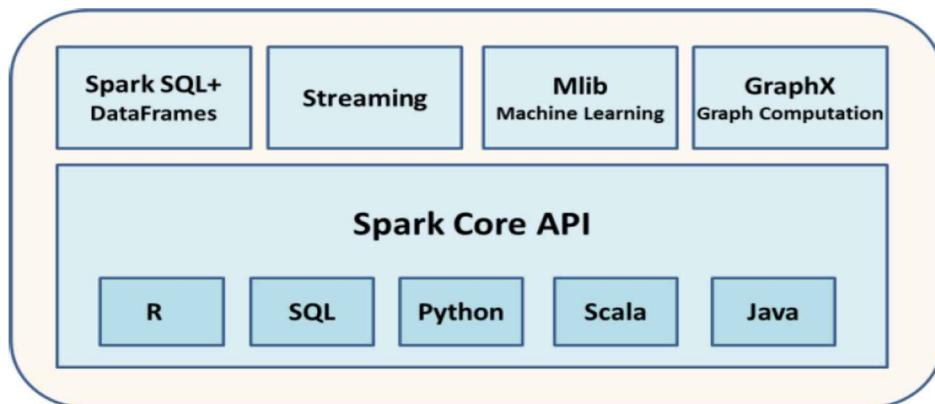
- Captura y manipulación de datos: Herramientas tales como Chukwa, Flume, Sqoop, Uima y Lucene.
- Almacenamiento: Dentro de este grupo incluimos a Hive y HBase.
- Tratamiento de datos: Para esta tarea podemos hacer uso de Mahout, Pig, oozie, y Jaql.
- Administración: Contamos con Zooquiper, Avro y Hue.

3.3.2. *Spark*

Posteriormente a la aparición de *Hadoop*, Apache desarrolló *Spark* con el fin de solucionar aquellas cuestiones que se cuestionaban a su predecesor. *Spark* es *framework* de licencia libre que en su comparativa con *Hadoop* ha demostrado ser más rápido (en algunas circunstancias hasta 100 veces más rápido) pero requiere más memoria (si los datos analizados no caben en la memoria no hay diferencia significativa entre ambos), pero a su vez, al ser mayor el rendimiento le permite el procesamiento en tiempo real y por lotes,

mientras que *Hadoop* solo es por lotes. Respecto a la facilidad de uso *Spark* muestra una ventaja significativa, gracias al uso de sus API's (del inglés, *Application Programming Interface*) de alto nivel (*R*, *SQL*, *Python*, *Scala* y *Java*).

FIGURA 9. ECOSISTEMA SPARK



Fuente: <http://spark.apache.org>

Spark no es una versión modificada de *Hadoop*, aunque utiliza el HDFS tiene su propio sistema de gestión de clúster. Su motor de funcionamiento, conocido como *Core de Spark*, permite trabajar sobre memoria y realizar procesos en tiempo real, y soporta a todas las API's. Si bien, podría utilizar el *MapReduce* o incluso con *Yarn*. De igual forma, cuenta con un conjunto de herramientas entre las que se incluyen:

- *Spark SQL*: Para el procesamiento de datos estructurados.
- *MLib*: Es una biblioteca de aprendizaje automático.
- *Graphx*: Es un motor de cálculo para el procesamiento gráfico.
- *Spark Streaming*: Para procesos de *streaming* en directo y tiempo real.

4. PROGRAMAS DE SOFTWARE MÁS UTILIZADOS

En este epígrafe ofrecemos una explicación breve de los programas informáticos utilizados en Data Science, donde hemos incluido aquellos que se emplean de forma general por casi todos los científicos de datos, con licencia gratuita, en Big Data y Data Science y, un único programa comercial, el IBM-SPSS, dada su gran utilización por los investigadores en el ámbito social y económico. Además, este software incluye, desde hace ya unos años la posibilidad de ejecutar script y de integrar los lenguajes R y Python con lo que amplía su capacidad de gestión de los datos.

Ambos lenguajes son los más acogidos por las plataformas que dan soporte al Big Data a la vez que también son ampliamente utilizados en la minería de datos. El R, es un lenguaje especializado en todos los ámbitos de la estadística y en la presentación de resultados y que es ampliamente empleado por la mayor parte de científicos del mundo universitario, en tanto que el lenguaje de programación orientado a objetos denominado Python, se ha incorporado al trabajo científico por ingenieros e informáticos que trabajan en Big Data. Ambos programas interactúan entre ellos y se complementan; se observa que cada vez más investigadores utilizan ambos softwares.

Cabe citar también el programa WEKA, que es una extensa colección de algoritmos escritos en JAVA, de cómoda utilización a través de menús, y de amplia difusión en todos los ámbitos

de la investigación. Integra los lenguajes R y Python, e interactúa con las herramientas de Big Data: Hadoop y Spark.

Scala y Julia son otros dos softwares a considerar y de más reciente creación, por lo que recogen, en su forma de programar, las funciones y la experiencia de los que les han precedido, además de que están ampliamente respaldados por la comunidad científica y que cada vez son más preferidos en el desarrollo del Big Data y en la implementación de rutinas y de los algoritmos más novedosos que van apareciendo.

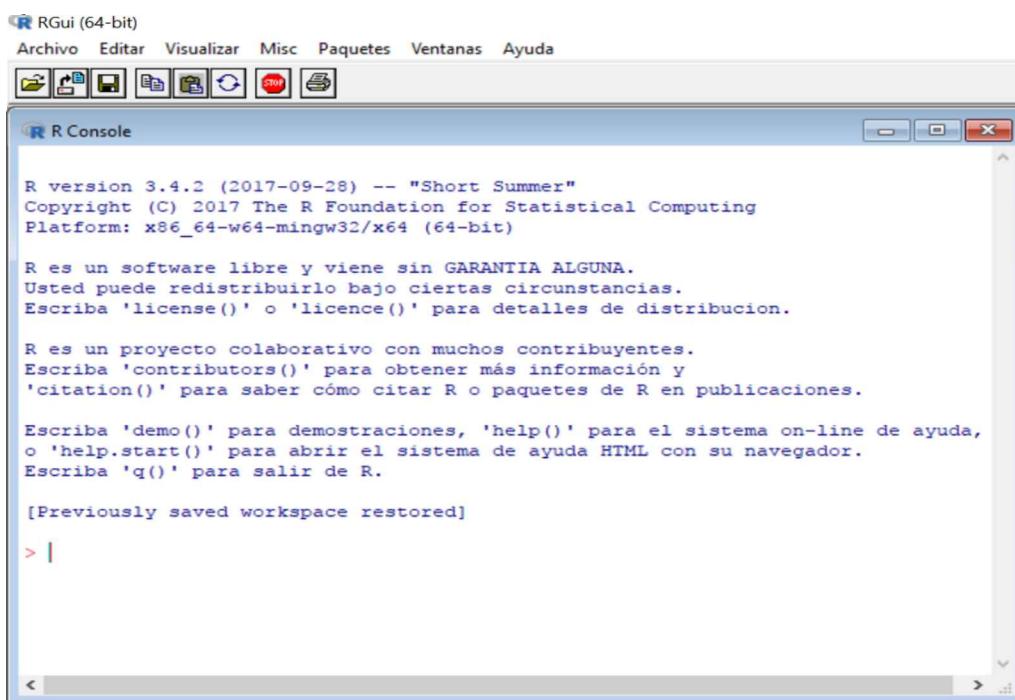
4.1. R y RSTUDIO

R es un entorno especialmente diseñado para el tratamiento de datos, cálculo y desarrollo gráfico. Permite trabajar con facilidad con vectores y matrices y ofrece diversas herramientas para el análisis de datos.

R es una implementación *open-source* del lenguaje S (Bell Labs -principios de los 90), que también es la base del sistema S-Plus (entorno comercial). R y S-Plus aún comparten una gran mayoría de código e instrucciones, si bien R es software libre, gratuito en donde los usuarios disponen de libertad para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De hecho, R dispone de una comunidad de desarrolladores/usuarios, que se dedican constantemente a la mejora y a la ampliación de las funcionalidades y capacidades del programa. En la web <http://www.r-project.org/> se encuentra disponible toda la información acerca de R. La instalación de R se realiza a través de la CRAN (*ComprehensiveR Archive Network*): <http://cran.r-project.org>

Una vez se instala R, se accede a esta consola:

FIGURA 10. CONSOLA R



Fuente: R

Actualmente R se distribuye para los siguientes Sistemas Operativos:

- Windows: entorno gráfico.
- Linux (Debian/Mandrake/SuSe/RedHat/VineLinux).
- MacOSX.
- Código fuente: ampliación a sistemas Unix.

Las funciones de R se agrupan en paquetes (packages, libraries), solo las funciones más habituales se incluyen por defecto en la distribución de R, el resto se encuentran disponibles en la Comprehensive R Archive Network (CRAN).

4.1.1. Consola R-Studio

RStudio es una interfaz que permite acceder de manera sencilla a toda la potencia de R. Para utilizar RStudio se requiere haber instalado R previamente. Al igual que R-project, RStudio es software libre.

El objetivo de los creadores de RStudio es desarrollar una herramienta potente que soporte los procedimientos y las técnicas requeridas para realizar análisis de calidad y dignos de confianza. Al mismo tiempo, pretenden que RStudio sea tan sencillo e intuitivo y proporcione un entorno de trabajo amigable, tanto para los ya experimentados como para los nuevos usuarios.

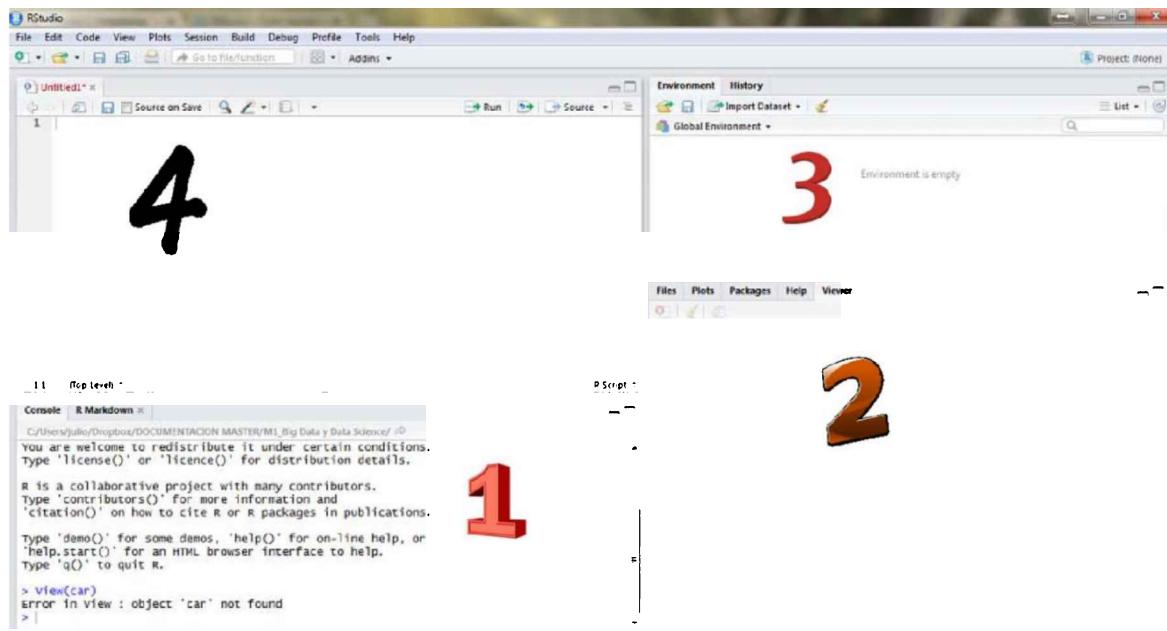
FIGURA 11. WEB RSTUDIO



La instalación de RStudio se puede realizar desde la página oficial del programa <http://www.rstudio.org>.

De manera general y simplificada, el **interface** del RStudio presenta una barra del menú principal en la parte superior y cuatro pantallas en las que se divide el resto.

FIGURA 12. ESTRUCTURA DE RSTUDIO



Fuente: RStudio

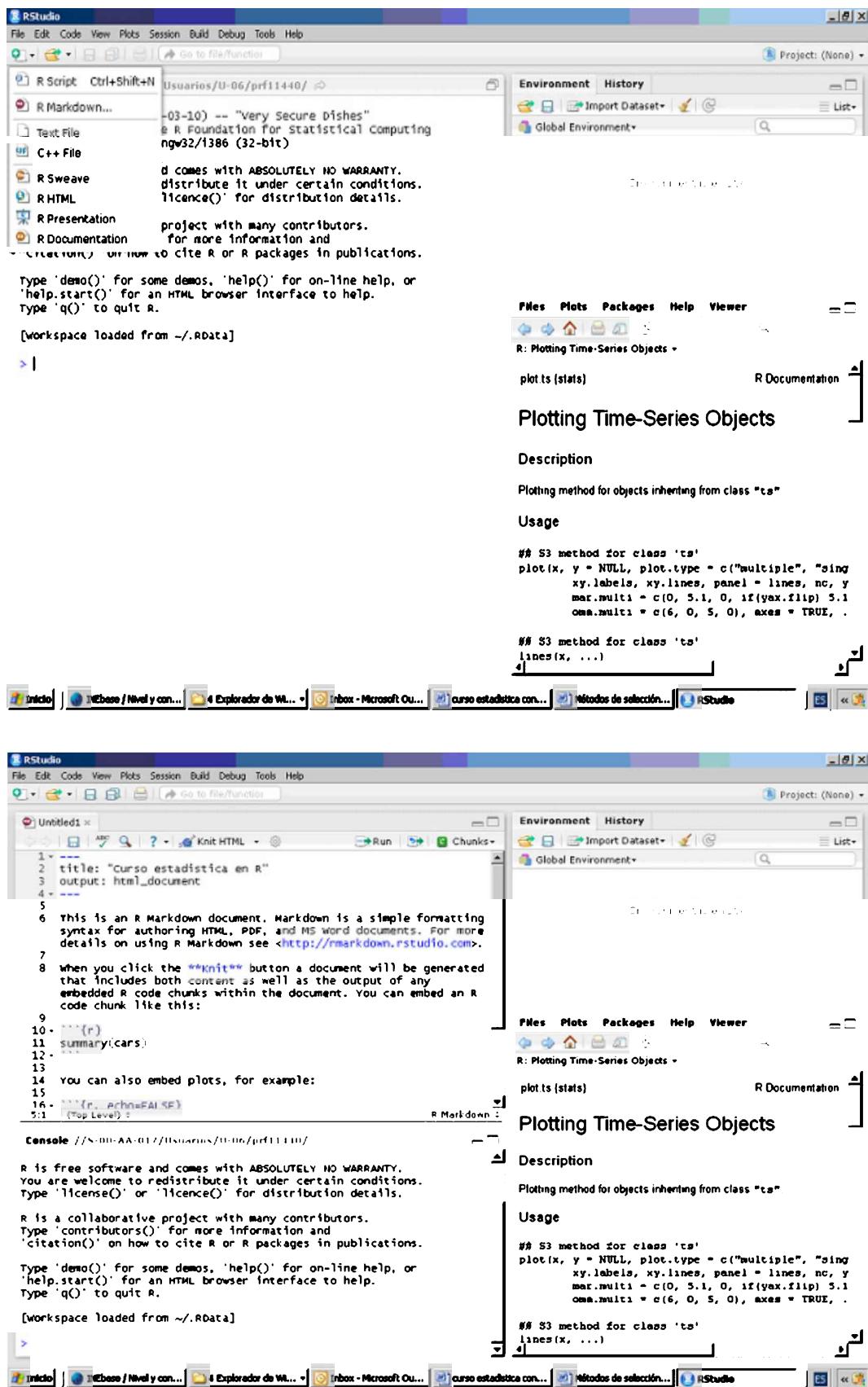
1. Es la **Consola** o interfaz de R, y por lo tanto es el espacio de trabajo, en el que se incluirán los comandos, las órdenes, se cargarán las librerías, etc...
2. En la pantalla inferior de la derecha se muestra el directorio de trabajo, los gráficos que se van generando, cargar e instalar librerías, ayuda y un visor HTML.
3. En la pantalla superior de la derecha se muestra el historial de objetos almacenados en la memoria, los comandos ejecutados, así como la posibilidad de cargar datos, entre otras cuestiones.
4. Finalmente, la ventana **Edición**. Aunque se puede desarrollar todo el trabajo en la consola, esta no es la forma más eficiente de trabajar en RStudio. Es muy útil tener un entorno donde manipular (corregir, repetir, guardar, etc.) las entradas de código que solicitemos a R. Para ejecutar lo que en ella se incluya se acciona *Run*. Es en esta ventana se pueden elaborar informes o textos en R Markdown.

4.1.2. R Markdown

Markdown es un lenguaje de marcado ligero creado por John Gruber que trata de conseguir la máxima legibilidad y facilidad de publicación tanto en su forma de entrada como de salida, inspirándose en muchas convenciones existentes para marcar mensajes de correo electrónico usando texto plano.

Para crear un documento Markdown desde la consola R-Studio, en el menú file seleccionamos **R Markdown**:

FIGURA 13. ABRIR EL R MARKDOWN DESDE LA CONSOLA RSTUDIO



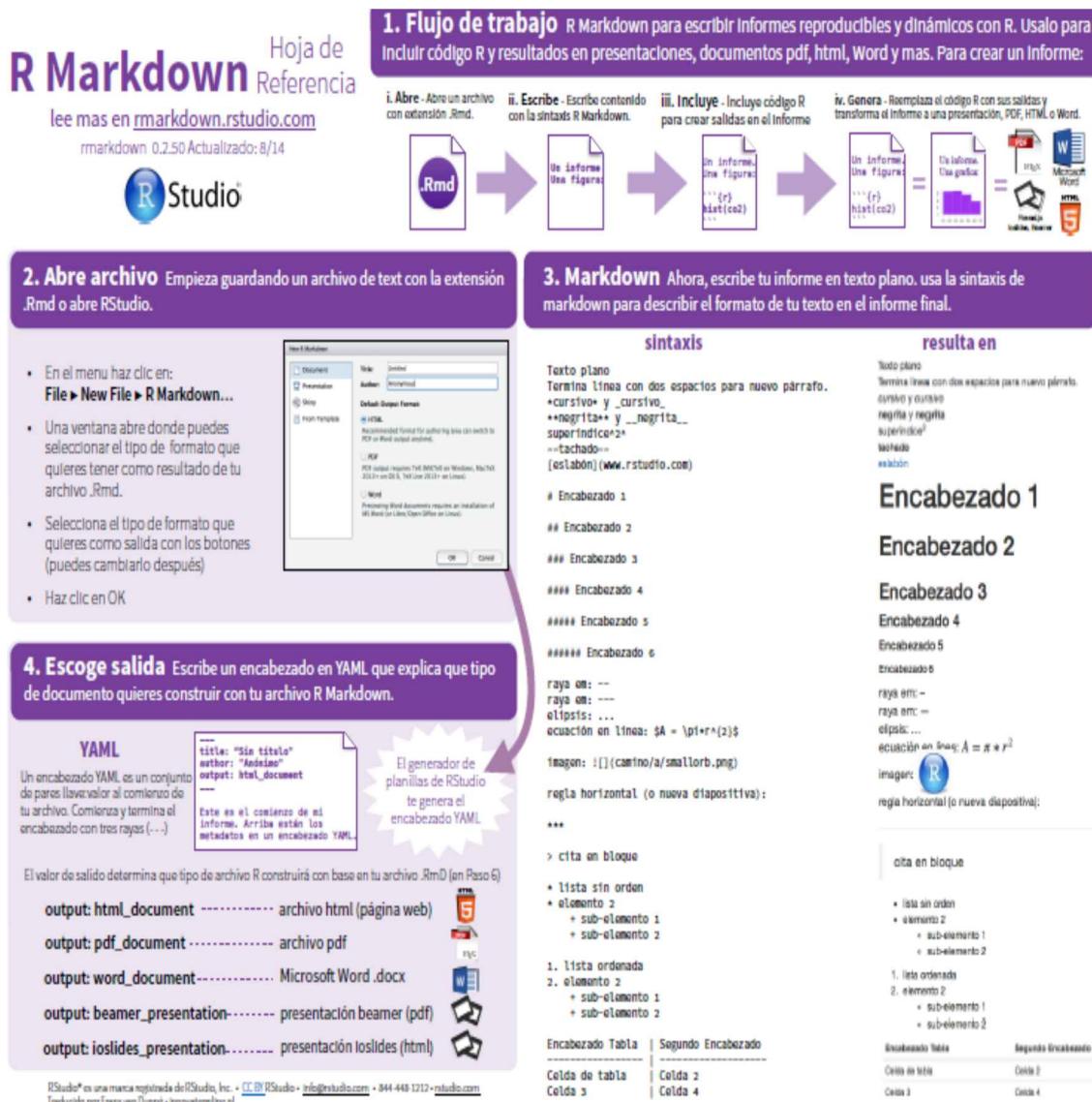
Fuente: RStudio

Markdown es un sistema muy flexible para escribir, copiar y pegar textos. Los scripts de R se ejecutan desde un chunk (Ctrl+Alt+I), o desde el menú de Markdown. Los chunk se pueden mostrar junto a sus resultados en el documento final, mostrar solo sus resultados, o dejarlos sin ejecución.

R Markdown genera documentos pdf, word o html, en la opción de menú: Knit.

El funcionamiento básico de R Markdown se ilustra a continuación:

FIGURA 14. ESQUEMA BÁSICO DEL R MARKDOWN



5. Incluye código Usa sintaxis de `knitr` para incluir código R en tu informe. R correrá el código e incluirá los resultados cuando generas el documento.

código incrustado
Surround code with back ticks and r.
R replaces inline code with its results.

pedazos de código
comienza un trozo (chunk) con ````{r}`.
Termina un trozo con `---`.

opciones para mostrar
Usa las opciones de `knitr` para cambiar el formato de un trozo.
Pon las opciones entre llaves encima del trozo correspondiente.

6. Genera usa tu archivo .Rmd como plantilla para generar un informe terminado.

Genera tu informe en dos maneras

1. Corre `rmarkdown::render("caminio/a/archivo")`
2. Haz clic en el botón `knit HTML` en la parte de arriba de la ventana de RStudio scripts

Cuando generas un informe,

- ejecutará cada trozo de código incrustado en el documento e incluirá los resultados
- construirá una nueva versión de tu informe en el formato que has indicado
- abre una prevista del archivo de salida en la ventana viewer
- guarda el archivo de salida en tu carpeta de trabajo

7. Documentos interactivos Convierte tu informe en un documento interactivo Shiny en 3 pasos

1. Añade `runtime: shiny` a encabezado YAML
2. En los trozos de código, añade funciones de Shiny `input` para incrustar widgets. Añade funciones `Shiny render` para salidas reactivas
3. Render con `rmarkdown::run` or click Run Document in RStudio

* Nota: tu informe será un app de Shiny, esto significa que tienes que seleccionar un formato de salida html, como `html_document` (para informe interactivo) o `ioslides_presentation` (para presentación interactiva).

8. Pública Comparte tu informe en línea

Rpubs.com
comparte documentos no-interactivos en el sitio de publicación markdown de RStudio. Sin costo.
www.rpubs.com

ShinyApps.io
aloja un documento interactivo en el servidor de RStudio. Opciones gratis y pagas.
www.shinyapps.io

9. Aprende más

Documentación y ejemplos - rmarkdown.rstudio.com
Más artículos - shiny.rstudio.com/articles
blog.rstudio.com
@rstudio

RStudio® and Shiny™ are trademarks of RStudio, Inc.
CC BY RStudio creativecommons.org/licenses/by/4.0/
644-448-1212 rstudio.com
Traducido por Fran van Durmen - translateforfun.net

Fuente: <https://rmarkdown.rstudio.com/>

4.1.3. Librerías en el programa estadístico R

La verdadera y notable importancia que actualmente tiene el programa R en el Data Science y Big Data se encuentra en la potencia de algunas de sus librerías. En la instalación de R se incorporan un conjunto de librerías que, en general, nos permiten llevar a cabo un gran número de tareas. Este programa estadístico tiene registradas actualmente alrededor de 9.000 librerías. Algunas de estas utilidades nos facilitan muchas tareas y por tanto somos más eficaces al trabajar con este software en procesos de análisis estadísticos: a la hora de importar y exportar los datos de otros programas, gestionando de forma más eficaz el tiempo de los procesos o los recursos del equipo, tanto la CPU como la memoria, o distribuyendo las tareas en paralelo ganando tiempo y capacidad, o implementando mejores presentaciones de gráficos que el código de la versión base o reuniendo en una única librería conjuntos de métodos y algoritmos que nos facilitan y ayudan en los procesos de la minería de datos.

Simplemente queremos exponer y mostrar a través de breves pinceladas las librerías que entendemos son las más útiles y las más empleadas por la gran mayoría de investigadores que manejan este programa tan potente. Una recopilación por temas, así como una descripción más detallada se puede encontrar en CRAN.

FIGURA 15. LIBRERÍAS DE R AGRUPADAS POR TEMAS

CRAN Task Views

Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
ExtremeValue	Extreme Value Analysis
Finance	Empirical Finance
FunctionalData	Functional Data Analysis
Genetics	Statistical Genetics
Graphics	Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization
HighPerformanceComputing	High-Performance and Parallel Computing with R
MachineLearning	Machine Learning & Statistical Learning
MedicalImaging	Medical Image Analysis
MetaAnalysis	Meta-Analysis
Multivariate	Multivariate Statistics
NaturalLanguageProcessing	Natural Language Processing
NumericalMathematics	Numerical Mathematics
OfficialStatistics	Official Statistics & Survey Methodology
Optimization	Optimization and Mathematical Programming
Pharmacokinetics	Analysis of Pharmacokinetic Data
Phylogenetics	Phylogenetics, Especially Comparative Methods
Psychometrics	Psychometric Models and Methods
ReproducibleResearch	Reproducible Research
Robust	Robust Statistical Methods
SocialSciences	Statistics for the Social Sciences
Spatial	Analysis of Spatial Data
SpatioTemporal	Handling and Analyzing Spatio-Temporal Data
Survival	Survival Analysis
TimeSeries	Time Series Analysis
WebTechnologies	Web Technologies and Services
gR	gRaphical Models in R

To automatically install these views, the `ctv` package needs to be installed, e.g., via

```
install.packages("ctv")
```

```
library("ctv")
```

and then the views can be installed via `install.views` or `update.views` (which first assesses which of the packages are already installed and up-to-date), e.g.,

```
install.views("Econometrics")
```

or

```
update.views("Econometrics")
```

Fuente: <https://cran.r-project.org/web/views/>

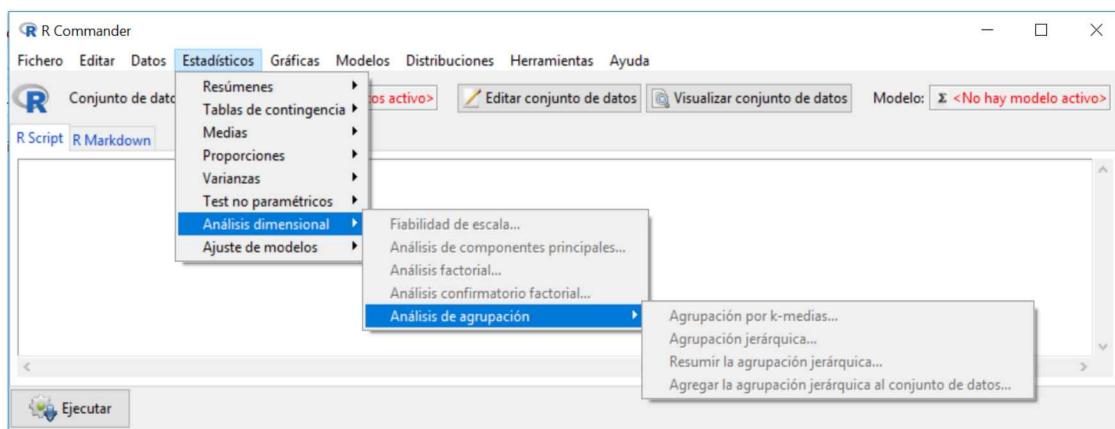
En relación con la **importación y exportación** de ficheros se dispone de algunas librerías que nos permiten comunicarnos con otros programas populares relacionados con la estadística como son el Excel, IBM-SPSS, SAS, Strata, etcétera. Algunas de estas son: **foreign**, **rio** y **xlsx**. También nos permiten conectar con otros ficheros generados por otros lenguajes de programación: json, xml, etc.

Los diferentes gráficos que ofrece el módulo base pueden ser ampliados y ganar en estética y formas por diferentes librerías como **ggplot2**, **lattice**, **Rgraphiz** y otros cuantas más. La librería **ggplot2** está basada en una estructura conocida como grammar of graphics, su idea básica es tratar de construir cada gráfico a través de unos pocos componentes que son iguales para la gran variedad de funciones y formas gráficas que contiene esta librería.

En cuanto al tratamiento y manipulación de las bases de datos existen múltiples opciones pero destacan especialmente tres librerías, utilizadas muy frecuentemente por parte de la comunidad de científicos: **dplyr**, **data.table** y **sqldf**. Las tres librerías son muy eficientes a la hora de importar datos y de realizar consultas y extracción de la información cuando las bases de datos contienen muchos registros. El paquete **dplyr** es una importante contribución ya que proporciona una gramática de verbos para la manipulación y la realización de operaciones con data frames: select, filter, arrange, rename, mutate, summarize, etc. Por su parte la librería **sqldf** nos permite implementar el lenguaje estructurado de consultas en bases de datos, SQL.

Cuando se empieza con el programa R a muchas personas les resulta muy útil activar la librería denominada **Rcmdr** (R Commander), dado que una buena parte del proceso de un proyecto estadístico se puede realizar a través de menús. Se pueden llevar a cabo la importación de ficheros de diversos formatos: texto, url, SPSS, Stata, Sas, Excel, etc. También realiza análisis básicos de los datos: resúmenes, ordenación, fusión de ficheros y agregación de variables. En la parte correspondiente al contraste de hipótesis abarca el contraste de proporciones, medias, varianzas, test no paramétricos y, una vez estimado un modelo, puede realizar contraste de residuos y otros métodos para detectar si el modelo estadístico es adecuado. Por otro lado, lleva a cabo análisis de fiabilidad de escalas, análisis clúster, factorial, de componentes principales, modelo lineal general y de regresión logística, modelos ANOVA y, por supuesto, realiza una amplia tipología de gráficos.

FIGURA 16. MENÚ DE LA LIBRERÍA R COMMANDER



Fuente: [Librería RCommander - R](#)

En la parte de Data Science en los últimos años se han incorporado diferentes paquetes en R que nos ayudan a probar de forma sencilla muchos algoritmos aplicados a las bases de datos sin tener que recurrir a una multitud de librerías. Estos algoritmos están agrupados en la parte de machine learning de la página oficial de CRAN:

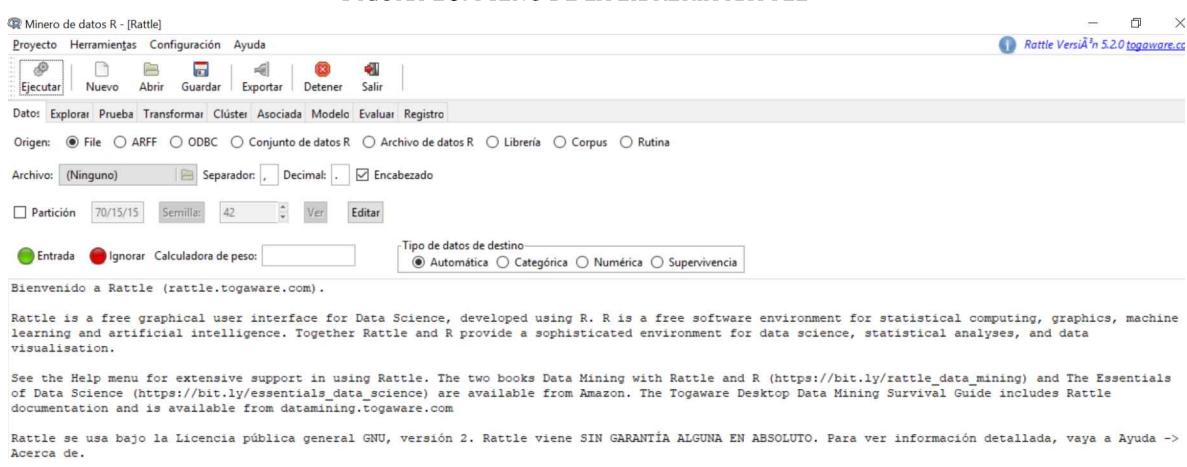
FIGURA 17. LIBRERÍAS DE R AGRUPADAS POR TEMAS

- [ahaz](#)
- [arules](#)
- [BayesTree](#)
- [biglasso](#)
- [bigRR](#)
- [bmrm](#)
- [Bonuta](#)
- [bst](#)
- [C50](#)
- [caret](#)
- [CORElearn](#)
- [CoxBoost](#)
- [Cubist](#)
- [darch](#)
- [deepnet](#)
- [e1071 \(core\)](#)
- [earth](#)
- [effects](#)
- [elasticnet](#)
- [ElemStatLearn](#)
- [evclass](#)
- [evtree](#)
- [FCNN4R](#)
- [frbs](#)
- [GAMBoost](#)
- [gmum_r](#)
- [gradDescent](#)
- [grplasso](#)
- [grpreg](#)
- [h2o](#)
- [hda](#)
- [hdi](#)
- [hdm](#)
- [ICEbox](#)
- [ipred](#)
- [kernlab \(core\)](#)
- [klaR](#)
- [lars](#)
- [lasso2](#)
- [LiblineaR](#)
- [LogicForest](#)
- [LogicReg](#)
- [LTRCTrees](#)
- [maptree](#)
- [mboost \(core\)](#)
- [mlr](#)
- [MXM](#)
- [ncvreg](#)
- [nnet \(core\)](#)
- [OneR](#)
- [opusminer](#)
- [pamr](#)
- [party](#)
- [partykit](#)
- [pdp](#)
- [penalized](#)
- [penalizedLDA](#)
- [penalizedSVM](#)
- [plotmo](#)
- [quantregForest](#)
- [randomForest \(core\)](#)
- [randomForestSRC](#)
- [ranger](#)
- [rattle](#)
- [Rborist](#)
- [RcppDL](#)
- [rda](#)
- [rdetools](#)
- [REEMtree](#)
- [relaxo](#)
- [rgenoud](#)
- [rgr](#)
- [RLT](#)
- [Rmalschains](#)
- [rminer](#)
- [rnn](#)
- [ROCR](#)
- [RoughSets](#)
- [rpart \(core\)](#)
- [RPMM](#)
- [RSNNS](#)
- [RWeka](#)
- [RXshrink](#)
- [sda](#)
- [SIS](#)
- [spa](#)
- [stabs](#)
- [SuperLearner](#)
- [sympath](#)
- [tgp](#)
- [tree](#)
- [varSelRF](#)
- [vcrpart](#)
- [wsrf](#)
- [xgboost](#)

Fuente: <https://cran.r-project.org/web/views/MachineLearning.html>

Una de las principales librerías de R en la construcción de modelos de Data Science y que se gestiona a través de menús es la librería rattle. Esta aplicación contiene los principales modelos de clasificación y de regresión, así como métodos de asociación y de agrupación. Dispone de varios formatos para la importación de los datos y presenta muchas medidas y utilidades para la contrastación y evaluación de los modelos, así como para el preprocesado de la información.

FIGURA 18. MENÚ DE LA LIBRERÍA RATTLE



Fuente: Librería rattle - R

En relación con el análisis de series temporales hay que destacar el paquete *forecast* porque ajusta y descompone de forma muy fácil la serie temporal a través de diferentes métodos: ARIMA, ARMA, AR, Exponencial Smoothing, etc.

Uno de los proyectos más utilizados en Data Mining y que engloba muchos análisis estadísticos y algoritmos de Data Science es el paquete *caret* (Classifications And REgression Training). Contiene funciones que agilizan el proceso de construcción del modelo, tanto para problemas de regresión como de clasificación. El paquete utiliza una serie de librerías determinadas de R, pero no intenta cargarlas todas, sino que las utiliza según sea necesario asumiendo que ya están instaladas. Este programa realiza de forma sencilla la división de la base de datos en entrenamiento y test, aunque también puede realizar la estimación a través de cross validation, el preprocessado de los datos, el ajuste y mejora de los modelos estimados a través de técnicas de remuestreo y también tiene incorporado procedimientos para realizar la selección de variables.

RWeka. Esta librería es una interfaz de R para conectar con el conjunto de algoritmos escritos en Java que tiene el programa de minería de datos Weka (Witten *et al.*, 2016). Este programa que contiene herramientas de preprocessamiento de datos, clasificación, regresión, agrupación, reglas de asociación y visualización aporta a R importantes procedimientos que aún no se han incorporado, lo que aumenta su capacidad de análisis.

Cuando tratamos con grandes volúmenes de datos los análisis en R están limitados, ya que solo pueden procesar un conjunto de datos que se ajuste a la capacidad de la memoria disponible del ordenador, por lo que es de vital importancia utilizar otras librerías y procedimientos que soporten los análisis con muchos registros. Algunas de estas utilidades han sido incorporadas recientemente y nos permiten conectar nuestros datos con otros servidores y llevar a cabo procesos en paralelo o gestionar la información en grandes sistemas con mucha capacidad de memoria. Entre las librerías que nos permiten ampliar las capacidades de procesamiento de R se pueden citar a *sparkR*, *sparklyr* y *h2o*.

La librería **SparkR** (<http://spark.apache.org/docs/latest/api/R/>) proporciona una interfaz para usar Apache Spark en R, utilizando el motor de cálculo de Spark lo que permite realizar análisis de datos a gran escala desde la interfaz de R.

Esta librería de R utiliza sus propios dataframes (SparkDataFrame), permitiendo hacer operaciones similares a como se realizan en R empleando la librería dplyr, pero trabajando con grandes volúmenes de datos. SparkR también soporta machine learning usando MLlib cuyo objetivo es hacer que los algoritmos utilizados en las grandes bases de datos sean escalables y fáciles de utilizar.

Los proveedores de Rstudio en 2016 publicaron un nuevo paquete denominado sparklyr, que ofrece un interfaz entre R y Apache Spark que brinda las siguientes características:

- Tratamiento de la información de datos Spark desde dplyr y SQL (vía DBI)
- Filtrado, agregación de datos y otras operaciones de los ficheros Spark desde R.
- Interfaces para aplicar algoritmos de minería de datos a través de Spark MLlib y H2O SparklingWater.
- Extensiones para proveer interfaces con otros paquetes Spark.
- Existencia de un soporte integrado para trabajar con DataFrames dentro del IDE RStudio.

Otra importante contribución es la incorporación del software H2O que se puede ejecutar desde el paquete estadístico de R llamado ***h2o***. Este software fue desarrollado en 2011 por expertos científicos en teoría del aprendizaje estadístico y optimización matemática de la empresa H2O.ai, que se fundó en Silicon Valley. H2O permite a los investigadores ajustar múltiples modelos potenciales como parte del descubrimiento de patrones en los datos.

Se utiliza para explorar y analizar conjuntos de datos mantenidos en sistemas de computación en la nube y en el sistema de archivos distribuido Apache Hadoop, así como en los sistemas operativos convencionales Linux, macOS y Microsoft Windows. Su interfaz gráfica de usuario es compatible con los principales navegadores de internet.

A través del paquete de R, ***h2o*** se pueden ejecutar múltiples algoritmos de regresión y clasificación por ejemplo: Random Forest, Gradient Boosting, xgboost y Deep Learning.

4.2. WEKA

Weka es una amplia colección de algoritmos de Data Science implementados en lenguaje Java. Es el acrónimo de Waikato Environment for Knowledge Analysis y también es el nombre de un ave endémica australiana que aparece como logotipo. Al estar escritos los procedimientos en Java, Weka es independiente de la arquitectura donde se instale y no tiene problemas de portabilidad ya que funciona en cualquier plataforma sobre la que se haya implementado una máquina virtual Java. Dispone de licencia GPL (GNU Public License.), lo que significa que este programa es de libre distribución y difusión. Weka está diseñado como una herramienta abierta por lo que se pueden crear nuevas extensiones y añadir nuevas funcionalidades de forma sencilla.

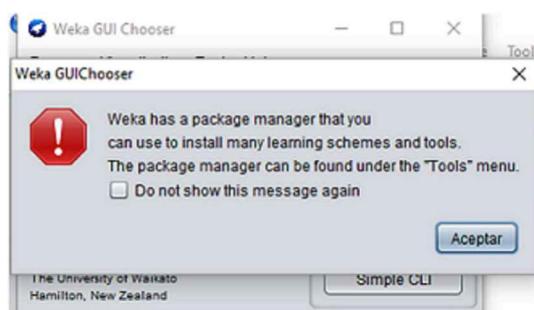
Weka ha sido desarrollado por un grupo de investigadores de la universidad de Waikato (Nueva Zelanda) y constituye un entorno que permite, de forma muy sencilla y eficaz, aplicar, analizar y evaluar las técnicas más relevantes disponibles para la minería de datos y el Big Data, sobre cualquier conjunto de datos cargado previamente por el usuario.

Este software está constituido por una serie de paquetes de código abierto con diferentes técnicas de preprocessado, clasificación y regresión, agrupamiento, asociación, series temporales, visualización, etcétera.

La instalación de la última versión estable, Weka 3.8, la podemos descargar desde la página oficial: <http://www.cs.waikato.ac.nz/ml/weka/downloading.html>

Una vez instalado el programa, el primer mensaje que aparece nos informa de que si queremos cargar otros algoritmos los podemos instalar desde la herramienta “Tools” del primer menú de la aplicación. Este mensaje aparece a partir de la versión 3.7.

FIGURA 19. MENSAJE QUE APARECE AL ABRIR WEKA



Fuente: Weka

En la primera pantalla del programa, las diferentes interfaces que podemos seleccionar son, Explorer, KnowledgeFlow, Workbench y Simple CLI.

Lo más normal, al principio, es trabajar con el Explorer dado que nos va a permitir realizar la ejecución de los diferentes algoritmos que están implementados en el programa, evidentemente sobre el fichero de entrada que carguemos.

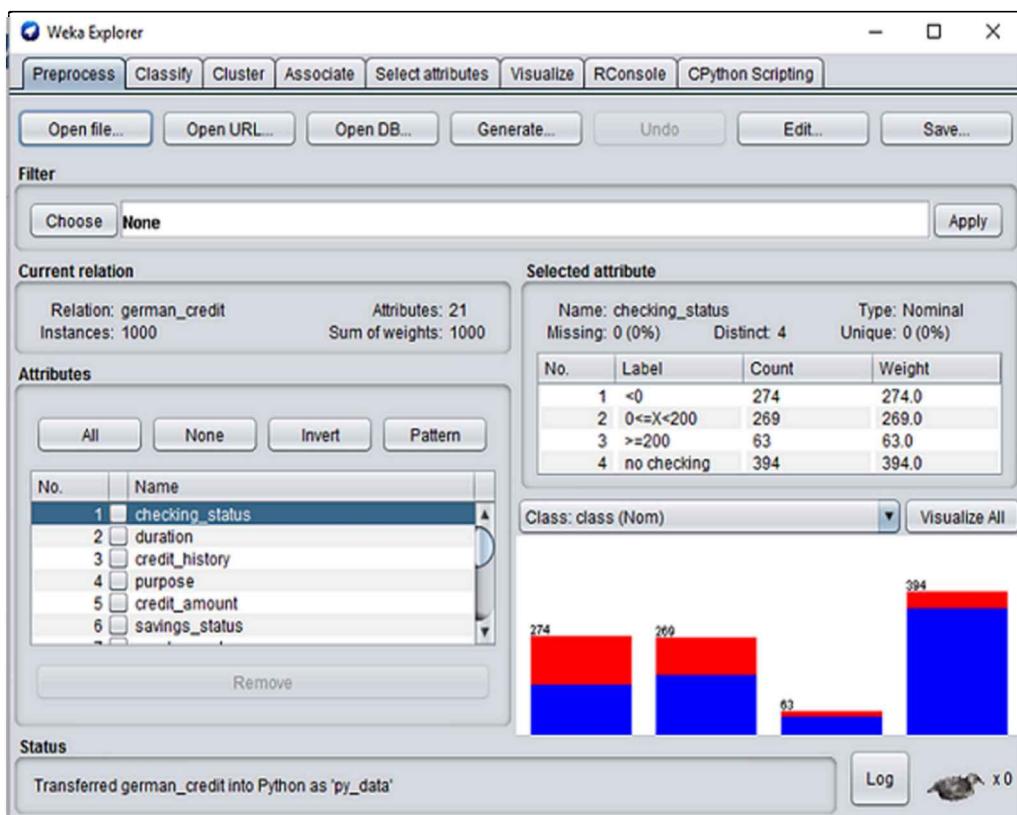
FIGURA 20. MENÚ INICIAL DEL PROGRAMA WEKA



Fuente: Weka

Para ver someramente las posibilidades de análisis que ofrece este programa vamos a empezar con el módulo Explorer. Después de activarlo, aparece la imagen de la Figura 21. Lo primero que hacemos es cargar una base de datos.

FIGURA 21. MENÚ EXPLORER



Fuente: Weka

El programa permite importar datos de múltiples formatos: csv, Excel, json, matlab, etcétera. Además de poder leer ficheros de la web, o de una base de datos.

En la pestaña **Preprocess** se encuentran las herramientas para cargar, filtrar y en general para realizar la manipulación de los datos.

En **Classify** se ubican todos los algoritmos de regresión y clasificación que nos servirán para entrenar los modelos elegidos y evaluar la precisión del método elegido. Weka habilita los algoritmos de regresión o de clasificación basándose en la última columna del fichero cargado, que es donde debemos de poner la variable dependiente o explicativa. Si es numérica habilitará los procedimientos de regresión y si es nominal los de clasificación.

La pestaña **Cluster** incluye varias técnicas para desarrollar análisis de agrupamiento o cluster.

En **Associate** podemos encontrar diversos algoritmos para desarrollar técnicas con las reglas de asociación.

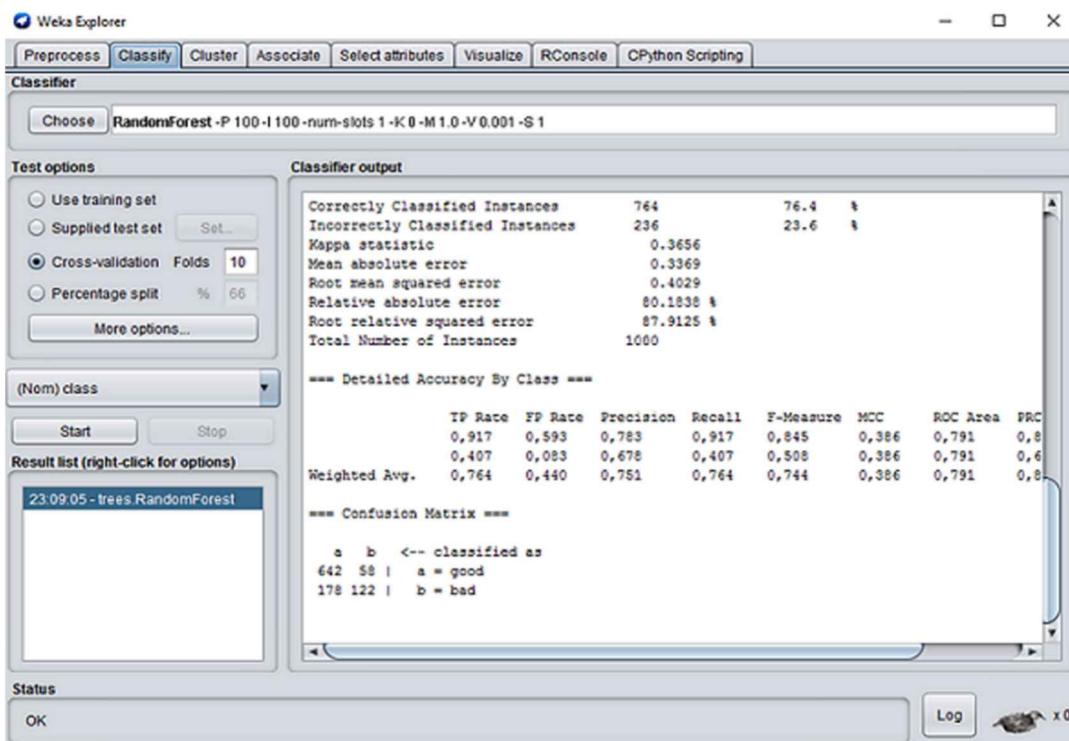
Select Attributes nos permite llevar a cabo la fase de selección de variables a través de una buena colección de evaluadores (Attribute evaluator) y métodos de búsqueda (Search Method).

Visualize nos ayuda a estudiar los datos mediante técnicas de visualización de forma interactiva, en dos o en tres dimensiones.

RConsola y Cpython Scripting conecta los programas R y Python con Weka, así que se puede escribir código o importar script tanto del programa R como en Python. Por su parte la integración desde el programa R se efectúa a través de las tres librerías siguientes: RWeka, RWekajars y Rjava.

En la Figura 22, como ejemplo visual, se observan los resultados aplicados al fichero creditg, base de datos que instala el programa en la carpeta DATA, una vez que se ha cargado en la aplicación el algoritmo RandomForest, a través del Menú/Classify de la carpeta Trees.

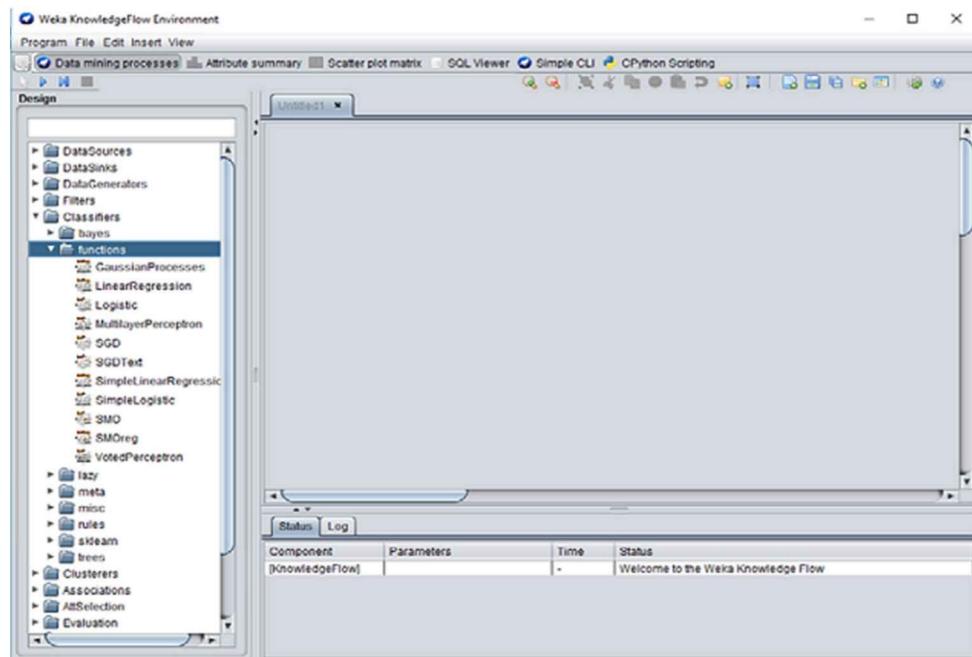
FIGURA 22. SALIDA ESTÁNDAR DE UN ALGORITMO APLICADO A LA BASE DE DATOS



Fuente: Weka

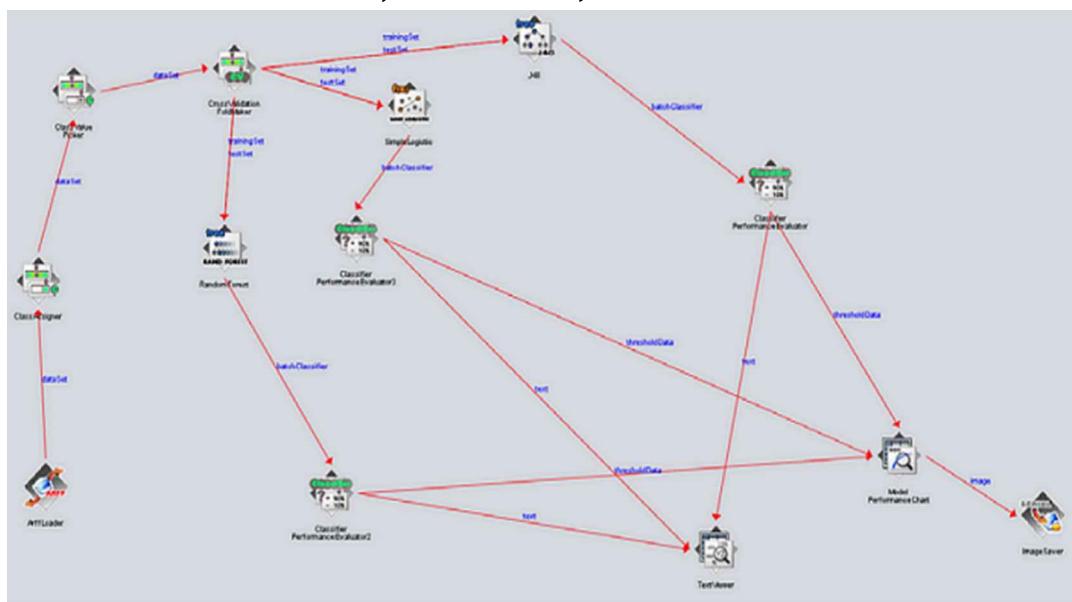
El módulo Knowledge es una interface gráfica muy cuidada y extensa que nos brinda además la posibilidad de llevar a cabo otras acciones que amplían los análisis del método Explorer. El panel de la izquierda de la Figura 23 da acceso a los elementos a configurar en el módulo, que nos permiten diseñar un experimento creando un circuito que comprenda desde la importación del fichero, la selección de variables y la posibilidad de integrar varios análisis y la salida de texto de los resultados, así como la visualización a través de procedimientos gráficos (Figura 24).

FIGURA 23. PANEL MÓDULO KNOWLEDGE FLOW



Fuente: Weka

FIGURA 24. EJEMPLO DE TRABAJO DE KNOWLEDGE FLOW



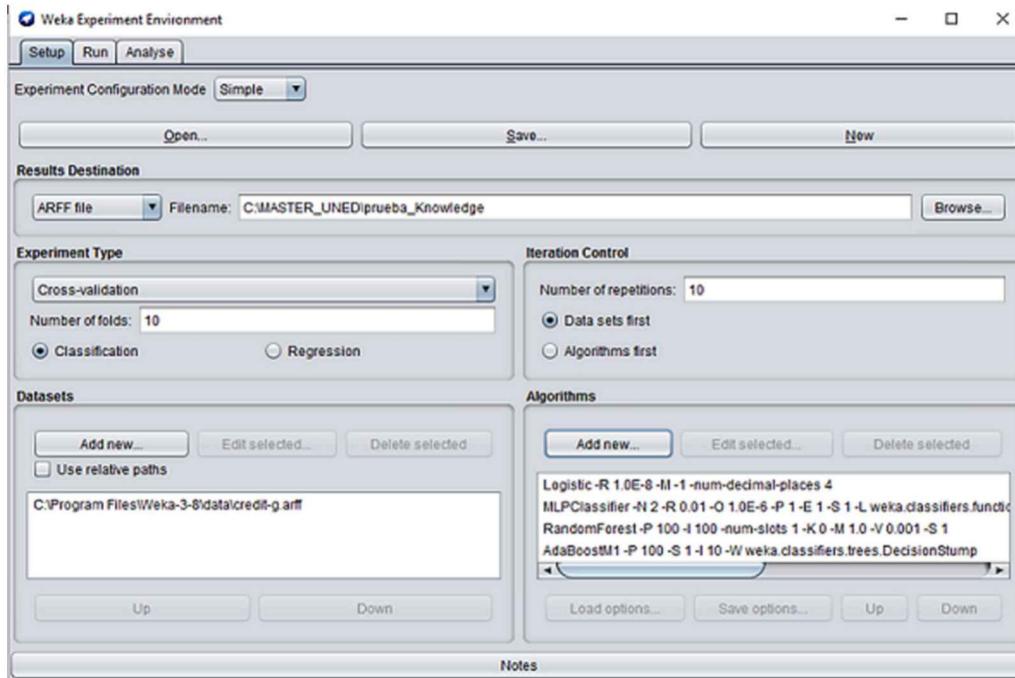
Fuente: Weka

El módulo Experimenter permite trabajar con varias bases de datos a la vez y con los algoritmos que dispone WEKA de forma conjunta. Una vez abierto este procedimiento podemos observar que este módulo dispone de tres pestañas: Setup, Run y Analyse.

En la Figura 25 se muestra la configuración de un experimento donde se ha elegido un fichero (pueden ser más) y cuatro algoritmos de clasificación que se van a aplicar a la base de datos: una regresión logística, una red neuronal (Multilayer Perceptron), un Árbol de decisión (Random Forest) y un Multiclasificador (AdaBoostM1).

En este módulo se puede elegir, al igual que en el Explorer, la forma en que queremos dividir la muestra: realizar una partición dividiendo los datos en una parte de Entrenamiento y otra de Test o si queremos llevar a cabo una validación cruzada. También nos permite repetir el experimento el número de veces que le indiquemos, la opción por defecto es 10.

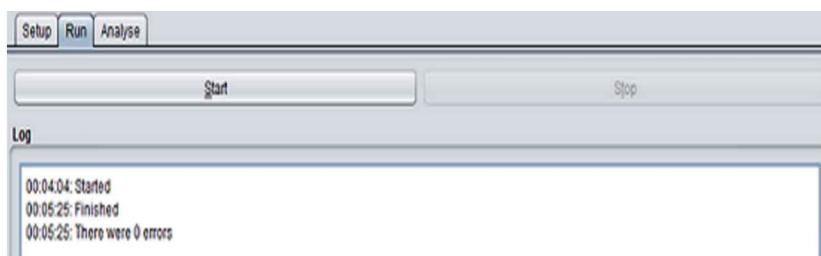
FIGURA 25. MÓDULO EXPERIMENTER



Fuente: Weka

Una vez configurado el experimento lo ejecutamos con la pestaña Run y nos irá ofreciendo información de cómo se van ejecutando los algoritmos (Figura 26).

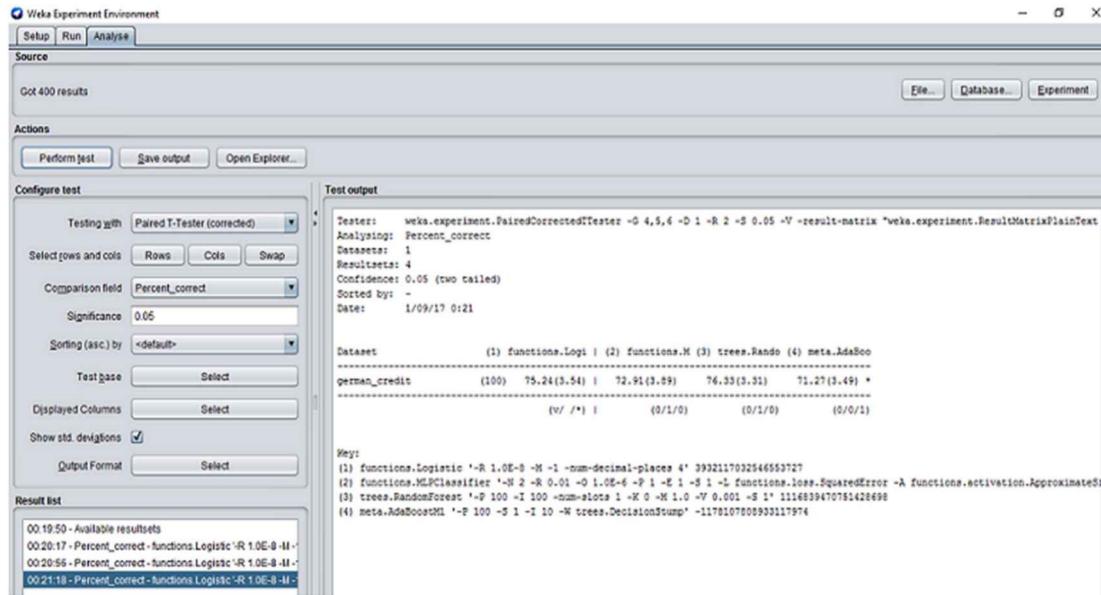
FIGURA 26. IMAGEN DE LA PESTAÑA RUN



Fuente: Weka

Una vez que ha finalizado la ejecución activamos la opción Analyse y dentro de la pantalla del menú la opción Experiment para traer los resultados a esta configuración. En este nivel podemos modificar los valores de algunos parámetros, así como observar contrastación de hipótesis, que ha estimado WEKA: porcentaje correcto o incorrecto, precisión, valor bajo la curva ROC, etcétera. En la Figura 27 se ofrecen los porcentajes estimados para los cuatro algoritmos elegidos con su correspondiente desviación estándar.

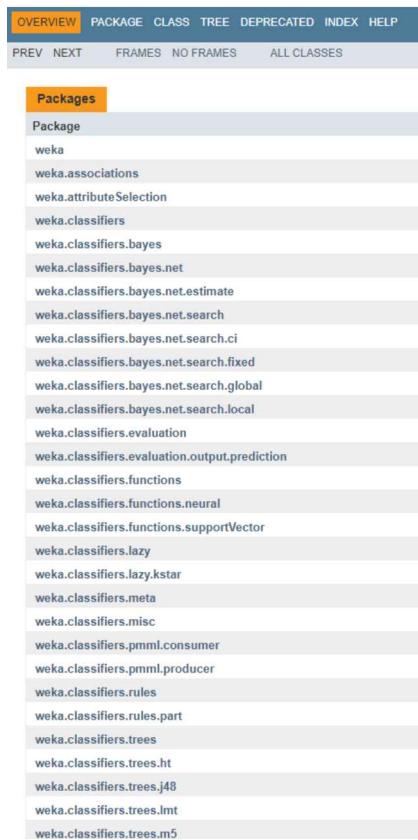
FIGURA 27. IMAGEN DE LA SALIDA ANALYSE



Fuente: Weka

Para ver los algoritmos y métodos de que dispone WEKA lo podemos efectuar de diferentes maneras, una de ellas es ver el contenido del fichero, **package-summary.html** de la carpeta donde se ha instalado el programa, normalmente, por defecto, es la siguiente: C:\Program\Files\Weka-3-8\doc\weka\. En la Figura 28 se encuentran los paquetes disponibles de Weka y en la Figura 29 se muestran las clases de Árboles de decisión y clasificación.

FIGURA 28. PAQUETES WEKA



Fuente: Weka

FIGURA 29. CLASS SUMMARY DE ÁRBOLES DE DECISIÓN

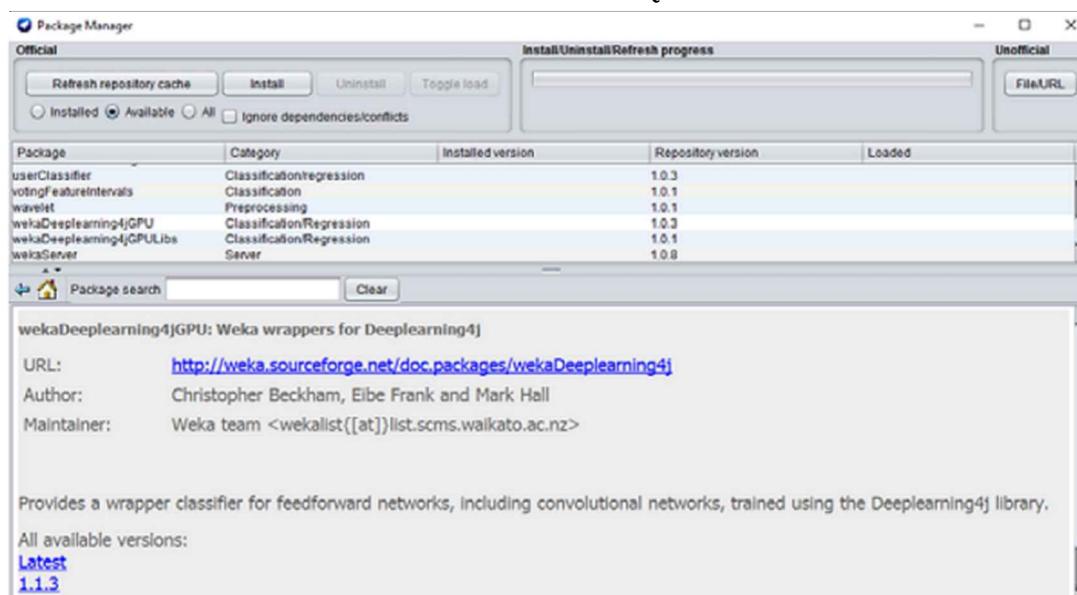
Class Summary	
Class	Description
ADTree	Class for generating an alternating decision tree.
BFTree	Class for building a best-first decision tree classifier.
DecisionStump	Class for building and using a decision stump.
FT	Classifier for building 'Functional trees', which are classification trees that could have logistic regression nodes.
Id3	Class for constructing an unpruned decision tree based on the ID3 algorithm.
J48	Class for generating a pruned or unpruned C4.5 decision tree.
J48graft	Class for generating a grafted (pruned or unpruned) C4.5 decision tree.
LADTree	Class for generating a multi-class alternating decision tree using the LogitBoost strategy.
LMT	Classifier for building 'logistic model trees', which are classification trees with logistic regression functional nodes.
M5P	M5Base.
NBTree	Class for generating a decision tree with naive Bayes classifiers at the leaves.
For more information, see:	
Ron Kohavi: Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid.	
RandomForest	Class for constructing a forest of random trees.
For more information see:	
Leo Breiman (2001).	
RandomTree	Class for constructing a tree that considers K randomly chosen attributes at each node.
REPTree	Fast decision tree learner.
SimpleCart	Class implementing minimal cost complexity pruning. Note when dealing with missing values, use "fractional instances" method instead of surrogate split method.
For more information, see:	
Leo Breiman, Jerome H.	
UserClassifier	Interactively classify through visual means.

Fuente: Weka

Muchos algoritmos de diferentes paquetes están disponibles para Weka, algunos de ellos que ya se encontraban en versiones anteriores y otros que se van incorporando, pero como se ha comentado al principio, hay que cargarlos desde el Menú Tools (Figura 30).

Desde este Menú también se pueden realizar otras tareas como son la visualización de otros ficheros en formato nativo, arff, o a través del lenguaje sql importándolos desde otras bases de datos (ArffViewer y SqlViewer). También se pueden editar y manejar desde esta pestaña redes bayesianas (Bayes net editor).

FIGURA 30. MENÚ DE CARGA DE PAQUETES DE WEKA



Fuente: Weka

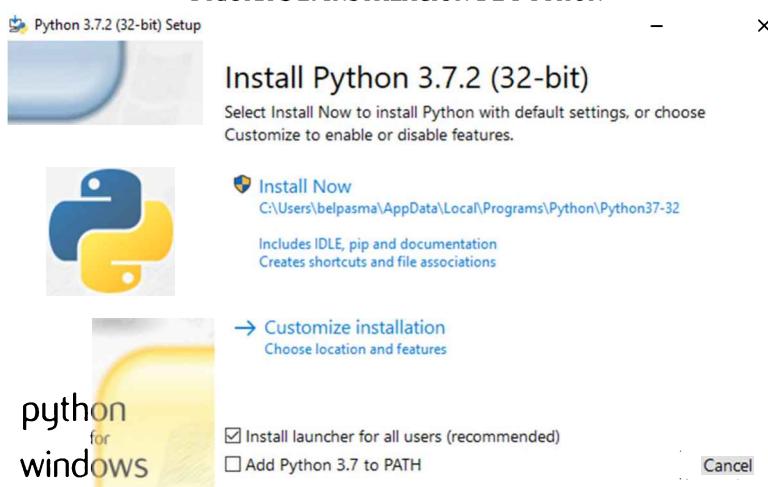
4.3. INTRODUCCIÓN AL LENGUAJE DE PROGRAMACIÓN PYTHON

El lenguaje Python es ampliamente utilizado por los científicos de datos. La razón fundamental es que es fácil de entender y resulta cómodo para escribir código, así que este lenguaje de programación se emplea de forma eficiente en Data Science, Inteligencia Artificial, en desarrollos para la Web y en otros muchos campos del conocimiento. La sintaxis de Python se puede considerar que es simple cuando se compara con otros lenguajes, siendo así preferida por los que empiezan a programar. Se puede afirmar que se aproxima al lenguaje natural y a los conceptos algorítmicos clásicos, lo que permite realizar una gran cantidad de tareas de forma muy fácil. Por otra parte, Python también está lleno de recursos y bibliotecas.

Utilizando expresiones más técnicas podemos definir a Python como un lenguaje de programación orientado a objetos, como un lenguaje interpretado y multiparadigma de alto nivel y que dispone de una gestión automática de recursos, además de tener un alto grado de introspección y de gestión de excepciones. Es libre y gratuito y dispone de un potente conjunto de librerías y puede clonar cualquier otro programa de cálculo numérico como Matlab o Matemática.

Para instalar Python podemos descargar el fichero ejecutable desde la página web oficial: <https://www.python.org/downloads/> y seguir los pasos que se nos indican.

FIGURA 31. INSTALACIÓN DE PYTHON



Fuente: Python

Sin embargo, es más práctico instalar la versión llamada Anaconda, considerada mucho más útil, dado que se instalan automáticamente más de 150 paquetes que contienen alrededor de 550 librerías, (módulos) lo que resulta más ventajoso para aplicaciones en todos los campos de conocimiento. Muchas de estas librerías están directamente relacionadas con Data Science y el Big Data.

Para descargar la versión de Anaconda iremos a la página web oficial que mantiene este programa: <https://www.anaconda.com/download/>

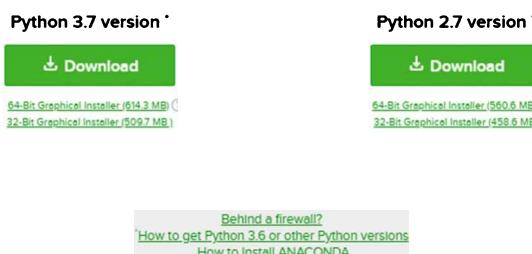
En primer lugar, elegimos el sistema operativo donde queremos utilizar el programa.

FIGURA 32. INSTALACIÓN ANACONDA 1



FIGURA 33. INSTALACIÓN ANACONDA 2

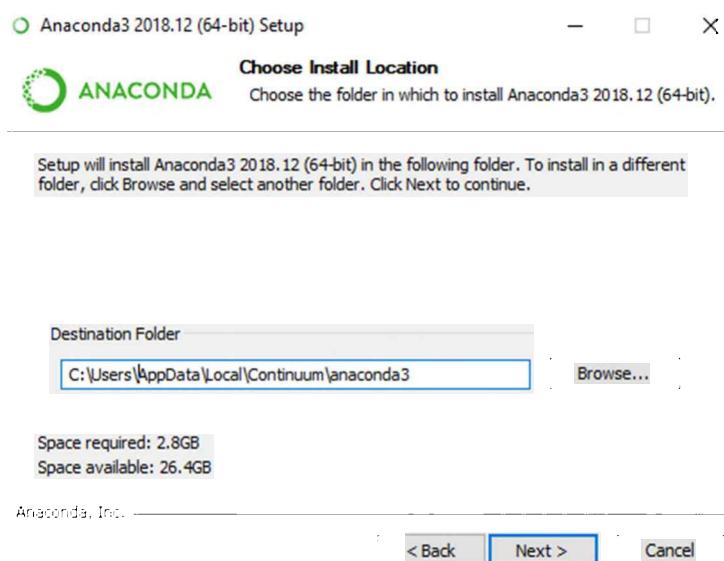
[Anaconda 2018.12 For Windows Installer](#)



Fuente: Python

Una vez bajado el programa procedemos a su instalación siguiendo las instrucciones: aceptando los términos del contrato, el tipo de instalación o cambiando el directorio de ubicación del programa.

FIGURA 34. INSTALACIÓN ANACONDA 3

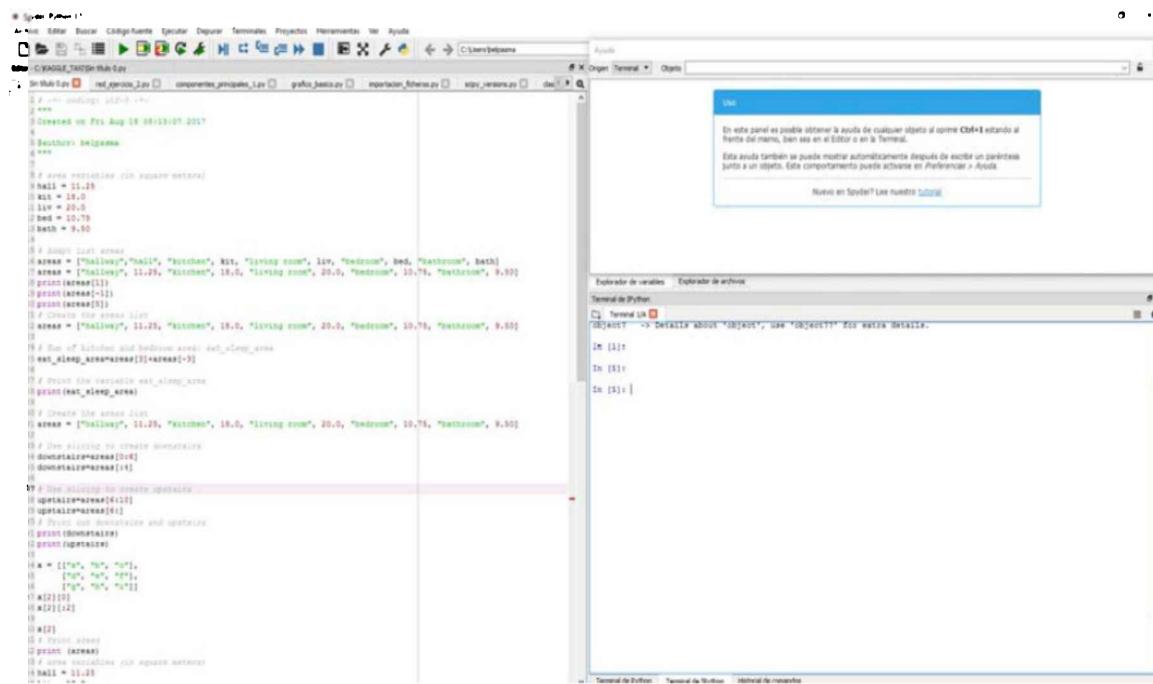


Fuente: Python

Cuando se instala Python podremos utilizar diferentes entornos de desarrollo, algunos de ellos se han implementado durante el proceso: Spyder, IPython o Jupyter Notebook.

En la Figura 35 se muestra las diferentes partes del IDE de Spyder que nos va a permitir la generación, el depurado y la ejecución de código. Reseñar que hay bastantes editores disponibles y que cada programador puede tener sus preferencias en función de su propia experiencia y el nivel de conocimiento disponible.

FIGURA 35. INTERFACE DE PYTHON



Fuente: Python

Aunque la información de los diferentes paquetes que se han instalado está disponible en la siguiente dirección: <https://docs.continuum.io/anaconda/packages/pkg-docs> vamos a sintetizar algunas de sus librerías dado que se centran en la gestión de las bases de datos y en el tratamiento estadístico de los datos. A modo de resumen, entre las más importantes para el desarrollo de un proceso de minería de datos, podemos citar las siguientes:

- Statsmodels
- Scikit-learn
- Scipy
- Matplotlib
- Pandas
- Numpy
- Conda

Statsmodels. Es el módulo más específico de Python que nos permite utilizar a través de sus clases y funciones la estimación de muchos modelos estadísticos diferentes, así como para la realización de pruebas estadísticas y la exploración de bases de datos. Contiene una extensa lista de estadísticas para cada estimador para asegurarse de que son correctos. El paquete tiene licencia Open Source Modified BSD (3 cláusulas). Entre los análisis que se pueden efectuar se incluyen modelos de componentes principales, test estadísticos, ANOVA,

análisis bayesianos, series temporales, etcétera. Una extensa documentación la podemos encontrar en la dirección web: statsmodels.org.

Scikit-learn. Esta librería es la parte central de Python para aplicar modelos de Data Science tanto paramétricos como no paramétricos. Modelos de clasificación y de regresión, de aprendizaje supervisado o no, que incluyen algoritmos y métodos como los vecinos próximos, regresión múltiple, redes neuronales, máquinas de vectores soporte, Stochastic Gradient Descent, algoritmos de ensemble y otros procedimientos para llevar a cabo trabajos de minería de datos y de Big Data.

Scipy. Es un ecosistema basado en Python de código abierto para matemáticas, ciencias e ingeniería. Proporciona entre otras utilidades muchas rutinas numéricas eficientes y fáciles de usar para el tratamiento de imágenes, la integración numérica y la optimización.

Matplotlib. Es una potente librería que puede generar gráficos de calidad, histogramas, espectros de potencia, gráficos de barras, gráficos de errores, diagramas de dispersión, etc., con solo unas pocas líneas de código. Otro módulo, pyplot, proporciona una interfaz tipo MATLAB, especialmente cuando se combina con IPython. Dispone de un control total de los estilos de línea, las propiedades de las fuentes, las propiedades de los ejes, etc., a través de una interfaz orientada a objetos o a través de un conjunto de funciones familiares para los usuarios de MATLAB. Existen otros programas que se integran en Python como mayabi o seaborn que nos permiten ampliar los análisis gráficos.

Pandas. Es la biblioteca de análisis de datos de Python. Este módulo es de código abierto, con licencia BSD que nos permite trabajar con la importación y exportación de todos los formatos de ficheros de datos, así como con estructuras de datos de forma muy eficaz y fácil de usar. También dispone de herramientas de análisis de datos para el lenguaje de programación Python.

Numpy. Es el paquete fundamental del que dispone Python para la computación científica actuando de forma muy eficaz en el tratamiento de los datos científicos: selección, ordenación, ajustes, interpolaciones, estadística, polinomios, funciones matemáticas, arrays con máscara..

Conda. Es una excelente utilidad, un gestor de entorno incluido con Anaconda que se encarga de instalar y actualizar paquetes y utilidades y sus dependencias en Python.

La integración con el programa R es bastante sencilla dado que con un solo comando de la librería conda se puede instalar fácilmente el lenguaje de programación R, y con el más de 80 de los paquetes y sus dependencias que son los más utilizados en la minería de datos. En la siguiente dirección web se puede ampliar esta información: <https://docs.anaconda.com/anaconda/user-guide/tasks/using-r-language/>.

La forma de interactuar de Python con Weka la proporciona el paquete WekaPyScript que es un software de aprendizaje que permite que los algoritmos y los métodos de preprocessamiento para la clasificación y regresión se escriban en Python, en contraposición al lenguaje de implementación de WEKA que es Java. Weka es una de las bibliotecas de aprendizaje automático más conocidas con un gran número de algoritmos implementados. Se puede instalar Jpype (<http://jpype.sourceforge.net/>) para acceder a las bibliotecas de clases de Weka. Una vez que se haya instalado, hay que descargar las versiones más recientes de Weka & Moa y copiar los ficheros: moa.jar, sizeofag.jar y weak.jar en el directorio de trabajo.

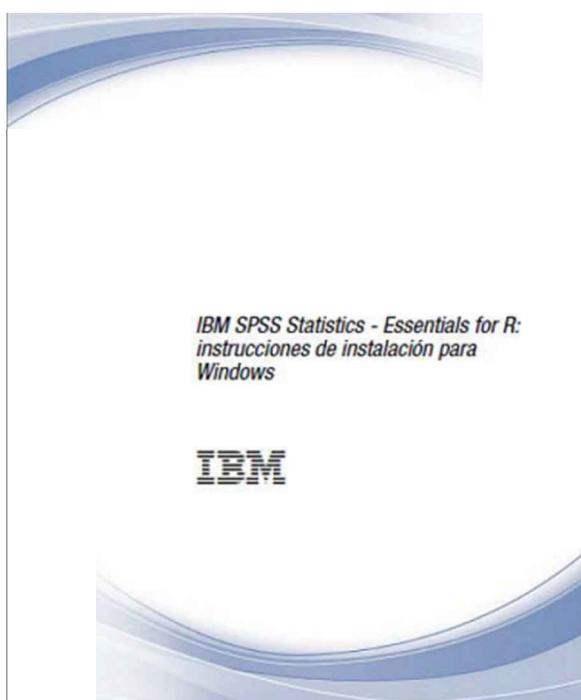
4.4. EL PROGRAMA IBM SPSS STATISTICS: CONEXIÓN CON R Y PYTHON

IBM SPSS Statistics es un software estadístico de pago muy utilizado por muchas organizaciones e investigadores, especialmente procedentes de las ciencias socioeconómicas, para resolver una gran variedad de problemas de investigación. En este apartado solo se pretende dar a conocer las aportaciones que realizan los programas R y Python ya que añaden muchas funcionalidades y dotan al programa SPSS de más recursos analíticos y de más potencia estadística.

El lenguaje Python está disponible desde hace ya unas cuantas versiones cuando se instala el programa SPSS, ya que es una opción que viene incorporada con el mismo programa y que solo hay que activarla en la instalación.

La instalación de R es un poco más compleja ya que hay que seguir unos pasos que se detallan en el documento "IBM SPSS Statistics - Essentials for R: instrucciones de instalación para Windows" y que se descarga desde la web.

FIGURA 36. MANUAL DE INSTALACIÓN DE COMPLEMENTOS DE R



Fuente: IBM SPSS

Siguiendo esta guía de instalación podrás observar como IBM SPSS Statistics - Essentials for R proporciona las herramientas necesarias para comenzar a desarrollar aplicaciones personalizadas de R para su uso con IBM SPSS Statistics. Cada versión de SPSS necesita una instalación concreta de R que se puede descargar desde la página: <http://www.r-project.org/>. Todas las extensiones R incluyen un diálogo personalizado y un comando de extensión. Los comandos de extensión se pueden ejecutar desde la sintaxis de comandos de SPSS Statistics del mismo modo que cualquier comando incorporado, como por ejemplo FREQUENCIES. Puede generar una sintaxis de comandos para cada comando de extensión desde el diálogo personalizado asociado.

TABLA 3. LISTADO DE EXTENSIONES DE R

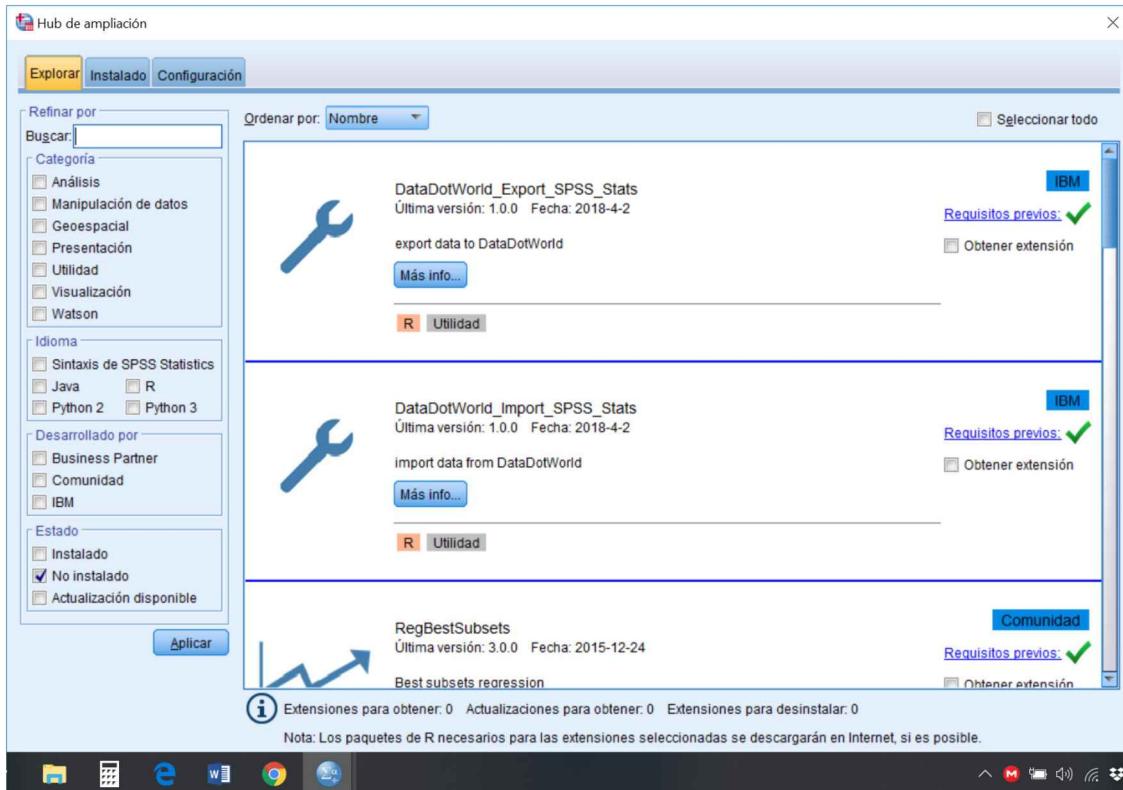
Ubicación de menú	Nombre de comando	Descripción
Analizar > Informes>Apriori	SPSSINC APRIORI	Descubrir conjuntos de elementos frecuentes, reglas de asociación mediante el algoritmo Apriori.
Analizar > Correlaciones > Correlaciones heterogéneas	SPSSINC HETCOR	Calcular correlaciones entre variables nominales, ordinales y de escala.
Analizar > Estadísticos descriptivos > Gráfico Q-Q de dos variables o de grupos	SPSSINC QQPLOT2	Gráfico Q-Q de dos variables o dos grupos.
Analizar > Regresión > Regresión cuantil	SPSSINC QUANTREG	Estimar uno o más cuantiles condicionales para un modelo lineal.
Analizar > Estimación de RanFor	SPSSINC RANFOR	Estimar bosque aleatorio.
Analizar > Estimación de Ranfor	SPSSINC RANPRED	Calcular valores pronosticados para datos nuevos mediante bosques de SPSSINC RANFOR.
Analizar > Regresión > Regresión robusta	SPSSINC ROBUST REGR	Estimar un modelo de regresión lineal mediante una regresión robusta, utilizando un estimador M.
Analizar > Regresión > Regresión tobit	SPSSINC TOBIT REGR	Estimar un modelo de regresión cuya variable dependiente tenga un límite inferior o superior fijo, o ambos.
Analizar > Supervivencia > Extensión de regresión de Cox	STATS COXREGR	Regresión de Cox (riesgos proporcionales).
Analizar > Clasificar > Pronosticar utilizando clúster de densidad	STATS DBPRED	Predicción basada en clústeres basados en densidad.
Analizar > Clasificar > Clústeres basados en densidad	STATS DBSCAN	Clústeres basados en densidad.
Analizar > Regresión > Sistemas de ecuación	STATS EQNSYSTEM	Estimar sistema de ecuaciones lineales.
Analizar > Escala > Rasch ampliado	STATS EXRASCH	Calcular modelos Rasch estándar y ampliados.
Analizar > Regresión > Regresión logística de Firth	STATS FIRTHLOG	Regresión logística de Firth.
Analizar > Predicciones>Modelos	GARCH STATS GARCH	Modelos GARCH.
Analizar > Modelos lineales generalizados > Regresión ampliada generalizada	STATS GBM	Estimar modelos de regresión ampliada generalizada.
Analizar > Modelos lineales generalizados > Predicción de regresión ampliada generalizada	STATS GBMPRED	Calcular predicciones para modelos de regresión ampliada generalizada.
Archivo > Obtener espacio de trabajo de R	STATS GET R	Obtener información sobre el contenido del espacio de trabajo de R y crear conjuntos de datos SPSS.
Analizar > Escala > Modelo de respuesta graduado	STATS GRM	Ajustar modelos de respuesta graduados a los datos ordinales.
Analizar > Escala > Modelo de respuesta de elemento	STATS IRM	Ajustar modelos de respuestas de elementos de tres parámetros.
Analizar > Loglineal > Análisis de clases latentes	STATS LATENT CLASS	Análisis de clases latentes.
Analizar > Estadísticos descriptivos > Calcular los valores P ajustados	STATS PADJUST	Calcular valores p ajustados para múltiples pruebas.
Analizar > Modelos lineales generalizados > Regresión proporcional	STATS PROPOR REGR	Modelos lineales para variables dependientes que son proporciones.
Analizar > Modelos lineales generalizados > Predicción de regresión proporcional	STATS PROPOR REGRPRED	Calcular valores pronosticados para modelos de regresión proporcional.
Analizar > Regresión > Discontinuidad de regresión	STATS RDD	Análisis de discontinuidad de regresión.
Analizar > Regresión > Importancia relativa de regresión	STATS RELIMP	Medidas de importancia relativa para la regresión.
Analizar > Supervivencia > Regresión paramétrica	STATS SURVREG	Regresión paramétrica de supervivencia.
Analizar > Clasificar > Máquinas de vectores de soporte	STATS SVM	Máquina de vectores de soporte.
Analizar > Modelos lineales generalizados > Modelos de recuento inflados de ceros	STATS ZEROINFL	Estimar y pronosticar un modelo de recuento inflado de ceros.

Fuente: IBM SPSS

Una vez instaladas las utilidades de R y Python desde el menú de SPSS /Ampliaciones/Hub de ampliación podemos instalar todos los métodos de visualización y análisis estadísticos

que nos ofrece y, que una vez que se hayan descargado, se quedan integrados en los diferentes menús de SPSS.

FIGURA 37. HUB DE AMPLIACIÓN DE SPSS



Fuente: IBM SPSS

Ahora ya se puede integrar la programación de R a través de la sintaxis de SPSS, simplemente empezando el script con el comando *BEGIN PROGRAM R*. y acabando con el comando *END PROGRAM*. Entre ambos comandos incluimos los comandos de R.

FIGURA 38. SCRIPT DE SPSS CON CÓDIGO DE R

The screenshot shows the 'Sintaxis_modelo_tobit.sps - IBM SPSS Statistics Editor de sintaxis' window. The menu bar includes Archivo, Editar, Ver, Datos, Transformar, Analizar, Marketing direct, Gráficos, Utilidades, Ejecutar, Herramientas, Ampliaciones, Ventana, Ayuda. The toolbar has various icons for file operations. The syntax editor window contains the following R script:

```

* Encoding: UTF-8.
BEGIN PROGRAM R.
#help.start()
#dat <- read.csv("http://www.ats.ucla.edu/stat/data/tobit.csv")
#head(dat)
library(survival)
tobin

str(tobin)

tobinfit<-survreg(Surv(durable, durable>0, type="left") ~age+quant,
                     data=tobin, dist="gaussian")
tobinfit

summary(tobinfit)
attributes(tobinfit)

fit<-lm(durable~ age + quant, data = tobin)
fit

END PROGRAM.

```

The status bar at the bottom indicates: 'IBM SPSS Statistics Processor está listo | Unicode:ON | In 1 Col 0 | NUM |'

Fuente: IBM SPSS

4.5. OTROS SOFTWARES: JULIA Y SCALA

4.5.1. Julia

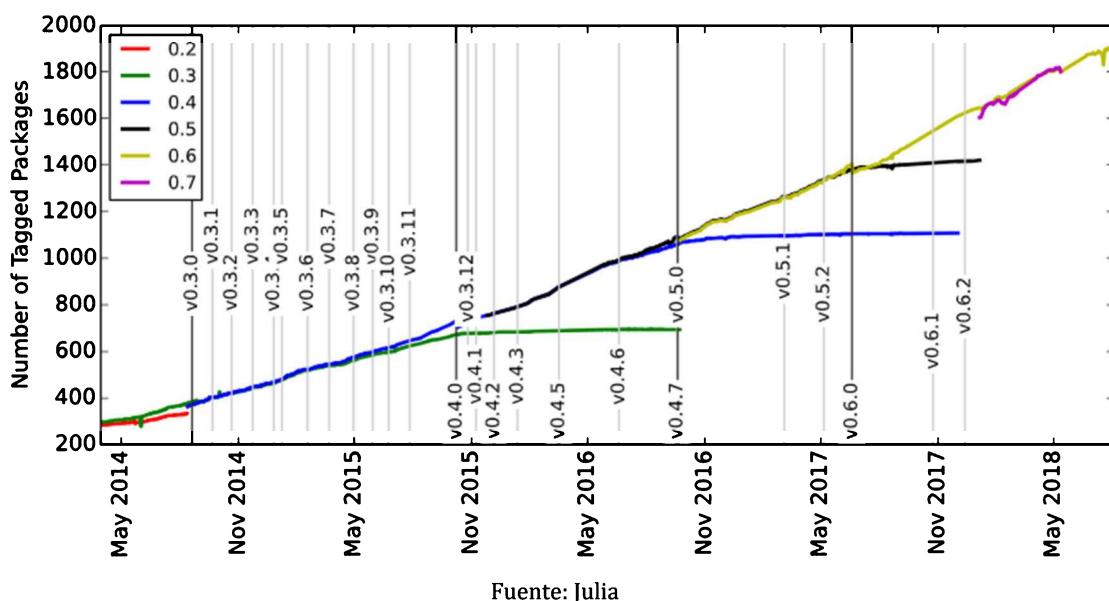
Este lenguaje de programación fue creado por Stefan Karpinsky junto con sus compañeros del MIT, Viral Shah y Jeff Bezanson ante la imposibilidad de que un solo programa pudiera resolver el trabajo de creación de una herramienta de simulación de redes en el que estaban trabajando. Decidieron crear su propio lenguaje y lo bautizaron como Julia y lo lanzaron el año 2009.

Es un lenguaje de programación dinámico de alto nivel y desempeño para la computación técnica (en este sentido se acerca a otros lenguajes compilados estáticamente como el C) con una sintaxis muy reconocible por otros programadores. Está dotado con un potente compilador muy avanzado JIT (compilador Just In Time) y dispone de mecanismos para la ejecución de procesos en paralelo y distribuidos, lo que le faculta para trabajar en la nube y con grandes volúmenes de datos (Big Data). No se impone ningún estilo de paralelismo sino que se provee al usuario con una serie de bloques de construcción clave para la computación distribuida.

Tiene una extensa biblioteca de funciones matemáticas y la biblioteca básica integra las mejores librerías de C y Fortran para el álgebra lineal, el proceso de cadenas, el procesamiento de señales, etcétera. Tiene también la facultad de llamar funciones de Python a través del programa PyCall y su interrelación con el lenguaje C es directa.

Dispone de una licencia MIT: Libre y de código abierto. La comunidad de desarrolladores ha contribuido en muy poco tiempo con una gran cantidad de paquetes. En la página oficial del programa: <https://julialang.org/> se puede ver un listado de los 1.540 paquetes registrados actualmente. A continuación, se muestra en la Figura 39 la tendencia en los últimos tres años en la integración de paquetes que se está produciendo de este joven programa que cada día es más empleado en las plataformas de Big Data.

FIGURA 39. EVOLUCIÓN DEL NÚMERO DE PAQUETES INCLUIDOS EN JULIA



4.5.2. Scala

El nombre surge de la combinación de dos palabras: scalable y language y la escalabilidad es una de las características más atractivas de este programa. Este programa apareció en 2003 y fue diseñado por Martin Odersky.

Pertenece a una de las últimas generaciones de lenguajes funcionales, en este sentido, Scala es uno de los más populares ya que es un lenguaje de programación multi-paradigma que soporta el paradigma funcional (la programación funcional está formada por funciones matemáticas). También es un programa orientado a objetos como Python, y como lenguaje multiparadigma permite al programador elegir diferentes "estilos" de desarrollo dependiendo del tipo de problema: programación lógica, funcional, orientada a objetos, etc.

Este programa se ejecuta sobre una máquina virtual Java, lo cual permite la integración de características o librerías entre ambos lenguajes. Esta integración es recíproca, es decir, podemos utilizar Java en Scala y éste en Java así que es posible, entre otras cosas, llamar a métodos de Java, heredar clases o implementar interfaces. Aunque también existen algunas diferencias entre ambos lenguajes: en Scala se reducen drásticamente el número de líneas de código que se escribirán en un proyecto y, como ya se ha apuntado, Scala favorece la concurrencia y paralelización en la ejecución del código. El código Java es más legible que el de Scala y trata todo como un objeto. Scala, sin embargo, aborda todo como una variable.

Desde la siguiente dirección de internet: <https://index.scala-lang.org/> se puede acceder a las diferentes librerías que dispone este lenguaje. Están librerías están agrupadas por grandes temas. En el apartado de machine learning se encuentran doce grandes proyectos de Scala. Se comentan someramente algunos de estos proyectos y sus características:

- **Deep Learning.scala.** Es una biblioteca sencilla para crear redes neuronales complejas a partir de construcciones con paradigmas de programación orientada a objetos y funcionales.
- **SparklingGraph.** Proporciona un conjunto de funciones fáciles de usar que permiten procesar gráficos utilizando Spark y GraphX.
- **Deeplearning4j/datavec.** Es una biblioteca con licencia de Apache 2.0 para operaciones ETL (Extract, Transform, Load) de machine learning. El propósito de DataVec es transformar datos sin procesar en formatos vectoriales utilizables que puedan ser tratados después con algoritmos de aprendizaje automático.
- **Sparkling Water.** Integra el motor de aprendizaje rápido y escalable de H2O con Spark. H2O es una plataforma para el aprendizaje distribuido y escalable. H2O utiliza interfaces conocidas como R, Python, Scala, Java, JSON y la interfaz de portátiles / web Flow, y funciona perfectamente con grandes tecnologías de datos como Hadoop y Spark. H2O proporciona implementaciones de muchos algoritmos populares como GBM, Random Forest, Redes Neuronales Profundas, Word2Vec y Conjuntos Apilados. H2O es extensible por lo que los desarrolladores puedan agregar sus propias transformaciones de datos y algoritmos personalizados y acceder a ellos.
- **NeuroFlow.** Es una biblioteca para entrenar y evaluar Redes Neuronales Artificiales. Los conjuntos de entrenamiento voluminosos pueden ser distribuidos sobre nodos físicos y entrenados en paralelo, usando Akka.
- **Spotify/featran.** Es una biblioteca de transformación de características para data science y machine learning.

TEMA 2: INTRODUCCIÓN AL LENGUAJE R

ÍNDICE

1. Introducción a R y Ayuda en Línea

2. Objetos en R

- 2.1. Vectores
- 2.2. Matrices
- 2.3. Listas
- 2.4. Factores
- 2.5. Dataframes

3. Realizar Consultas

- 3.1. Condiciones Simples
- 3.2. Condiciones Múltiples
- 3.3. Filtrar dataframes

4. Funciones

- 4.1. Definir una función
- 4.2. Funciones incorporadas
- 4.3. Funciones definidas por el usuario

5. Estructuras de Control

- 5.1. Sentencias condicionales
- 5.2. Bucles

6. La Familia de Funciones Apply

- 6.1. Las funciones apply vs. bucles

7. Gráficos

OBJETIVOS	PALABRAS CLAVE
<ul style="list-style-type: none"> · Introducción y operativa básica con R · Trabajar con los distintos tipos de objetos en R. · Realizar consultas. · Utilizar las funciones incorporadas en R y definir funciones propias. · Programación de sentencias condicionales y bucles. · Realizar una estadística descriptiva de nuestros datos a partir de las funciones apply y de la generación de gráficos. 	<ul style="list-style-type: none"> · R · Librerías · Funciones · Vectores, matrices, listas, factores, dataframes · Estructuras de control

1. INTRODUCCIÓN A R Y AYUDA EN LÍNEA

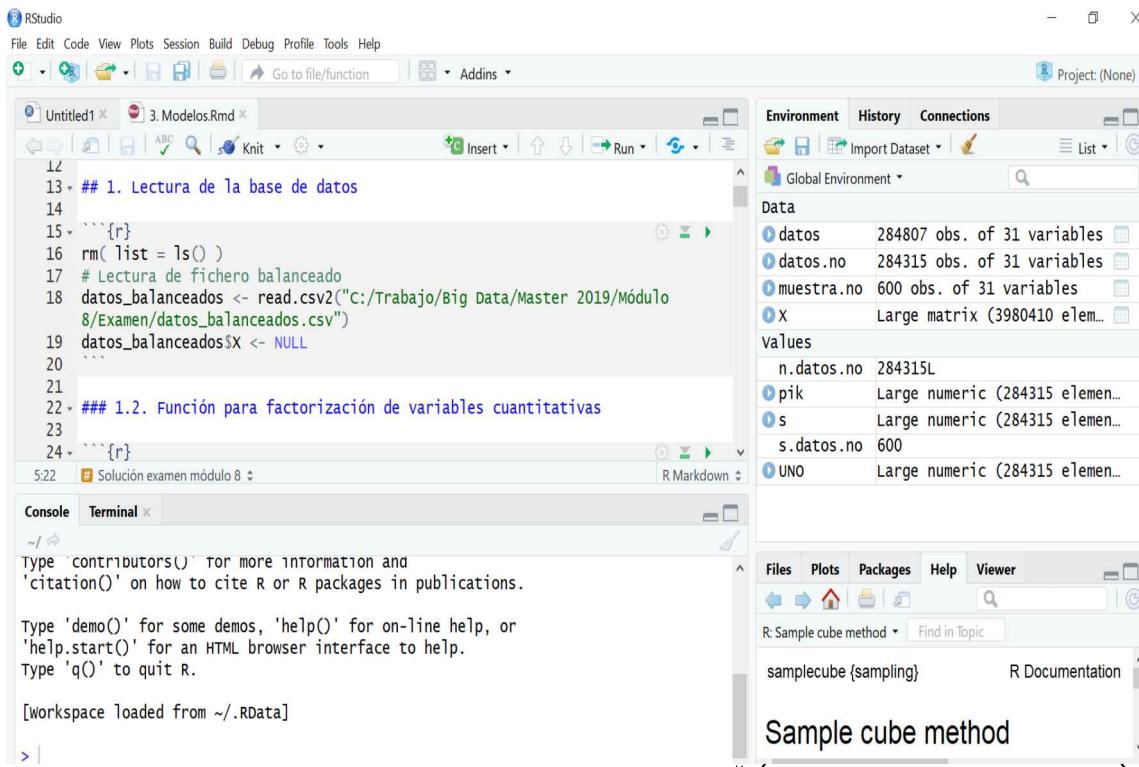
R es un lenguaje de programación y un entorno de software orientado al análisis estadístico, además del cálculo numérico, cabe destacar su potencial para la representación gráfica y la creación de informes.

R es gratuito y se distribuye bajo la Licencia Pública General de GNU. Está disponible para los sistemas operativos: Windows, Mac y Linux.

La interfaz propia de R es poco amigable, una simple consola para escribir y ejecutar código. La mejor plataforma para utilizar R es RStudio.

RStudio es un IDE muy popular, que ofrece un entorno amigable para trabajar en R. Un IDE (Integrated Development Environment) es un entorno de desarrollo integrado es una aplicación informática que proporciona servicios integrales para facilitarle al programador el desarrollo de software.

FIGURA 1. CONSOLA RSTUDIO



Fuente: RStudio IDE

A grandes rasgos, RStudio se compone de cuatro secciones:

1. Izquierda-Arriba (esta sección se abrirá cuando sea necesario):
 - Editor de código: Para escribir y guardar scripts de R. También se puede editar/guardar cualquier archivo de texto (.csv, .txt), HTML, etc.
 - Visor de objetos: Para consultar el contenido de ciertos objetos R.
2. Izquierda-Abajo: La Consola, donde se escriben y ejecutan los comandos de R.
3. Derecha-Arriba:

- Environment: Muestra el entorno de trabajo, en el que iremos viendo los objetos R (variables y funciones) que vayamos creando, cargando,... Obsérvese que esta pestaña contiene ciertos iconos que permiten guardar el contenido de la memoria, cargar el contenido de la memoria de una sesión de trabajo anterior, importar archivos de datos (CSV, Excel, SPSS,...) y limpiar el contenido de la memoria.
- History: Guarda un historial de comandos R según se van introduciendo en la consola.

4. Derecha-Abajo:

- Files: Explorador de archivos. Por defecto, el directorio actual es home.
- Plots: Se mostrarán los gráficos que generemos en R.
- Packages: Podemos ver qué paquetes tenemos instalados (un paquete es una colección de funciones que aumenta la funcionalidad de R). También nos permite descargar e instalar nuevos paquetes, y borrar paquetes instalados.
- Help: Permite acceder a la ayuda de R.
- Viewer: Muestra contenido web local.

Por otra parte, la ayuda en línea de R proporciona información muy útil de cómo utilizar las funciones. La ayuda se encuentra disponible directamente para una función dada. Por ejemplo:

```
?lm  
starting httpd help server ... done
```

El comando `help(lm)` o `help("lm")` tiene el mismo efecto. Esta última función se debe usar para acceder a la ayuda con caracteres no-convencionales:

```
help("!")
```

El mismo resultado se obtiene en el cajetín de la pestaña "help" de la ventana que se despliega en el cuadrante derecho-abajo de la consola R-Studio.

2. OBJETOS EN R

En cualquier lenguaje de programación es necesario usar *variables* para almacenar información. Las variables no son más que ubicaciones de memoria reservadas para almacenar valores. Los datos a almacenar pueden ser de varios *tipos*, como: carácter, numérico (entero o coma flotante), lógico, etc.

A diferencia de otros lenguajes de programación (como *C* y *Java*), en *R* las variables no se declaran como un tipo de datos. Las variables se asignan con *objetos R*, y el *tipo de datos* del objeto R, se convierte en el tipo de datos de la variable. Hay muchas clases de *objetos en R*, las más comunes para almacenar datos son:

- Vectores.
- Matrices.
- Listas.
- Factores.
- Dataframes.

Durante una sesión de *R*, todos los objetos estarán en memoria y se pueden guardar en disco para futuras sesiones.

2.1. VECTORES

Los vectores son los *objetos de datos* en R más básicos (estructura de datos *unidimensional*).

2.1.1. Crear

Para *asignar* a una variable un valor determinado, se suele utilizar el operador `<-`. También se puede utilizar el operador `=`. La función `class` permite conocer la *clase* del objeto. La función `typeof` permite conocer el *tipo* del objeto, cómo se almacena el objeto en memoria. El carácter `#` se utiliza para introducir un *comentario*.

A continuación, se explica cómo *crear* vectores.

Vectores de un solo elemento

```
variable <- 3
class(variable)
[1] "numeric"
typeof(variable)
[1] "double"

variable <- 3L # Sufijo "L": el número debe ser almacenado como un entero
class(variable)
[1] "integer"
typeof(variable)
[1] "integer"

variable <- 3.5
class(variable)
[1] "numeric"
typeof(variable)
[1] "double"

variable <- "manzana"
class(variable)
[1] "character"
typeof(variable)
[1] "character"

variable <- "3"
class(variable)
[1] "character"
typeof(variable)
[1] "character"

variable <- TRUE
class(variable)
[1] "logical"
typeof(variable)
[1] "logical"

variable <- FALSE
class(variable)
[1] "logical"
typeof(variable)
[1] "logical"
```

Vectores de varios elementos

En los ejemplos anteriores, hemos creado vectores de un solo elemento. Se pueden crear vectores de *varios* elementos utilizando:

- El operador : genera una secuencia de números (con un incremento de 1 o -1) para crear un vector.
- La función c reúne varios elementos para formar un vector.
- La función seq genera una secuencia de números para crear un vector, se puede especificar el incremento o el número de elementos.
- La función rep replica los elementos de un vector.

Otras funciones de interés:

- La función length muestra el número de elementos de un objeto (longitud).
- La función str muestra la estructura de un objeto.
- La función summary muestra un resumen estadístico de un objeto.

```
x <- 1:5 # Genera una secuencia de números del 1 al 5
x
[1] 1 2 3 4 5

class(x)
[1] "integer"

typeof(x)
[1] "integer"

length(x)
[1] 5

str(x)
int [1:5] 1 2 3 4 5

summary(x)
   Min. 1st Qu. Median Mean 3rd Qu. Max.
1       2      3     3     4      5

x <- 9:5 # Genera una secuencia de números del 9 al 5
x
[1] 9 8 7 6 5

x <- c(1, 4, 3, 5)
x
[1] 1 4 3 5

class(x)
[1] "numeric"

typeof(x)
[1] "double"

x <- seq(1, 9, 2) # Genera una secuencia de números del 1 al 9, con un incremento de
2
x
[1] 1 3 5 7 9

seq(1, 9, length = 5) # Genera una secuencia de 5 números, del 1 al 9
[1] 1 3 5 7 9

seq(1, 9, length = 6) # Genera una secuencia de 6 números, del 1 al 9
[1] 1.0 2.6 4.2 5.8 7.4 9.0

rep(2, 4) # Repite "2" cuatro veces
[1] 2 2 2 2

rep(1:4, 3) # Repite "1,2,3,4" tres veces
```

```
[1] 1 2 3 4 1 2 3 4 1 2 3 4  
x <- c("manzana", "pera", "naranja")  
x <- c(TRUE, FALSE, TRUE)
```

Si un vector es de una clase o tipo concreto (carácter, entero, numérico de doble precisión, lógico, ...), ¿qué pasaría si mezclásemos elementos de diferentes tipos?

```
x <- c(3, TRUE, "naranja")  
x  
[1] "3" "TRUE" "naranja"  
  
class(x)  
[1] "character"  
  
x <- c(4.25, FALSE, 12L, TRUE)  
x  
4.25 0.00 12.00 1.00  
  
class(x)  
[1] "numeric"
```

2.1.2. Seleccionar elementos

Los elementos de un vector se seleccionan mediante el *índice*, es decir, la posición que ocupa un elemento en el vector. Para la indexación se utilizan los *corchetes* []. La indexación comienza en la posición 1. Si se asigna un valor *negativo* al índice, se eliminará ese elemento del resultado. TRUE y FALSE, también se pueden usar en la indexación.

```
x <- c("lunes", "martes", "miércoles", "jueves", "viernes", "sábado", "domingo")  
z <- x[c(2,3,6)] # Elementos 2º, 3º y 6º del vector  
z  
[1] "martes" "miércoles" "sábado"  
  
z <- x[-7]  
z  
[1] "lunes" "martes" "miércoles" "jueves" "viernes" "sábado"  
  
z <- x[c(-2,-5)]  
z  
[1] "lunes" "miércoles" "jueves" "sábado" "domingo"  
  
z <- x[c(TRUE, FALSE, FALSE, FALSE, FALSE, TRUE, FALSE)]  
z  
[1] "lunes" "sábado"  
  
z <- x[c(F,T,F,F,T,F,F)]  
z  
[1] "martes" "viernes"  
  
# Actualizar un elemento del vector  
x[2] <- "MARTES"  
x  
[1] "lunes" "MARTES" "miércoles" "jueves" "viernes" "sábado" "domingo"  
  
v <- seq(1,2,.1)  
v  
[1] 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0  
  
# Añadir un elemento al final del vector  
v <- c(v,2.1)  
v
```

```
[1] 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0 2.1

# Añadir un elemento entre los elementos 4º y 5º
v <- c(v[1:4], 1.35, v[5:length(v)])
v
[1] 1.00 1.10 1.20 1.30 1.35 1.40 1.50 1.60 1.70 1.80 1.90 2.00 2.10
```

También podemos asignar *nombres* a los elementos de un vector, mediante la función *names*.

```
precios <- c(0.23, 0.35, 0.14, 0.2, 0.23)
names(precios) <- c("Manzana", "Uva", "Pera", "Naranja", "Cereza")
precios
Manzana    Uva    Pera    Naranja    Cereza
0.23     0.35    0.14     0.20     0.23
```

```
# Seleccionar un elemento utilizando el atributo "nombre"
precios["Naranja"]
Naranja
0.2
```

```
# Seleccionar varios elementos mediante el atributo "nombre"
precios[c("Manzana", "Uva")]
Manzana    Uva
0.23     0.35
```

```
# Eliminar los nombres asignados
names(precios) <- NULL
```

2.1.3. Trabajar con vectores

Operaciones aritméticas

```
v1 <- c(5:1, 6:8, 12:9)
v1
[1] 5 4 3 2 1 6 7 8 12 11 10 9

# Operar con un vector
v1 + 6
[1] 11 10 9 8 7 12 13 14 18 17 16 15

(v1 - 5)^2
[1] 0 1 4 9 16 1 4 9 49 36 25 16
```

Dos vectores de la misma longitud se pueden sumar, restar, multiplicar o dividir, obteniendo un vector resultante.

```
# Creamos dos vectores
v1 <- c(4,9,5,6,0,12)
v2 <- c(5,12,0,9,2,3)

# Sumar vectores
resultado <- v1 + v2
resultado
[1] 9 21 5 15 2 15

# Restar vectores
resultado <- v1 - v2
resultado
[1] -1 -3 5 -3 -2 9

# Multiplicar vectores
resultado <- v1 * v2
```

```
resultado
[1] 20 108 0 54 0 36

# Dividir vectores
resultado <- v1 / v2
resultado
[1] 0.8000000 0.7500000 Inf 0.6666667 0.0000000 4.0000000
```

NA (Not Available/Missing Value): No disponible. Constante lógica que indica un valor perdido.

Inf: Infinito positivo y negativo. Por ejemplo: 1/0, log(0).

NaN (Not a Number): No es un número. Expresa un resultado imposible de calcular, como es el caso de las indeterminaciones, la raíz cuadrada de un número negativo, el logaritmo de un número negativo, etc. Por ejemplo: 0/0, sqrt(-1), log(-1). Para poder controlar estas situaciones, *R* dispone de las siguientes funciones: is.na, is.finite, is.infinite, is.nan.

Si realizamos operaciones aritméticas con dos vectores de diferente longitud, los elementos del vector más corto se repetirían hasta tener la misma longitud que el otro. Veamos un ejemplo.

```
v1 <- c(4,9,5,6,0,12)
v2 <- c(5,12)
# v2 se convierte en c(5,12,5,12,5,12)

v1 + v2
[1] 9 21 10 18 5 24
```

Ordenar vectores

Los elementos de un vector se pueden ordenar mediante la función sort.

```
v <- c(8,4,-5,12,0,7)
v <- sort(v)
v
[1] -5 0 4 7 8 12
sort(v, decreasing = TRUE) # orden decreciente
[1] 12 8 7 4 0 -5

v <- c("rojo","amarillo","verde","azul")
sort(v)
[1] "amarillo" "azul" "rojo" "verde"
sort(v, decreasing = TRUE) # orden decreciente
[1] "verde" "rojo" "azul" "amarillo"
```

2.2. MATRICES

En las matrices los elementos están dispuestos en una estructura de *dos dimensiones* (filas y columnas). Al igual que los vectores, todos los elementos de una matriz serán del *mismo tipo*. Aunque podemos crear una matriz que contenga solo caracteres o solo valores lógicos, no es de mucha utilidad. Generalmente, se utilizan matrices que contengan *números*, para realizar cálculos matemáticos.

2.2.1. Crear

Una matriz se puede crear mediante la función matrix. Veamos la sintaxis de dicha función.

```
matrix(data, nrow, ncol, byrow, dimnames)
```

Descripción de los *argumentos* de la función:

- *data*: vector de entrada, desde el cual se obtendrán los elementos de la matriz.
- *nrow*: número de filas que se crearán.
- *ncol*: número de columnas que se crearán.
- *byrow*: si es TRUE, los elementos del vector de entrada se llenan por fila.
- *dimnames*: nombres asignados a las filas y columnas.

```
# Los elementos se colocan secuencialmente por fila
m <- matrix(1:12, nrow = 4, byrow = TRUE)
m
[,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
[4,]   10   11   12

# Los elementos se colocan secuencialmente por columna
n <- matrix(1:12, nrow = 4) # Por defecto, byrow = FALSE
n
[,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12

# Definir nombres de columnas y filas
nombresfilas <- c("row1", "row2", "row3", "row4")
nombresscolumnas <- c("col1", "col2", "col3")
m <- matrix(1:12, nrow = 4, byrow = TRUE, dimnames = list(nombresfilas, nombresscolumnas))
m
      col1 col2 col3
row1    1    2    3
row2    4    5    6
row3    7    8    9
row4   10   11   12
```

Función *dim*: dimensión del objeto (filas/columnas).

Función *nrow*: número de filas del objeto.

Función *ncol*: número de columnas del objeto.

Función *rownames*: nombres de las filas del objeto.

Función *colnames*: nombres de las columnas del objeto.

```
class(m)
[1] "matrix"

typeof(m)
[1] "integer"

length(m)
[1] 12

dim(m)
[1] 4 3
```

```
nrow(m)
[1] 4

ncol(m)
[1] 3

rownames(m)
[1] "row1" "row2" "row3" "row4"

colnames(m)
[1] "col1" "col2" "col3"

colnames(m) <- c("a", "b", "c")
colnames(m)
[1] "a" "b" "c"

También podemos crear matrices a partir de vectores, mediante las funciones rbind y cbind.

x <- 1:10
y <- 11:20
z <- 21:30

m <- cbind(x, y, z) # Por columnas
m
      x  y  z
[1,] 1 11 21
[2,] 2 12 22
[3,] 3 13 23
[4,] 4 14 24
[5,] 5 15 25
[6,] 6 16 26
[7,] 7 17 27
[8,] 8 18 28
[9,] 9 19 29
[10,] 10 20 30

n <- rbind(x, y, z) # Por filas
n
 [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
x     1     2     3     4     5     6     7     8     9     10
y    11    12    13    14    15    16    17    18    19    20
z    21    22    23    24    25    26    27    28    29    30

n <- cbind(m, 31:40) # Añadir una columna a una matriz
n
      x  y  z
[1,] 1 11 21 31
[2,] 2 12 22 32
[3,] 3 13 23 33
[4,] 4 14 24 34
[5,] 5 15 25 35
[6,] 6 16 26 36
[7,] 7 17 27 37
[8,] 8 18 28 38
[9,] 9 19 29 39
[10,] 10 20 30 40

colnames(n) <- NULL # Eliminar los nombres de las columnas
n
```

```
[,1] [,2] [,3] [,4]
[1,] 1 11 21 31
[2,] 2 12 22 32
[3,] 3 13 23 33
[4,] 4 14 24 34
[5,] 5 15 25 35
[6,] 6 16 26 36
[7,] 7 17 27 37
[8,] 8 18 28 38
[9,] 9 19 29 39
[10,] 10 20 30 40
```

2.2.2. Seleccionar elementos

Los elementos de una matriz se seleccionan mediante el *índice de la fila* y el *índice de la columna*. También se pueden seleccionar elementos utilizando el *nombre* de la fila o de la columna.

Para la indexación se utilizan los *corthetes*, con *dos partes* separadas por una “coma”: la de la izquierda se refiere a las filas y la de la derecha a las columnas [filas, columnas].

```
# El elemento 1a fila y 3a columna
m[1,3]
z
21

# Todos los elementos de la 1a fila
v <- m[1,]
v
x y z
1 11 21
is.vector(v)
[1] TRUE

# Todos los elementos de la 3a columna
v <- m[,3]
v
[1] 21 22 23 24 25 26 27 28 29 30
is.vector(v)
[1] TRUE

# Tres primeras filas. La 1a y 3a columna
n <- m[1:3,c(1,3)]
n
      x  z
[1,] 1 21
[2,] 2 22
[3,] 3 23
is.matrix(n)
[1] TRUE

# Indicar las columnas por nombre
m[1:3,c("x","z")]
      x  z
[1,] 1 21
[2,] 2 22
[3,] 3 23

# Actualizar un elemento de la matriz
m[2,3] <- 50
m[2,]
```

```

x  y  z
2 12 50

# Actualizar toda la 1ª columna de la matriz
m[,1] <- 31:40

```

2.2.3. Trabajar con matrices

Operaciones aritméticas

Diversas operaciones matemáticas se pueden realizar con las matrices. El resultado de la operación también será una matriz.

Para poder operar, las matrices deben tener la misma dimensión, es decir, el mismo número de filas y de columnas.

```

m <- matrix(c(3, 9, -1, 4, 2, 6), nrow = 2)
m
      [,1] [,2] [,3]
[1,]    3   -1    2
[2,]    9    4    6

n <- matrix(c(5, 2, 0, 9, 3, 4), nrow = 2)
n
      [,1] [,2] [,3]
[1,]    5    0    3
[2,]    2    9    4

# Sumar matrices
x <- m + n
x
      [,1] [,2] [,3]
[1,]    8   -1    5
[2,]   11   13   10

# Restar matrices
x <- m - n
x
      [,1] [,2] [,3]
[1,]   -2   -1   -1
[2,]    7   -5    2

# Multiplicar matrices
x <- m * n
x
      [,1] [,2] [,3]
[1,]   15    0    6
[2,]   18   36   24

# Dividir matrices
x <- m / n
x
      [,1]      [,2]      [,3]
[1,]  0.6      -Inf  0.6666667
[2,]  4.5  0.4444444  1.5000000

```

2.3. LISTAS

Las listas son *objetos R* que contienen elementos de *diferentes tipos*, como: números, cadenas de texto, vectores, matrices, etc. Las listas son estructuras de datos que encapsulan toda esa

información heterogénea. Una lista también puede contener otras listas que, a su vez, pudieran contener otras.

2.3.1. Crear

Una lista se crea utilizando la función `list`.

A continuación, creamos una lista que contiene: cadenas de texto, números, vectores y valores lógicos.

```
lista <- list("rojo", "azul", c(15,28,9), TRUE, 23.14, 108.1)
lista
[[1]]
[1] "rojo"

[[2]]
[1] "azul"

[[3]]
[1] 15 28 9

[[4]]
[1] TRUE

[[5]]
[1] 23.14

[[6]]
[1] 108.1

class(lista)
[1] "list"

length(lista)
[1] 6
```

2.3.2. Seleccionar elementos

Se puede acceder a los elementos de una lista, mediante el *índice* del elemento en la lista. Las listas disponen de un operador para extraer elementos: los dobles corchetes `[[]]`.

```
# Acceder al primer elemento de la lista
lista[1]
[[1]]
[1] "rojo"

lista[[1]]
[1] "rojo"

length(lista[[3]])
[1] 3

lista[[3]][2]
[1] 28
```

Es posible asignar nombres a los elementos de una lista y se puede acceder a ellos usando el *nombre*, mediante el operador `$`.

```
# Creamos una lista que contiene: un vector, una matriz y una lista
lista <- list(c("Enero","Febrero","Marzo"), matrix(c(3,9,5,1,-2,8), nrow=2),
             list("verde",12.3))
```

```
# Asignar nombres a los elementos de la lista
names(lista) <- c("trim1", "matriz", "lista2")
lista
$trim1
[1] "Enero"   "Febrero" "Marzo"

$matriz
[,1] [,2] [,3]
[1,]    3    5   -2
[2,]    9    1    8

$lista2
$lista2[[1]]
[1] "verde"

$lista2[[2]]
[1] 12.3

# Acceder a un elemento de la lista usando su nombre
lista$trim1
[1] "Enero"   "Febrero" "Marzo"

lista$trim1[2]
[1] "Febrero"

lista$matriz[2,]
[1] 9 1 8

lista$matriz[2,1]
[1] 9
```

2.3.3. Manipular elementos

Podemos *añadir, eliminar y actualizar* elementos de una lista.

```
lista <- list(a = 1:3, b = c("hola", "adiós"))
# Añadir un elemento (es añadido al final de la lista)
lista$c <- matrix(1:4, 2, 2)
lista
$a
[1] 1 2 3

$b
[1] "hola"   "adiós"

$c
[,1] [,2]
[1,]    1    3
[2,]    2    4

# Borrar un elemento
lista$a <- NULL

# Actualizar un elemento
lista$b <- 6:4
lista
$b
[1] 6 5 4

$c
```

```
[,1] [,2]
[1,]   1   3
[2,]   2   4
```

2.3.4. Unir listas

Veamos cómo unir varias listas en una sola lista.

```
lista1 <- list(1:3)
lista2 <- list(1,2,3)
lista3 <- list("Enero","Febrero","Marzo")

# Unir varias listas
lista4 <- c(lista1,lista2,lista3)
lista4
[[1]]
[1] 1 2 3

[[2]]
[1] 1

[[3]]
[1] 2

[[4]]
[1] 3

[[5]]
[1] "Enero"

[[6]]
[1] "Febrero"

[[7]]
[1] "Marzo"
```

2.3.5. Convertir lista en vector

La función `unlist` toma una lista como entrada y produce un vector.

```
lista <- list(letters[1:5], 1:5)
lista
[[1]]
[1] "a" "b" "c" "d" "e"

[[2]]
[1] 1 2 3 4 5

# Convertir lista en vector
v <- unlist(lista)
v
[1] "a" "b" "c" "d" "e" "1" "2" "3" "4" "5"
```

2.4. FACTORES

Los factores son objetos de datos que se utilizan para *clasificar* los datos y almacenarlos como *niveles*. Son útiles en las variables que tienen un número limitado de valores únicos. Internamente *R* va a almacenar los valores como enteros.

Los factores se crean utilizando la función factor, tomando un vector como entrada. Dicho vector debe ser de *tipo* carácter o entero.

```
# Creamos un vector
v <- c("hombre", "mujer", "mujer", "hombre", "mujer", "hombre", "mujer")
v
[1] "hombre" "mujer" "mujer" "hombre" "mujer" "hombre" "mujer"

is.factor(v)
[1] FALSE

# Crear factor
v <- factor(v)
v
[1] hombre mujer mujer hombre mujer hombre mujer
Levels: hombre mujer

is.factor(v)
[1] TRUE

# Niveles
levels(v)
[1] "hombre" "mujer"

# Creamos un vector con los 12 meses del año
meses <- c("Ene", "Feb", "Mar", "Abr", "May", "Jun", "Jul", "Ago", "Sep", "Oct", "Nov", "Dic")

# Crear factor
factor(meses)
[1] Ene Feb Mar Abr May Jun Jul Ago Sep Oct Nov Dic
Levels: Abr Ago Dic Ene Feb Jul Jun Mar May Nov Oct Sep
```

Comprobamos que los niveles del factor están ordenados por *orden alfabético*. Podemos crear un factor en el que definimos los *niveles* (levels) de antemano.

```
# Crear factor, definiendo los niveles, tal y como están en el vector "meses"
meses <- factor(meses, levels = meses)
meses
[1] Ene Feb Mar Abr May Jun Jul Ago Sep Oct Nov Dic
Levels: Ene Feb Mar Abr May Jun Jul Ago Sep Oct Nov Dic
```

2.5. DATAFRAMES

Un dataframe es una tabla (estructura de datos bidimensional), en la que cada columna contiene valores de una variable y cada fila contiene un valor de cada columna. Las filas representan *observaciones* y las columnas representan *variables*.

En un dataframe los nombres de las columnas son obligatorios. Las columnas pueden ser de diferente tipo (carácter, entero, numérico de doble precisión, lógico, factor, etc.). Todos los elementos de una columna son del mismo tipo.

2.5.1. Crear

Podemos crear un dataframe mediante la función data.frame.

En el siguiente ejemplo, creamos un dataframe desde una matriz.

```
# Crear un dataframe desde una matriz
m <- matrix(1:12, nrow = 4)
```

```
colnames(m) <- c("col1", "col2", "col3")
df <- data.frame(m)
df
```

	col1	col2	col3
1	1	5	9
2	2	6	10
3	3	7	11
4	4	8	12

Veamos un ejemplo más completo.

```
class(df$sueldo)
[1] "numeric"

typeof(df$sueldo)
[1] "double"

class(df$alta)
[1] "Date"

typeof(df$alta)
[1] "double"
```

2.5.2. Seleccionar elementos

Los elementos se seleccionan de manera similar a lo ya visto con matrices y listas. Veamos unos ejemplos:

```
df[1,2]
[1] "Juan"

df[c(3,5),c(2,4)]
  nombre      alta
  3 Luis 2014-11-15
  5 Isabel 2015-03-27

df[, "nombre"]
[1] "Juan" "Ana" "Luis" "Pedro" "Isabel"

df[1:3, "sueldo"]
[1] 1200 1400 1500

df$sueldo
[1] 1200 1400 1500 1750 1500

df$nombre[1:3]
[1] "Juan" "Ana" "Luis"
```

2.5.3. Manipular dataframes

```
# Actualizar una celda del dataframe
df$nombre[1] <- "Juan Carlos"
```

Además de actualizar celdas concretas de un dataframe, también podemos *añadir*, *eliminar* y *actualizar* filas y columnas.

Columnas

```
# Añadir las columnas: "dpto" y "numhijos"
df$dpto <- c("Informática", "Contabilidad", "Informática", "Recursos Humanos", "Contabilidad")
df$numhijos <- c(0, 1, 0, 2, 1)
df

  id      nombre sueldo      alta          dpto numhijos
1  1 Juan Carlos   1200 2012-01-01 Informática       0
2  2        Ana    1400 2013-09-23 Contabilidad       1
3  3        Luis    1500 2014-11-15 Informática       0
4  4        Pedro   1750 2014-05-11 Recursos Humanos     2
5  5      Isabel   1500 2015-03-27 Contabilidad       1
```

```
# Actualizar la columna "sueldo", con un incremento del 10%
df$sueldo <- df$sueldo + (df$sueldo * 0.1)
df
   id    nombre sueldo      alta        dpto numhijos
1 1 Juan Carlos  1320 2012-01-01 Informática 0
2 2 Ana       1540 2013-09-23 Contabilidad 1
3 3 Luis      1650 2014-11-15 Informática 0
4 4 Pedro     1925 2014-05-11 Recursos Humanos 2
5 5 Isabel    1650 2015-03-27 Contabilidad 1

# Eliminar la columna "numhijos"
df$numhijos <- NULL
df
   id    nombre sueldo      alta        dpto
1 1 Juan Carlos  1320 2012-01-01 Informática
2 2 Ana       1540 2013-09-23 Contabilidad
3 3 Luis      1650 2014-11-15 Informática
4 4 Pedro     1925 2014-05-11 Recursos Humanos
5 5 Isabel    1650 2015-03-27 Contabilidad
```

También se puede emplear la función cbind para añadir columnas a un dataframe.

Filas

Para poder añadir nuevas filas a un dataframe, la *estructura* de los nuevos datos debe ser la misma al dataframe existente. Para hacer esto, utilizamos la función rbind.

```
# Añadir una fila
df <- rbind(df, list(6, "Marcos", 1300, "2013-09-28", "Contabilidad"))
df
   id    nombre sueldo      alta        dpto
1 1 Juan Carlos  1320 2012-01-01 Informática
2 2 Ana       1540 2013-09-23 Contabilidad
3 3 Luis      1650 2014-11-15 Informática
4 4 Pedro     1925 2014-05-11 Recursos Humanos
5 5 Isabel    1650 2015-03-27 Contabilidad
6 6 Marcos    1300 2013-09-28 Contabilidad
```

En el siguiente ejemplo, vamos a explicar cómo unir dos dataframes.

```
# Creamos un segundo dataframe con la misma estructura
df2 <- data.frame(
  id = 7:9,
  nombre = c("Marta", "Pablo", "Lucía"),
  sueldo = c(1200, 1450, 1800),
  alta = as.Date(c("2013-05-21", "2013-07-30", "2014-06-17")),
  dpto = c("Informática", "Recursos Humanos", "Contabilidad"),
  stringsAsFactors = FALSE)

# Unir ambos dataframes
df3 <- rbind(df, df2)
df3
   id    nombre sueldo      alta        dpto
1 1 Juan Carlos  1320 2012-01-01 Informática
2 2 Ana       1540 2013-09-23 Contabilidad
3 3 Luis      1650 2014-11-15 Informática
4 4 Pedro     1925 2014-05-11 Recursos Humanos
5 5 Isabel    1650 2015-03-27 Contabilidad
6 6 Marcos    1300 2013-09-28 Contabilidad
7 7 Marta     1200 2013-05-21 Informática
```

```
8 8      Pablo  1450 2013-07-30 Recursos Humanos
9 9      Lucía  1800 2014-06-17 Contabilidad
```

Factores

Es muy habitual crear *factores* en ciertas columnas de un dataframe.

```
df3$dpto <- factor(df3$dpto)
df3$dpto
[1] Informática      Contabilidad      Informática      Recursos Humanos
[5] Contabilidad     Contabilidad     Informática      Recursos Humanos
[9] Contabilidad
Levels: Contabilidad Informática Recursos Humanos
```

Ordenar dataframes

Las filas de un dataframe se pueden ordenar por columnas, mediante la función `order`.

```
# Ordenar por la columna "nombre"
df3[order(df3$nombre),]
  id    nombre sueldo      alta      dpto
  2 2      Ana  1540 2013-09-23  Contabilidad
  5 5      Isabel  1650 2015-03-27  Contabilidad
  1 1      Juan Carlos  1320 2012-01-01  Informática
  9 9      Lucía  1800 2014-06-17  Contabilidad
  3 3      Luis  1650 2014-11-15  Informática
  6 6      Marcos  1300 2013-09-28  Contabilidad
  7 7      Marta  1200 2013-05-21  Informática
  8 8      Pablo  1450 2013-07-30 Recursos Humanos
  4 4      Pedro  1925 2014-05-11 Recursos Humanos

# Ordenar por la columna "sueldo" (orden decreciente)
df3[order(df3$sueldo, decreasing=TRUE),]
  id    nombre sueldo      alta      dpto
  4 4      Pedro  1925 2014-05-11 Recursos Humanos
  9 9      Lucía  1800 2014-06-17  Contabilidad
  3 3      Luis  1650 2014-11-15  Informática
  5 5      Isabel  1650 2015-03-27  Contabilidad
  2 2      Ana  1540 2013-09-23  Contabilidad
  8 8      Pablo  1450 2013-07-30 Recursos Humanos
  1 1      Juan Carlos  1320 2012-01-01  Informática
  6 6      Marcos  1300 2013-09-28  Contabilidad
  7 7      Marta  1200 2013-05-21  Informática

# Ordenar por las columnas "dpto" y "sueldo"
df3[order(df3$dpto, df3$sueldo),]
  id    nombre sueldo      alta      dpto
  6 6      Marcos  1300 2013-09-28  Contabilidad
  2 2      Ana  1540 2013-09-23  Contabilidad
  5 5      Isabel  1650 2015-03-27  Contabilidad
  9 9      Lucía  1800 2014-06-17  Contabilidad
  7 7      Marta  1200 2013-05-21  Informática
  1 1      Juan Carlos  1320 2012-01-01  Informática
  3 3      Luis  1650 2014-11-15  Informática
  8 8      Pablo  1450 2013-07-30 Recursos Humanos
  4 4      Pedro  1925 2014-05-11 Recursos Humanos
```

2.5.4. Importar/Exportar datos externos

Podemos leer datos que están almacenados fuera del entorno de R. R puede leer y escribir archivos de muy diversos formatos, como: CSV, Excel, XML, SPSS, SAS,...

2.5.4.1. Directorio de Trabajo

Podemos verificar cuál es el *directorio de trabajo* actual en *R*, mediante la función `getwd`.

También se puede establecer un nuevo *directorio de trabajo* en *R*, utilizando la función `setwd`.

```
# Obtener el "directorio de trabajo" actual
getwd() # Por defecto en "Windows", la carpeta "Documentos"

# Cambiar el "directorio de trabajo"
setwd("C:\mi_ruta\carpeta") # No válido
setwd("C:/mi_ruta/carpeta") # Válido
```

2.5.4.2. Import Dataset

Para utilizar esta herramienta, seleccionar *Import Dataset* en el menú *File*, o hacer clic en el ícono *Import Dataset* de la pestaña *Environment*. Consideraciones a tener en cuenta:

- Si se va a importar un archivo de *texto* utilizando el paquete `readr`¹², es necesario que dicho paquete esté instalado. Si no estuviera instalado, podemos utilizar este comando `install.packages("readr")`.
- Si se va a importar un archivo *Excel* utilizando el paquete `readxl`¹³, es necesario que dicho paquete esté instalado. Si no estuviera instalado, podemos utilizar este comando `install.packages("readxl")`.
- Si se va a importar un archivo de *SPSS*, *SAS* o *Stata* es necesario tener instalado el paquete `haven`¹⁴. Si no estuviera instalado, podemos utilizar este comando `install.packages("haven")`.

2.5.4.3. Archivos CSV

Mediante las funciones `read.csv` o `read.table` (del paquete básico) se puede **leer** un *archivo CSV* (u otros tipos de archivos de texto). Una vez analizado el archivo, creará un *dataframe* con los datos.

Los *argumentos* más importantes de la función `read.csv`, son los siguientes:

- *file*: Nombre del archivo (si fuera preciso, se indica la ruta completa). Si no indicásemos la ruta completa, buscará el archivo en el *directorio de trabajo* actual.
- *header*: ¿La primera línea del archivo contiene los nombres de las columnas? `TRUE` (por defecto) o `FALSE`.
- *sep*: Cuál es el carácter *delimitador* entre los campos (columnas), por defecto `sep = ","`.
- *quote*: Cuál es el carácter para *entrecomillar* las cadenas de texto, por defecto *dobles comillas*. Si las cadenas de texto no estuvieran entrecomilladas, indicar `quote = ""`.
- *dec*: Separador *decimal*. Usar el *punto* o la *coma*, por defecto `dec = ","`.
- *stringsAsFactors*: ¿Convertir los vectores tipo carácter a *fatores*? `TRUE` (por defecto) o `FALSE`.
- *encoding*: ¿Cuál es el formato del archivo de texto? Es decir, la *codificación* o juego de caracteres utilizado: "ASCII", "UTF-8", "ISO-8859-1",...

¹² Hadley Wickham, Jim Hester and Romain Francois (2018). `readr`: Read Rectangular Text Data. R package version 1.3.1. <http://CRAN.R-project.org/package=readr>

¹³ Hadley Wickham and Jennifer Bryan (2018). `readxl`: Read Excel Files. R package version 1.2.0. <https://CRAN.R-project.org/package=readxl>

¹⁴ Hadley Wickham and Evan Miller (2018). `haven`: Import and Export 'SPSS', 'Stata' and 'SAS' Files. R package version 2.0.0. <https://CRAN.R-project.org/package=haven>

```
# Leemos un "archivo csv" y lo guardamos en el dataframe "df"
df <- read.csv("ejemplo.csv")
```

También se puede leer un archivo ubicado en un servidor, indicando la URL.

```
# URL del archivo a leer. Carácter delimitador: tabulador
df <- read.csv("http://datanalytics.com/uploads/datos_treemap.txt", sep = "\t")
head(df) # Mostrar las 6 primeras filas (por defecto)

# URL del archivo a leer.
df <- read.table("https://raw.githubusercontent.com/oscarperpinan/R/master/data/aranez.csv", sep=",", header=TRUE)
tail(df, 3) # Mostrar las 3 últimas filas
```

Si desea más información para leer archivos de texto, escriba en la consola: ?read.table.

Mediante la función write.csv se puede *escribir* un *archivo CSV* (u otros tipos de archivos de texto).

```
# Partiendo del dataframe "df", escribir un archivo de texto
# Sin entrecollar. Sin nombres de filas
write.csv(df, "ejemplo.txt", quote = FALSE, row.names = FALSE)
```

Si desea más información para escribir archivos de texto, escriba en la consola: ?write.table.

2.5.4.4. Archivos Excel

El paquete xlsx¹⁵ permite *leer* y *escribir* archivos en formato *Excel*. Es más completo y eficiente que el paquete readxl (que solo permite leer). Este paquete no forma parte de la instalación de R. Podemos utilizar este código para instalar dicho paquete:

```
install.packages("xlsx")
```

Mediante la función library cargamos en R el paquete especificado.

```
# Cargar paquete "xlsx" (se presupone que está instalado)
library(xlsx)
# Ahora podemos utilizar las funciones propias del paquete "xlsx"

# Leemos un "archivo Excel" y lo guardamos en el dataframe "df"
df <- read.xlsx("ejemplo.xlsx")

# Partiendo del dataframe "df", escribir un archivo Excel
write.xlsx(df, "ejemplo2.xlsx", row.names = FALSE)
```

Si desea más información sobre las funciones read.xlsx o write.xlsx, escriba en la consola: ?read.xlsx o ?write.xlsx.

3. REALIZAR CONSULTAS

3.1. CONDICIONES SIMPLES

Las condiciones simples generalmente se establecen mediante los *operadores de comparación*, que son los siguientes:

¹⁵ Adrian A. Dragulescu (2018). xlsx: Read, write, format Excel 2007 and Excel 97/2000/XP/2003 files. R package version 0.6.1. <https://CRAN.R-project.org/package=xlsx>

Igual: ==. Mayor: >. Menor: <. Mayor o igual: >=. Menor o igual: <=. Distinto/No Igual: !=.

El resultado de la comparación será un valor lógico o booleano: *cierto o falso*.

Veamos algunos ejemplos.

```
x <- seq(-1, 1, .25)
x
[1] -1.00 -0.75 -0.50 -0.25  0.00  0.25  0.50  0.75  1.00

x < 0
[1] TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE

x >= 0.5
[1] FALSE FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE

x == -0.5
[1] FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE

# "Extraer" los elementos que cumplan la condición
v <- x[x < 0]
v
[1] -1.00 -0.75 -0.50 -0.25

m <- matrix(1:9, 3)
m
[,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
v <- m[m >= 6]
v
[1] 6 7 8 9

# Comparar dos vectores entre sí
x <- c(1,9,14,12,0)
y <- c(5,3,25,12,-5)
x > y
[1] FALSE TRUE FALSE FALSE TRUE

x == y
[1] FALSE FALSE FALSE TRUE FALSE

x != y
[1] TRUE TRUE TRUE FALSE TRUE
```

El operador %in% es propio de R, y se utiliza para identificar si un elemento de un vector forma parte de otro vector. Devuelve un vector lógico.

```
frutas <- c("Manzana", "Uva", "Pera", "Naranja", "Cereza")

c("Pera", "Fresa") %in% frutas
[1] TRUE FALSE

frutas2 <- c("Fresa", "Pera", "Uva", "Plátano")
iguales <- frutas2[frutas2 %in% frutas]
iguales
[1] "Pera" "Uva"

distintos <- frutas2[!(frutas2 %in% frutas)]
distintos
```

[1] "Fresa" "Plátano"

Dado que una condición es una *expresión lógica o booleana* (cierto/falso), también se podrían utilizar aquellas funciones que devuelven un valor lógico, como: is.integer, is.numeric, is.na, is.infinite, etc.

```
x <- c(10:7,4,NA,0,5/0)
x
[1] 10  9   8   7   4  NA   0 Inf

is.integer(x)
[1] FALSE

is.numeric(x)
[1] TRUE

is.na(x)
[1] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE

is.finite(x)
[1] TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE FALSE

is.infinite(x)
[1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
```

3.2. CONDICIONES MÚLTIPLES

Se utiliza el *álgebra booleana* para establecer condiciones múltiples. Los **operadores lógicos** permiten agrupar expresiones lógicas (condiciones). Los *operadores lógicos* son los siguientes:

- Operador **AND**: & Operador "Y" lógico: devuelve el valor booleano TRUE, si **ambas** condiciones son TRUE; en caso contrario, devuelve FALSE.
- Operador **OR**: | Operador "O" lógico: devuelve el valor booleano TRUE, si **una** de las condiciones es TRUE; en caso contrario, devuelve FALSE.
- Operador **NOT**: ! Operador "de negación" lógica: devuelve el valor booleano TRUE, si la condición es FALSE; si la condición es TRUE, devuelve FALSE.

```
x <- seq(-1, 1, .25)
x
[1] -1.00 -0.75 -0.50 -0.25  0.00  0.25  0.50  0.75  1.00
```

Dadas estas dos condiciones:

1. x > 0
2. x < 0.5

Si queremos obtener aquellos elementos que cumplan *ambas* condiciones: usaremos el operador &.

```
# (Qué elementos son mayores que cero) Y (Qué elementos son menores que 0.5)
x > 0 & x < 0.5
[1] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
```

Si queremos obtener aquellos elementos que cumplan *una* de las dos condiciones: usaremos el operador |.

```
# (Qué elementos son mayores que cero) 0 (Qué elementos son menores que 0.5)
x > 0 | x < 0.5
[1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

Si queremos obtener aquellos elementos que *no* cumplan cierta condición: usaremos el operador !.

```
# Qué elementos son mayores que cero
x > 0
[1] FALSE FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE

# Qué elementos no son mayores que cero
!(x > 0)
[1] TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE

# "Extraer" los elementos que cumplan la condición múltiple
v <- x[x > 0.75 | x < 0.5]
v
[1] -1.00 -0.75 -0.50 -0.25  0.00  0.25  1.00
```

3.3. FILTRAR DATAFRAMES

A continuación, vamos a realizar varias consultas en un dataframe, es decir, extraer aquellas filas que cumplan ciertas condiciones. También se conoce como *filtrar*.

En los siguientes ejemplos, vamos a utilizar el *dataset* iris. R contiene varios datasets para practicar.

```
iris <- iris
head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
 1          5.1         3.5          1.4         0.2  setosa
 2          4.9         3.0          1.4         0.2  setosa
 3          4.7         3.2          1.3         0.2  setosa
 4          4.6         3.1          1.5         0.2  setosa
 5          5.0         3.6          1.4         0.2  setosa
 6          5.4         3.9          1.7         0.4  setosa

consulta <- iris[iris$Sepal.Length > 7,]
head(consulta)
  Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
103          7.1         3.0          5.9         2.1 virginica
106          7.6         3.0          6.6         2.1 virginica
108          7.3         2.9          6.3         1.8 virginica
110          7.2         3.6          6.1         2.5 virginica
118          7.7         3.8          6.7         2.2 virginica
119          7.7         2.6          6.9         2.3 virginica

consulta <- iris[iris$Sepal.Width < 3 & iris$Species == "setosa",]
head(consulta)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
 9           4.4         2.9          1.4         0.2  setosa
42           4.5         2.3          1.3         0.3  setosa

consulta <- iris[iris$Sepal.Width < 3 & iris$Species %in% c("setosa","virginica"),]
head(consulta)
  Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
 9           4.4         2.9          1.4         0.2  setosa
42           4.5         2.3          1.3         0.3  setosa
102          5.8         2.7          5.1         1.9 virginica
```

```

104      6.3      2.9      5.6      1.8 virginica
107      4.9      2.5      4.5      1.7 virginica
108      7.3      2.9      6.3      1.8 virginica

consulta <- iris[iris$Sepal.Length < 4.5 | iris$Sepal.Width < 3,]
head(consulta)
  Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
  9           4.4       2.9       1.4       0.2   setosa
 14          4.3       3.0       1.1       0.1   setosa
 39          4.4       3.0       1.3       0.2   setosa
 42          4.5       2.3       1.3       0.3   setosa
 43          4.4       3.2       1.3       0.2   setosa
 54          5.5       2.3       4.0       1.3 versicolor

```

Para filtrar es muy útil la función `subset`, obteniendo un *subconjunto* de un dataframe.

```

consulta <- subset(iris, Sepal.Length > 7)
head(consulta,3)
  Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
 103         7.1       3.0       5.9       2.1 virginica
 106         7.6       3.0       6.6       2.1 virginica
 108         7.3       2.9       6.3       1.8 virginica

consulta <- subset(iris, Sepal.Length > 7, select = c("Sepal.Length", "Sepal.Width"))
head(consulta,3)
  Sepal.Length Sepal.Width
 103         7.1       3.0
 106         7.6       3.0
 108         7.3       2.9

# MÁXIMA longitud de un pétalo (Petal.Length)
max(iris$Petal.Length)
[1] 6.9

# ¿Qué especie tiene el pétalo con mayor longitud?
subset(iris, Petal.Length == max(Petal.Length))
  Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
 119         7.7       2.6       6.9       2.3 virginica

```

4. FUNCIONES

Una función es un grupo de sentencias o instrucciones organizadas conjuntamente para realizar una *tarea específica*. R tiene un gran número de funciones *integratedas* y el usuario puede crear sus *propias* funciones.

En R, una función es un *objeto*. El intérprete de R puede pasar el control a la función, junto con los *argumentos* que puedan ser necesarios, para que la función realice las acciones encomendadas.

La función, a su vez, realiza su tarea y devuelve el control al intérprete, así como cualquier *resultado* que se pueda almacenar en otros objetos.

4.1. DEFINIR UNA FUNCIÓN

Una función en R se crea utilizando la palabra clave `function`. La *sintaxis* básica para definir una función en R es la siguiente:

```
nombre_funcion <- function(arg_1, arg_2, ...) {
  Cuerpo de la función
}
```

Una función se compone de:

- *Nombre* de la función: este es el nombre asignado a la función. Se almacena en el entorno *R* como un objeto con este nombre.
- *Argumentos*: Un argumento es un parámetro. Cuando se llama a una función, se pasa un valor como argumento. Los argumentos son opcionales, es decir, una función puede no contener argumentos. Los argumentos pueden tener valores por defecto.
- *Cuerpo* de la función: el cuerpo de la función contiene un grupo de instrucciones que definen lo que hace la función.
- *Valor devuelto*: el valor devuelto por la función es la última instrucción en el cuerpo de la función.

R tiene muchas *funciones incorporadas* (agrupadas en paquetes), que se pueden llamar directamente desde el código creado por el usuario. También podemos crear y utilizar nuestras propias funciones, denominadas *funciones definidas por el usuario*.

4.2. FUNCIONES INCORPORADAS

Algunas de estas funciones ya las hemos visto anteriormente. Cada *paquete de R* contiene sus propias funciones, para que sean utilizadas por el usuario según sus necesidades.

Varios *paquetes* vienen preinstalados en *R* y se cargan al iniciar la sesión, como por ejemplo: base, datasets, utils, graphics, stats,...

Veamos algunas funciones más.

```
# Funciones matemáticas
min(iris$Sepal.Length)      # Mínimo
max(iris$Sepal.Length)       # Máximo
range(iris$Sepal.Length)     # Rango (mínimo y máximo)
sum(iris$Sepal.Length)        # Suma
mean(iris$Sepal.Length)       # Media
median(iris$Sepal.Length)     # Mediana
quantile(iris$Sepal.Length)   # Cuantiles
var(iris$Sepal.Length)        # Varianza
sd(iris$Sepal.Length)         # Desviación estándar

sample(1:5) # Muestra aleatoria
sessionInfo() # Información sobre la sesión actual
ls() # lista los objetos guardados en la memoria
rm(vector) # borra el objeto "vector" de la memoria
newobject <- edit(cars) # edita y crea un objeto en la memoria
fix(newobject)           # edita y guarda el contenido de un objeto
```

Guía de referencia sobre las funciones más importantes en *R*.

- [R Reference Card](#): Guía muy útil.
- [R Reference Card 2.0](#): Una ampliación de la guía anterior.

4.3. FUNCIONES DEFINIDAS POR EL USUARIO

El usuario puede crear sus propias funciones. Una vez creadas, se pueden usar como las funciones integradas en R. A continuación, se muestra un ejemplo de cómo se *crea* y *utiliza* una función.

Vamos a crear la función elevar para realizar potencias. Esta función tiene dos *argumentos*: *x* e *y*, y nos devolverá el resultado de *xy*.

```
# Crear la función "elevar"
elevar <- function(x, y){
  x ^ y
}
```

Ahora queremos aplicar la función elevar a una secuencia de números del 1 al 10, utilizando como potencia el cuadrado.

```
# Llamar a la función "elevar"
elevar(x = 1:10, y = 2)
[1]  1  4  9 16 25 36 49 64 81 100
```

En la llamada a una función, los argumentos pueden proporcionarse en el mismo orden en que se define la función; o en un orden diferente, pero indicando el *nombre* de los argumentos.

Se podría llamar a la función elevar de cualquiera de estas formas:

```
elevar(1:10, 2)

elevar(y = 2, x = 1:10)

elevar(1:10, y = 2)
```

También se puede crear una función indicando los valores *por defecto* de los argumentos, que se emplearán cuando no se proporcione el argumento.

```
# Crear la función "elevar"
elevar <- function(x, y = 2){
  x ^ y
}
# Llamar a la función "elevar" sin proporcionar el argumento "y" (por defecto, y = 2)
elevar(1:10)
[1]  1  4  9 16 25 36 49 64 81 100
```

Veamos otro ejemplo, una función que calcula la media, mediana y desviación estándar.

```
calcular <- function(x){
  media <- mean(x)
  mediana <- median(x)
  desviacion <- sd(x)
  c(media, mediana, desviacion)
}

calcular(1:10)
[1] 5.50000 5.50000 3.02765
```

5. ESTRUCTURAS DE CONTROL

En programación, las *estructuras de control* permiten modificar el *flujo* de ejecución de las instrucciones de un programa.

Las *estructuras de control* se pueden dividir en dos grupos:

- *Sentencias condicionales*: redirigen el curso de la acción, según la evaluación de una condición.
- *Bucles*: son estructuras de control iterativas o de repetición. Como ejecutar un conjunto de instrucciones cierto número de veces, o repetir un bloque de instrucciones mientras se cumpla una condición.

5.1. SENTENCIAS CONDICIONALES

En programación, una *sentencia condicional* es una instrucción o grupo de instrucciones que se pueden ejecutar o no, dependiendo de si la condición se evalúa como *cierta* o *falsa*.

5.1.1. *Sentencia if*

Una *sentencia if* consta de una *condición* (expresión booleana), seguida de una o más *instrucciones*.

```
if(condicion) {  
    // bloque de código que se ejecuta si la condición es verdadera  
}
```

Si la condición se evalúa como *verdadera*, entonces se ejecutará el bloque de código de la *sentencia if*. Si la condición se evalúa como *falsa*, se ejecutará el código que haya después del final de la *sentencia if* (después de la llave de cierre).

Veamos un ejemplo.

```
x <- 5L  
if(is.integer(x)) {  
    cat(x, "es un entero")  
}  
5 es un entero
```

5.1.2. *Sentencia if...else*

Una *sentencia if* puede ir seguida de una *sentencia else* que se ejecuta cuando la condición es *falsa*.

```
if(condicion) {  
    // bloque de código que se ejecuta si la condición es verdadera  
} else {  
    // bloque de código que se ejecuta si la condición es falsa  
}
```

Si la condición se evalúa como *verdadera*, entonces se ejecutará el bloque de código de la *sentencia if*. Si la condición se evalúa como *falsa*, se ejecutará el bloque de código de la *sentencia else*.

Veamos un ejemplo (el operador `%%` divide y devuelve el *resto* de la división).

```
x <- sample(1:5, size = 1)
if(x%%2 == 0) {
  cat(x, "es par")
} else {
  cat(x, "es impar")
}
2 es par
```

En R también se puede utilizar la función `ifelse`.

```
x <- sample(1:5, size = 1)
x
[1] 5
ifelse(x%%2 == 0, "es par", "es impar")
[1] "es impar"
```

5.1.3. Sentencias anidadas

Una *sentencia if* puede ir seguida de una *sentencia else if...else*.

```
if(condicion1) {
  // bloque de código que se ejecuta si "condicion1" es verdadera
} else if(condicion2) {
  // bloque de código que se ejecuta si "condicion2" es verdadera
} else if(condicion3) {
  // bloque de código que se ejecuta si "condicion3" es verdadera
} else {
  // bloque de código que se ejecuta si "ninguna" condición es verdadera
}
```

El siguiente ejemplo mide la “sensación térmica”, respecto a los siguientes rangos de “temperatura”:

- $[-10, 5]$: Mucho frío.
- $[5, 15]$: Frío.
- $[15, 25]$: Normal.
- $[25, 30]$: Calor.
- $[30, 45]$: Mucho calor.

```
temperatura <- sample(-10:45, size = 1)
temperatura
[1] 6

if(temperatura < 5) {
  print("Mucho frío")
} else if(temperatura < 15) {
  print("Frío")
} else if(temperatura < 25) {
  print("Normal")
} else if(temperatura < 30) {
  print("Calor")
} else {
  print("Mucho calor")
}
[1] "Frío"
```

Similar al ejemplo anterior, pero empleando una función.

```
sensaciontermica <- function(temperatura){
  if(temperatura < 5) {
```

```

    "Mucho frío"
} else if(temperatura < 15) {
    "Frío"
} else if(temperatura < 25) {
    "Normal"
} else if(temperatura < 30) {
    "Calor"
} else {
    "Mucho calor"
}
}

n <- sample(-10:45, size = 1)
n
[1] 44
sensaciontermica(n)
[1] "Mucho calor"

```

5.2. BUCLES

Es posible que necesitemos ejecutar un bloque de código varias veces. En *R* básicamente hay dos tipos de bucles:

- **for**: Repite un bloque de código un *número* concreto de veces.
- **while**: Repite un bloque de código *mientras* se cumpla una *condición*.

5.2.1. Bucle For

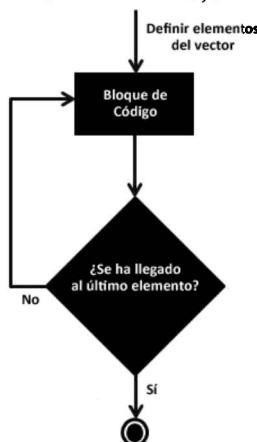
Un *bucle For* es una estructura de control de repetición, permite que un bucle se ejecute un *número* específico de veces.

```

for (variable in vector) {
    // bloque de código
}

```

FIGURA 2. DIAGRAMA DE FLUJO POR BUCLE FOR



Fuente: Elaboración propia

Vamos a crear un *bucle For* que se repetirá 5 veces.

```

for (i in 1:5) {
    print(paste("Iteración nº", i))
}

```

```
[1] "Iteración nº 1"
[1] "Iteración nº 2"
[1] "Iteración nº 3"
[1] "Iteración nº 4"
[1] "Iteración nº 5"
```

Veamos otro ejemplo.

```
for(i in c(2,5,10,20,50)) {
  print(paste("El cuadrado de", i, "es", i^2))
}
[1] "El cuadrado de 2 es 4"
[1] "El cuadrado de 5 es 25"
[1] "El cuadrado de 10 es 100"
[1] "El cuadrado de 20 es 400"
[1] "El cuadrado de 50 es 2500"
```

En R los *bucles For* son particularmente flexibles, ya que no se limitan a enteros para definir el vector de entrada, como en los dos ejemplos anteriores. Podemos pasar vectores de caracteres, vectores lógicos, listas o expresiones.

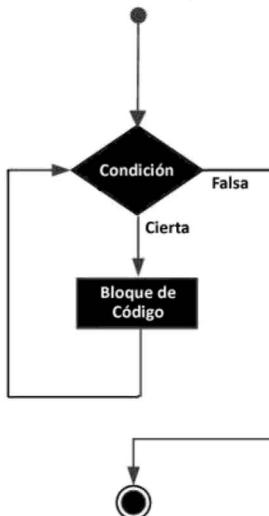
```
for (i in LETTERS[1:4]) {
  print(i)
}
[1] "A"
[1] "B"
[1] "C"
[1] "D"
```

5.2.2. Bucle While

El *bucle While* ejecuta el mismo código una y otra vez, mientras se cumpla una *condición*.

```
while (condicion) {
  // bloque de código
}
```

FIGURA 3. DIAGRAMA DE FLUJO POR BUCLE WHILE



Fuente: Elaboración propia

Veamos un ejemplo.

```
n <- 1 # Iniciar "contador"
while (n < 7) {
```

```
print(paste("El cuadrado de", n, "es", n^2))
n <- n + 1 # Actualizar "contador"
}
[1] "El cuadrado de 1 es 1"
[1] "El cuadrado de 2 es 4"
[1] "El cuadrado de 3 es 9"
[1] "El cuadrado de 4 es 16"
[1] "El cuadrado de 5 es 25"
[1] "El cuadrado de 6 es 36"
```

En el siguiente ejemplo, nos proponemos obtener los 20 primeros números de la *sucesión de Fibonacci*. La sucesión comienza con los números 0 y 1, y a partir de estos, *cada término es la suma de los dos anteriores*.

```
fibonacci <- c(0,1)
n <- 3
while (n <= 20) {
  fibonacci[n] <- fibonacci[n-1] + fibonacci[n-2]
  n <- n + 1
}
fibonacci
[1] 0 1 1 2 3 5 8 13 21 34 55 89 144 233
[15] 377 610 987 1597 2584 4181
```

En R, cuando el bloque de código de un bucle suponga mucho tiempo de procesamiento, es aconsejable utilizar una función de la *familia apply*.

6. LA FAMILIA DE FUNCIONES APPLY

Las funciones *apply*, *sapply*, *lapply* y *tapply* resultan extremadamente útiles, y permiten evitar los bucles. Su objetivo fundamental es *aplicar* (de ahí el nombre) una función a todos los elementos de un objeto. En realidad, son funciones que de alguna manera ejecutan un bucle, pero este bucle se ejecuta en código compilado, lo que hace que sea más *rápido y eficiente* que utilizar los comandos *for* o *while* que siempre deben ser interpretados. No siempre será posible sustituir un bucle por una función de la familia *apply*, pero cuando lo sea, su utilización es muy ventajosa.

apply

Aplica una función a una *matriz*. *apply(m, i, fun)*: si *i* vale 1, aplica la función *fun* a todas las filas de la matriz *m*; si *i* vale 2, la aplica a las columnas.

```
m <- matrix(1:9,nrow = 3, ncol = 3)
m
[,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9

apply(m, 1, sum) # Operar con filas
[1] 12 15 18

apply(m, 2, sum) # Operar con columnas
[1] 6 15 24
```

lapply

Aplica una función a una *lista* `lapply(l, fun)`: aplica la función *fun* a todos los elementos de la lista *l*. El resultado también es una lista.

Supongamos que tenemos una lista con las edades de los alumnos de 5 aulas y queremos determinar el número de alumnos en cada aula, así como la media de edad.

```
edades <- list(aula1=c(12,11,14,11,12,11,12,11,14,12,13,12,11),
               aula2=c(11,12,12,11,13,13,13,14,12,12,11),
               aula3=c(10,9,9,9,9,8,8,7,7,9,9,10,10,9,8,7,8,8),
               aula4=c(14,16,14,14,15,14,14,16),
               aula5=c(17,17,18,17,16,16,16))

lapply(edades,length) # Número de alumnos en cada aula
$aula1
[1] 13

$aula2
[1] 11

$aula3
[1] 18

$aula4
[1] 8

$aula5
[1] 7

lapply(edades,mean) # Media de edad en cada aula
$aula1
[1] 12

$aula2
[1] 12.18182

$aula3
[1] 8.555556

$aula4
[1] 14.625

$aula5
[1] 16.71429
```

sapply

Aplica una función a una *lista* `sapply(l, fun)`: al igual que `lapply`, aplica la función *fun* a todos los elementos de la lista *l*, pero devuelve el resultado en forma de vector.

```
sapply(edades,length) # Número de alumnos en cada aula
aula1 aula2 aula3 aula4 aula5
13    11    18     8    7

sapply(edades,mean) # Valor medio de edad en cada aula
aula1   aula2   aula3   aula4   aula5
12.000000 12.181818 8.555556 14.625000 16.714286
```

tapply

Aplica una función a un *vector* tapply(v, factor, fun): aplica la función *fun* al vector *v*, *agrupando* por los niveles definidos en un *factor*.

Por ejemplo, tenemos las edades y el sexo de un grupo de chicos y chicas, y queremos calcular la edad media para cada sexo.

```
grupo <- data.frame(edad=c(12,13,12,11,13,14,15,11),
                     sexo=c("Hombre","Mujer","Hombre","Hombre","Mujer","Hombre","Mujer",
                           "Hombre"))
levels(grupo$sexo) # La columna "sexo" es un factor
[1] "Hombre" "Mujer"

tapply(grupo$edad, grupo$sexo, mean)
      Hombre     Mujer
12.00000 13.66667
```

6.1. LAS FUNCIONES APPLY VS. BUCLES

En este apartado vamos a mostrar cómo usar las *funciones apply* en lugar de los *bucles*. Hay que tener presente que las *funciones apply* son más eficientes que los bucles.

```
# Utilizando un bucle "for"
x <- NULL
crono <- Sys.time()
for(i in 1:5) {
  x[i] <- i^10
}
Sys.time() - crono
  Time difference of 0.01682090759 secs
# Utilizando la función "sapply"
crono <- Sys.time()
y <- sapply(1:5, function(x) x^10)
Sys.time() - crono
  Time difference of 0.002928972244 secs
y
[1]      1    1024   59049  1048576  9765625
```

Ambos algoritmos hacen lo mismo, pero la función *sapply* lo realiza en menos tiempo que el bucle *for*.

También podemos crear previamente la función a utilizar, esto es útil cuando el cuerpo de la función se compone de varias instrucciones.

```
potencias2 <- function(x){
  2^x
}

y <- sapply(0:20, function(x) potencias2(x))
y
[1]      1      2      4      8     16      32      64     128
[9]    256    512   1024   2048   4096   8192  16384  32768
[17] 65536 131072 262144 524288 1048576
```

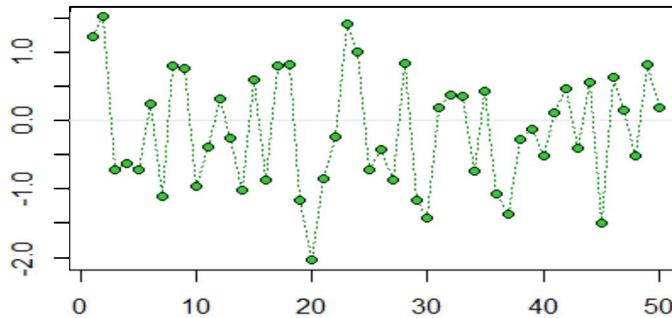
7. GRÁFICOS

Si utilizamos el comando `demo(graphics)` podemos ver muchas de las posibilidades que nos ofrece R en lo que se refiere a la generación de gráficos.

```
demo(graphics)
-----
> # Copyright (C) 1997-2009 The R Core Team
>
> require(datasets)
>
> require(grDevices); require(graphics)

> ## Here is some code which illustrates some of the differences between
> ## R and S graphics capabilities. Note that colors are generally specified
> ## by a character string name (taken from the X11 rgb.txt file) and that line
> ## textures are given similarly. The parameter "bg" sets the background
> ## parameter for the plot and there is also an "fg" parameter which sets
> ## the foreground color.
>
>
> x <- stats::rnorm(50)
> opar <- par(bg = "white")
> plot(x, ann = FALSE, type = "n")
> abline(h = 0, col = gray(.90))
> lines(x, col = "green4", lty = "dotted")
> points(x, bg = "limegreen", pch = 21)
> title(main = "Simple Use of Color In a Plot",
+       xlab = "Just a Whisper of a Label",
+       col.main = "blue", col.lab = gray(.8),
+       cex.main = 1.2, cex.lab = 1.0, font.main = 4, font.lab = 3)
```

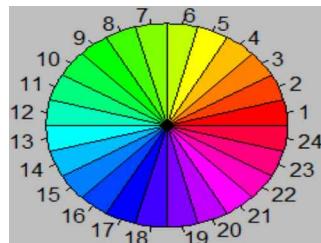
Simple Use of Color In a Plot



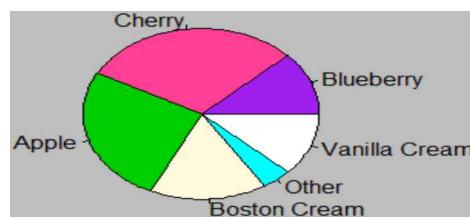
Just a Whisper of a Label

```
> ## A little color wheel. This code just plots equally spaced hues in
> ## a pie chart. If you have a cheap SVGA monitor (like me) you will
> ## probably find that numerically equispaced does not mean visually
> ## equispaced. On my display at home, these colors tend to cluster at
> ## the RGB primaries. On the other hand on the SGI Indy at work the
> ## effect is near perfect.
>
> par(bg = "gray")
> pie(rep(1,24), col = rainbow(24), radius = 0.9)
```

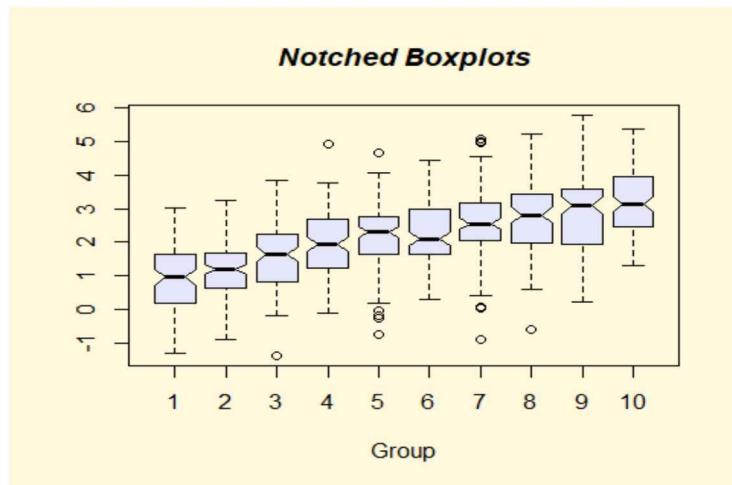
```
> title(main = "A Sample Color Wheel", cex.main = 1.4, font.main = 3)
> title(xlab = "(Use this as a test of monitor linearity)",
+       cex.lab = 0.8, font.lab = 3)
```

A Sample Color Wheel*(Use this as a test of monitor linearity)*

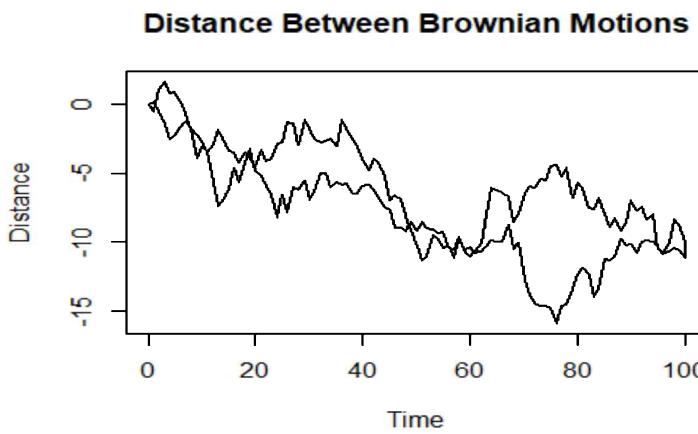
```
> ## We have already confessed to having these. This is just showing off X11
> ## color names (and the example (from the postscript manual) is pretty "cute".
>
> pie.sales <- c(0.12, 0.3, 0.26, 0.16, 0.04, 0.12)
> names(pie.sales) <- c("Blueberry", "Cherry",
+                         "Apple", "Boston Cream", "Other", "Vanilla Cream")
> pie(pie.sales,
+      col = c("purple", "violetred1", "green3", "cornsilk", "cyan", "white"))
> title(main = "January Pie Sales", cex.main = 1.8, font.main = 1)
> title(xlab = "(Don't try this at home kids)", cex.lab = 0.8, font.lab = 3)
```

January Pie Sales*(Don't try this at home kids)*

```
> ## Boxplots: I couldn't resist the capability for filling the "box".
> ## The use of color seems like a useful addition, it focuses attention
> ## on the central bulk of the data.
>
> par(bg="cornsilk")
> n <- 10
> g <- gl(n, 100, n*100)
> x <- rnorm(n*100) + sqrt(as.numeric(g))
> boxplot(split(x,g), col="lavender", notch=TRUE)
> title(main="Notched Boxplots", xlab="Group", font.main=4, font.lab=1)
```



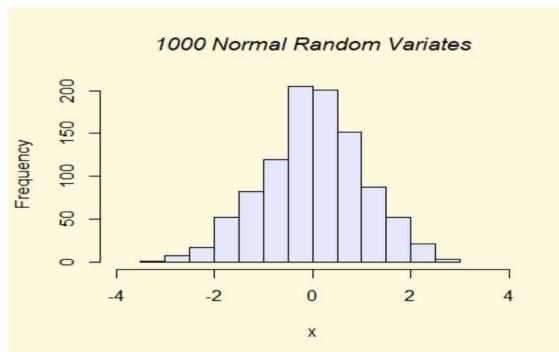
```
> ## An example showing how to fill between curves.
>
> par(bg="white")
> n <- 100
> x <- c(0,cumsum(rnorm(n)))
> y <- c(0,cumsum(rnorm(n)))
> xx <- c(0:n, n:0)
> yy <- c(x, rev(y))
> plot(xx, yy, type="n", xlab="Time", ylab="Distance")
> polygon(xx, yy, col="gray")
> title("Distance Between Brownian Motions")
```



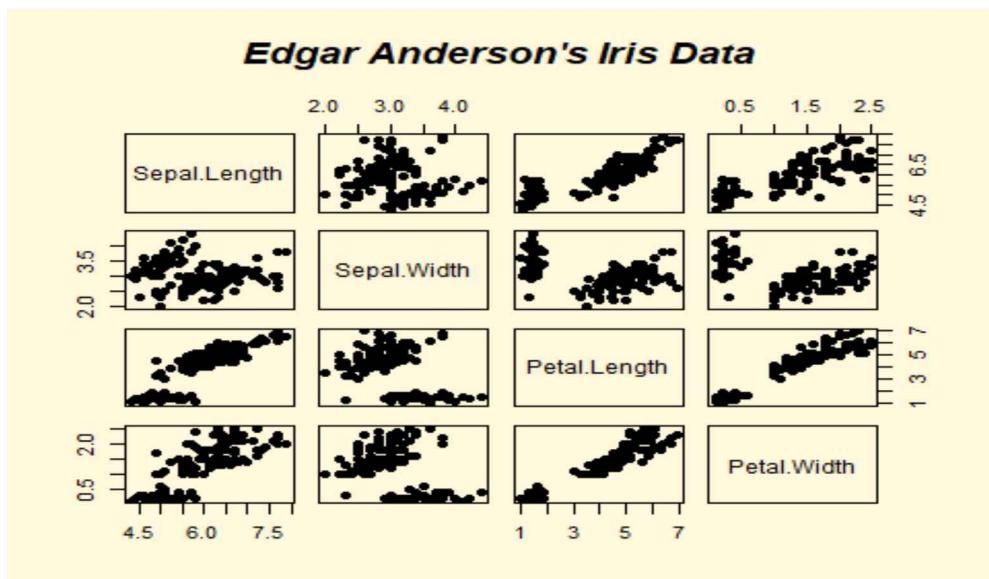
```
> ## Colored plot margins, axis labels and titles. You do need to be
> ## careful with these kinds of effects. It's easy to go completely
> ## over the top and you can end up with your lunch all over the keyboard.
> ## On the other hand, my market research clients love it.
>
> x <- c(0.00, 0.40, 0.86, 0.85, 0.69, 0.48, 0.54, 1.09, 1.11, 1.73, 2.05, 2.02)
> par(bg="lightgray")
> plot(x, type="n", axes=FALSE, ann=FALSE)
> usr <- par("usr")
> rect(usr[1], usr[3], usr[2], usr[4], col="cornsilk", border="black")
> lines(x, col="blue")
> points(x, pch=21, bg="lightcyan", cex=1.25)
> axis(2, col.axis="blue", las=1)
> axis(1, at=1:12, lab=month.abb, col.axis="blue")
> box()
> title(main= "The Level of Interest in R", font.main=4, col.main="red")
> title(xlab= "1996", col.lab="red")
```



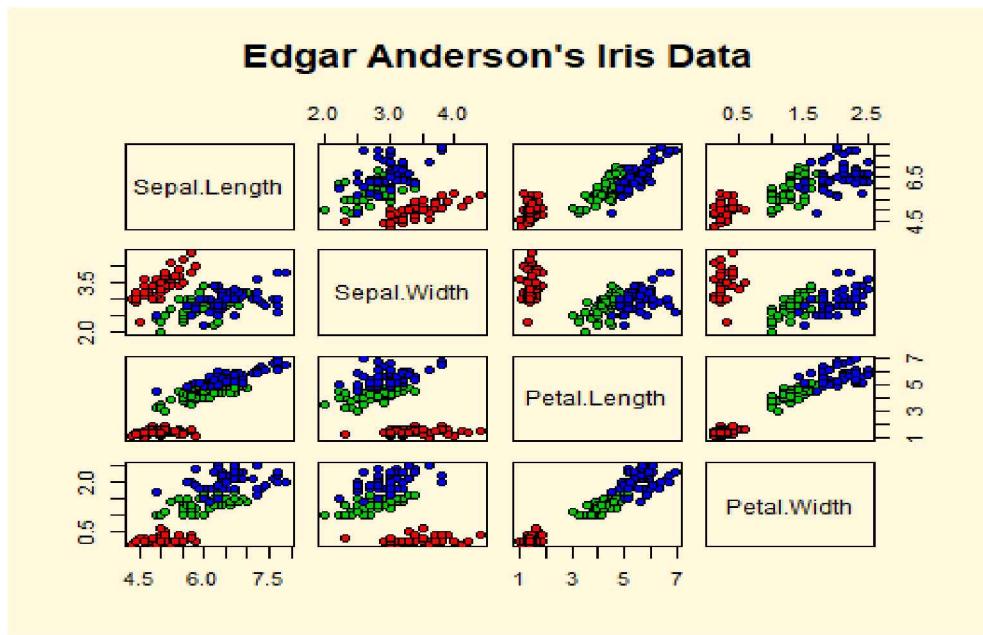
```
> ## A filled histogram, showing how to change the font used for the
> ## main title without changing the other annotation.
>
> par(bg="cornsilk")
> x <- rnorm(1000)
> hist(x, xlim=range(-4, 4, x), col="lavender", main="")
> title(main="1000 Normal Random Variates", font.main=3)
```



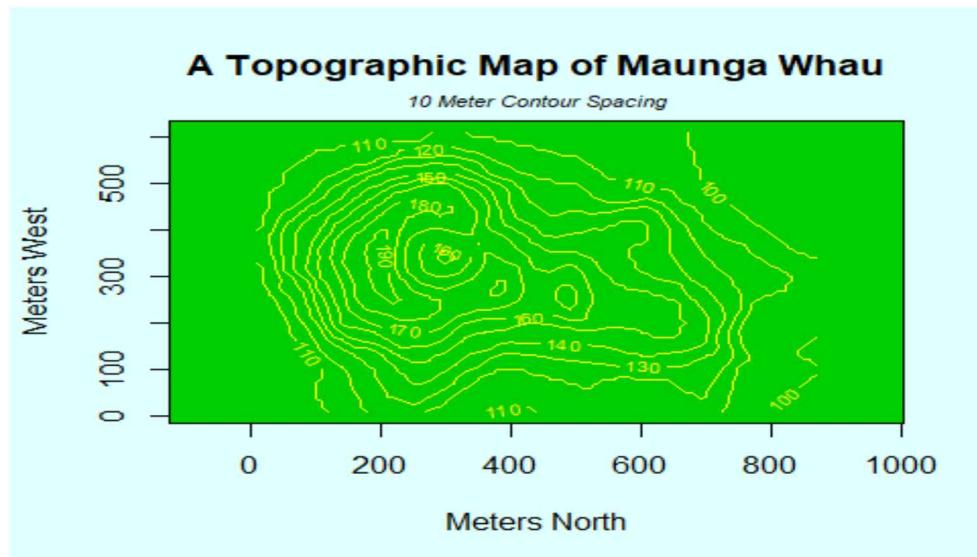
```
> ## A scatterplot matrix
> ## The good old Iris data (yet again)
>
> pairs(iris[1:4], main="Edgar Anderson's Iris Data", font.main=4, pch=19)
```



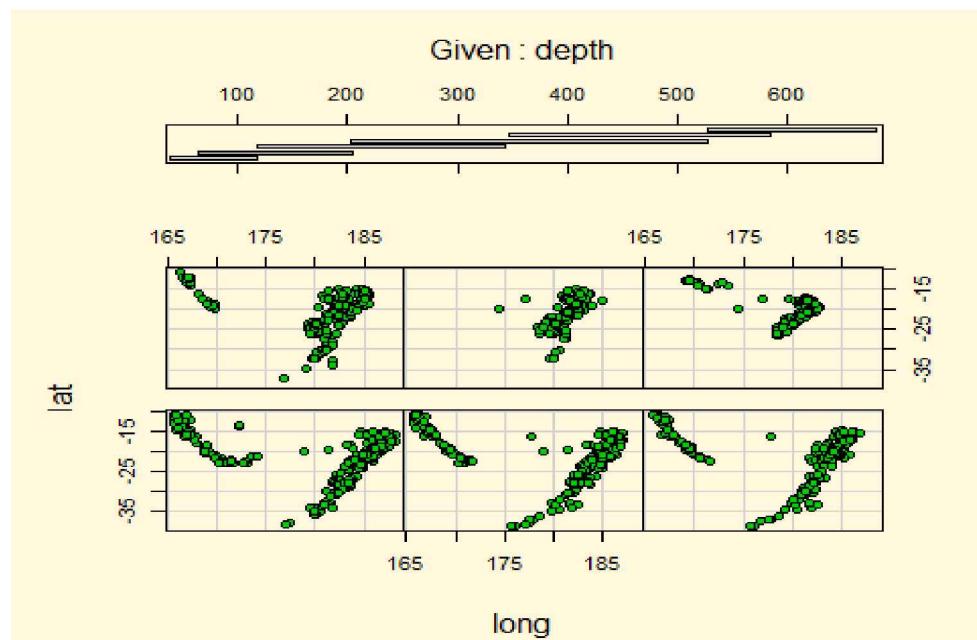
```
> pairs(iris[1:4], main="Edgar Anderson's Iris Data", pch=21,
+       bg = c("red", "green3", "blue")[unclass(iris$Species)])
```



```
> ## Contour plotting
> ## This produces a topographic map of one of Auckland's many volcanic "peaks".
>
> x <- 10*1:nrow(volcano)
> y <- 10*1:ncol(volcano)
> lev <- pretty(range(volcano), 10)
> par(bg = "lightcyan")
> pin <- par("pin")
> xdelta <- diff(range(x))
> ydelta <- diff(range(y))
> xscale <- pin[1]/xdelta
> yscale <- pin[2]/ydelta
> scale <- min(xscale, yscale)
> xadd <- 0.5*(pin[1]/scale - xdelta)
> yadd <- 0.5*(pin[2]/scale - ydelta)
> plot(numeric(0), numeric(0),
+       xlim = range(x)+c(-1,1)*xadd, ylim = range(y)+c(-1,1)*yadd,
+       type = "n", ann = FALSE)
> usr <- par("usr")
> rect(usr[1], usr[3], usr[2], usr[4], col="green3")
> contour(x, y, volcano, levels = lev, col="yellow", lty="solid", add=TRUE)
> box()
> title("A Topographic Map of Maunga Whau", font= 4)
> title(xlab = "Meters North", ylab = "Meters West", font= 3)
> mtext("10 Meter Contour Spacing", side=3, line=0.35, outer=FALSE,
+       at = mean(par("usr")[1:2]), cex=0.7, font=3)
```



```
> ## Conditioning plots
>
> par(bg="cornsilk")
> coplot(lat ~ long | depth, data = quakes, pch = 21, bg = "green3")
> par(opar)
```



Al realizar un gráfico en R, éste es enviado a un dispositivo gráfico, que no es más que una ventana gráfica o un archivo. En R studio está en la pestaña Plots del cuadrante inferior-derecho.

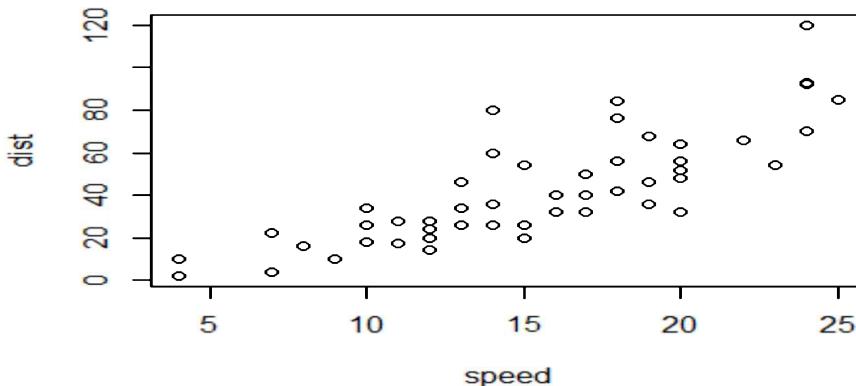
En R existen diferentes sistemas gráficos que podemos utilizar: base, lattice o ggplot2, que ofrecen distintas posibilidades y calidades.

Los gráficos se construyen tratando cada aspecto del gráfico separadamente a través de una serie de funciones, donde se han de establecer una serie de parámetros que debemos tener fijados de antemano (ajustar márgenes, definir tipo, líneas, etc.).

En este apartado vamos a ver un conjunto de gráficos elementales realizados con el paquete Base de R.

Un gráfico elemental X e Y se realiza con el siguiente script:

```
library(datasets)
data(cars)
with(cars, plot(speed, dist))
```



Funciones como "plot()" o "hist()" lanzan un device gráfico (si no está ya creado) y representan un nuevo gráfico en el dispositivo gráfico.

Estas funciones tienen muchos argumentos, como poner un título, etiquetas a los ejes, etc. Para ver los argumentos utilizamos la ayuda de la función, ?plot.

?plot

Por otro lado el sistema gráfico básico tiene muchos parámetros que podemos establecer y ajustar. Todos estos parámetros están documentados en ?par.

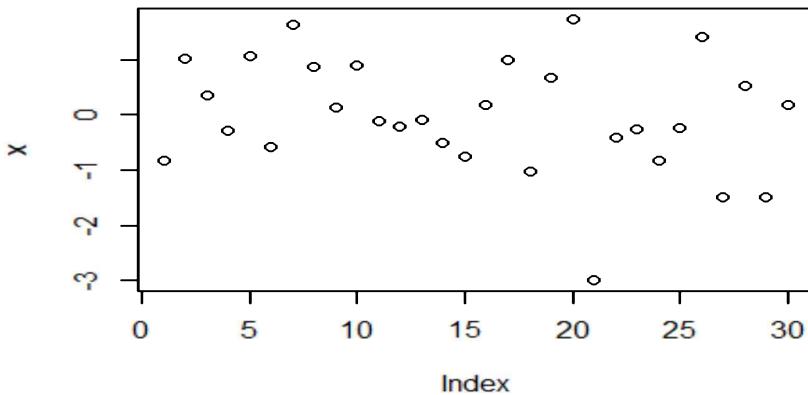
?par

Veamos a continuación algunas de las funciones gráficas más utilizadas. Todas ellas admiten los siguientes argumentos generales:

- add = FALSE (por defecto): si es TRUE superpone al gráfico existente.
- axes = TRUE (por defecto): si es FALSE no dibuja los ejes ni la caja del gráfico.
- type =: especifica el tipo de gráfico, p = puntos (por defecto), l = líneas, b = puntos conectados por lineas, h = líneas verticales.
- xlim =, ylim =: especifica los límites inferiores y superiores de los ejes.
- xlab =, ylab =: añade etiquetas a los ejes.
- main =: añade el título principal.
- sub =: añade un subtítulo (letra más pequeña).

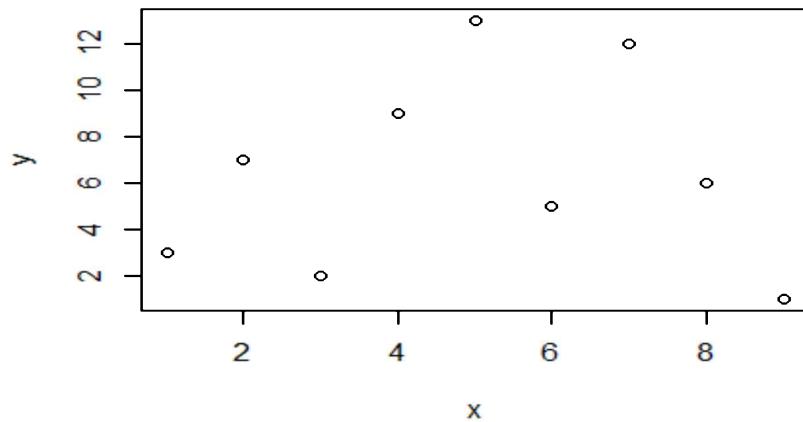
Un gráfico de dispersión se realiza con la función "plot()".

```
x <- rnorm(30)
plot(x)
```

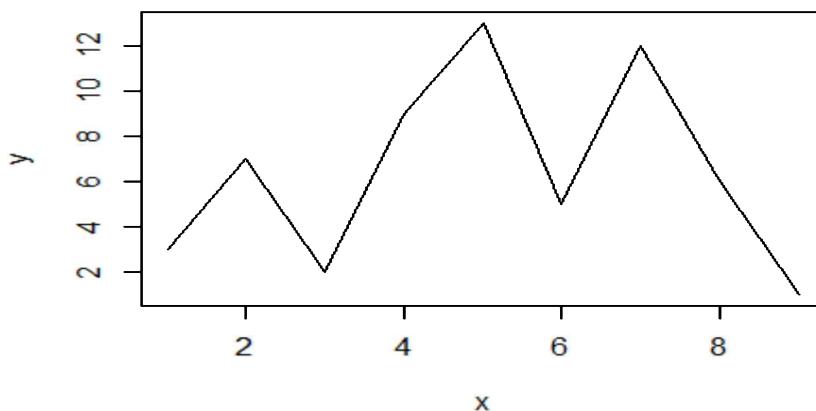


La función "plot()" también sirve para realizar gráficos bivariados, de puntos, líneas o puntos y líneas con títulos y nombres de ejes:

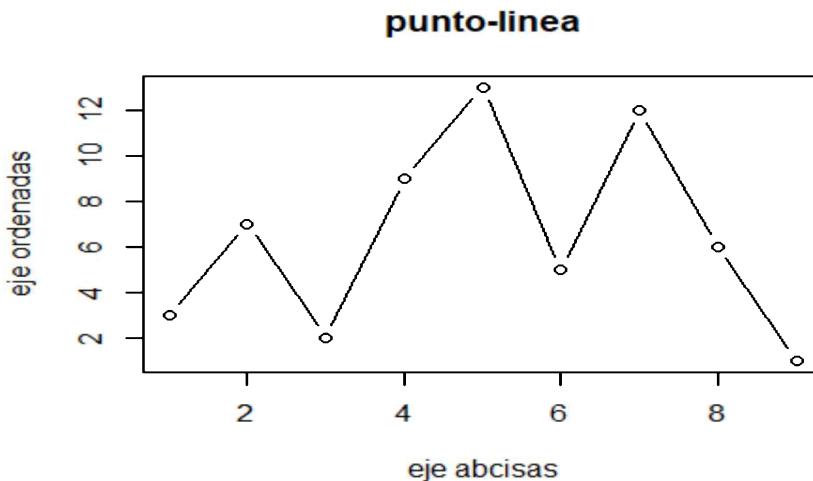
```
x <- 1:9
y <- c(3, 7, 2, 9, 13, 5, 12, 6, 1)
plot(x, y)
```



```
plot(x, y, type = "l")
```



```
plot(x, y, type = "b", main = "punto-linea", xlab = "eje abcisas", ylab = "eje ordenadas")
```



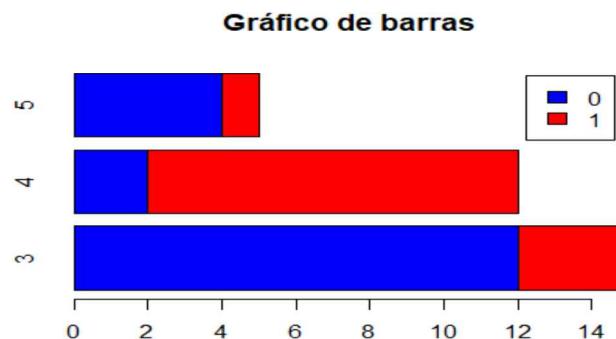
Un gráfico de barras se crea con la función "barplot(x)", donde x es un vector o una matriz.

```
x <- table(mtcars$gear)
barplot(x, main = "Gráfico de barras", names.arg = c("3 Gears", "4 Gears",
"5 Gears"))
```



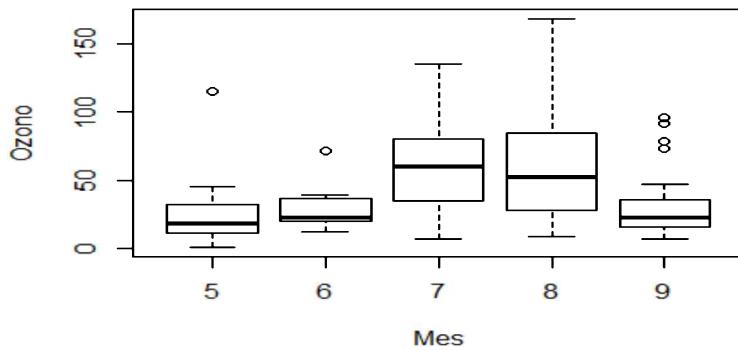
Un gráfico de barras horizontales, con cuadro de leyenda:

```
x <- table(mtcars$vs, mtcars$gear)
barplot(x, main = "Gráfico de barras", horiz = TRUE, col = c("blue", "red"), legend =
rownames(x))
```



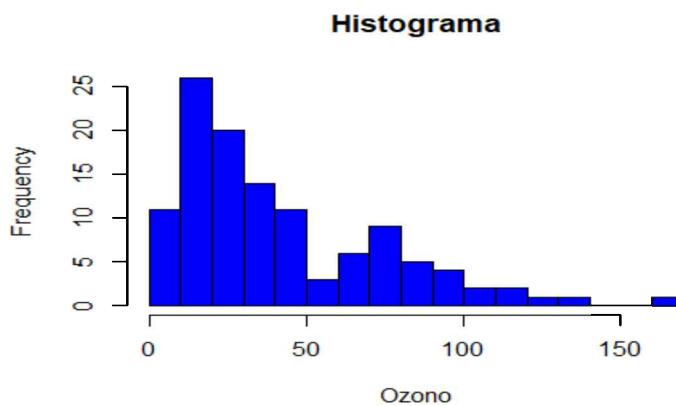
Los diagramas de caja-bigote se crean con la función "boxplot(x, data =)", donde x es una fórmula y data es el data.frame que contiene los datos.

```
airquality$Month <- factor(airquality$Month)
boxplot(Ozone ~ Month, data = airquality, xlab = "Mes", ylab = "Ozono")
```



Los histogramas se crean con la función "hist(x)", donde x es un vector numérico. Con la opción `frec = FALSE` se representan las densidades en lugar de las frecuencias. La opción `breaks = n` controla el número de barras.

```
hist(airquality$Ozone, breaks = 15, col = "blue", main = "Histograma", xlab = "Ozono")
```



La función "pie()" realiza gráficos de quesos.

```
precios <- c(0.23, 0.35, 0.14, 0.2, 0.23)
names(precios) <- c("manzana", "uva", "pera", "naranja", "cereza")
pie(precios, col = rainbow(length(names(precios))), main = "Precios frutas")
```



SEGUNDO BLOQUE. MÉTODOS ESTADÍSTICOS MULTIVARIANTES

TEMA 3: MODELO LINEAL GENERAL Y MODELO LINEAL GENERALIZADO

ÍNDICE

1. Modelo Lineal General

- 1.1. Modelo de Regresión Lineal
- 1.2. Extensiones al Modelo de Regresión Lineal
- 1.3. Modelos con variables cualitativas explicativas
- 1.4. Modelos con variable dependiente multivariante: MANOVA y MANCOVA
- 1.5. Estimación por máxima verosimilitud restringida (REML) en modelos mixtos
- 1.6. Ajuste de modelos mixtos con R

2. Modelo Lineal Generalizado

- 2.1. Formulación general
- 2.2. Modelos con variables cualitativas endógenas
- 2.3. Modelo Tobit

3. Evaluación de Modelos

- 3.1. Devianza. Estadístico G₂ de Wilks de razón de verosimilitudes
- 3.2. Estadístico χ^2 de Pearson
- 3.3. Criterio de Información de Akaike (AIC) y Criterio de Información Bayesiano (BIC)
- 3.4. Prueba de Hosmer-Lemeshaw
- 3.5. Medidas tipo R₂
- 3.6. Métodos específicos para modelos de clasificación

OBJETIVOS	PALABRAS CLAVE
<ul style="list-style-type: none"> - Conocer los principios básicos del diseño experimental. - Construir modelos de regresión y clasificación a partir del Modelo Lineal General y el Modelo Lineal Generalizado. - Contrastar e interpretar los parámetros del modelo. - Utilizar procedimientos forward, backward y stepwise de selección de variables. - Evaluar y comparar modelos de cara a la selección del más adecuado. 	<ul style="list-style-type: none"> · Contraste de hipótesis. · Diseño experimental. · Modelo Lineal General. · Modelo Lineal Generalizado. · Residuos de la Devianza. · Residuos de Pearson. · Coeficiente de determinación.

1. MODELO LINEAL GENERAL

1.1. MODELO DE REGRESIÓN LINEAL

1.1.1. Introducción

El concepto de regresión es uno de los pilares de la estadística, y data al menos desde principios de 1800 con los trabajos de Legendre, Gauss y Laplace. Es posible que el término *regresión* sea debido a Francis Galton, quien acuñó el término "regresión hacia la media" para describir la observación de que los hijos de padres muy altos tienden a ser algo más bajos que sus progenitores, y por el contrario los hijos de padres muy bajos suelen ser algo más altos, y por lo tanto acercarse en ambos casos más a la media de la población. Este fenómeno que se produce en muchas más circunstancias en la naturaleza queda muy bien explicado por Stephen M. Stiegler con el siguiente ejemplo: supongamos que efectuamos en dos momentos diferentes de tiempo un examen sobre una materia concreta a un alumno, y evaluamos primero uno de ellos, observando que obtiene una nota mucho más alta que la media de sus compañeros de clase. ¿Cómo de buena esperamos que sea la puntuación en el segundo examen? Probablemente alta, pero también probablemente no tan alta como en la primera ocasión, ya que probablemente el gran éxito en la primera ocasión se deba a dos componentes: por un lado, la capacidad del alumno (componente estable o permanente) y por otro un cierto grado de suerte (componente transitorio y en cierta medida aleatorio). El coeficiente que medía esa regresión hacia la media pasó desde entonces a indicarse con la letra r.

La *regresión lineal* es la técnica básica del análisis econométrico. Mediante dicha técnica tratamos de determinar relaciones de dependencia de tipo lineal entre una variable dependiente o endógena, respecto de una o varias variables explicativas o exógenas. Gujarati (1975) define el análisis de regresión como el estudio de la dependencia de la variable dependiente, sobre una o más variables explicativas, con el objeto de estimar o predecir el valor promedio poblacional de la primera en términos de los valores conocidos o fijos (en medias muestrales repetidas) de las últimas.

En este capítulo abordaremos en primer lugar el estudio del caso de una única ecuación de tipo lineal con una variable dependiente y una independiente, y la generalización del modelo al caso de múltiples variables exógenas. A continuación, se estudiarán las extensiones del modelo lineal general, el modelo lineal generalizado y, por último, otros modelos derivados, como la estimación curvilínea, la regresión Ridge, la regresión bayesiana u otros modelos que no dependen de hipótesis previas sobre las distribuciones, como la regresión robusta o la regresión no paramétrica.

1.1.2. Modelo de Regresión Lineal Simple

Partimos de la existencia de una relación lineal entre una variable endógena (Y_i) y una variable exógena (X_i):

$$Y_i = \beta_0 + \beta_1 X_i + e_i, i = 1, 2, \dots, n \quad [1]$$

Nuestro objetivo consiste en estimar los dos parámetros de la ecuación anterior a partir de los datos muestrales de los que disponemos. Para ello utilizaremos el método de los *Mínimos Cuadrados Ordinarios (MCO)*, pero antes de ver en qué consiste este método debemos plantear ciertas hipótesis sobre el comportamiento de las variables que integran el modelo.

La variable e_i la denominamos término de perturbación o error, y en ella recogemos todos aquellos factores que pueden influir a la hora de explicar el comportamiento de la variable Y_i y que, sin embargo, no están reflejados en las variables explicativas, X_i . Estos factores deberían ser poco importantes, ya que no debería existir ninguna variable explicativa relevante omitida en el modelo de regresión. En caso contrario estaríamos incurriendo en lo que se conoce como un error de especificación del modelo. El término de perturbación también recogería los posibles errores de medida de la variable dependiente, Y_i .

De lo anterior se desprende que, a la hora de estimar los parámetros del modelo, resultará de vital importancia que dicho término de error no ejerza ninguna influencia determinante en la explicación del comportamiento de la variable dependiente. Por ello, si el modelo está bien especificado, cuando se aplica el método de Mínimos Cuadrados Ordinarios, cabe realizar las siguientes hipótesis de comportamiento sobre el término de error:

1. La esperanza matemática de e_i es cero, tal que $E(e_i) = 0$. Es decir, el comportamiento del término de error no presenta un sesgo sistemático en ninguna dirección determinada. Por ejemplo, si estamos realizando un experimento en el cual tenemos que medir la longitud de un determinado objeto, a veces al medir dicha longitud cometemos un error de medida por exceso y otras por defecto, pero en media los errores estarán compensados.
2. La covarianza entre e_i y e_j es nula, y por tanto, $E(e_i \cdot e_j) = 0$. Esto quiere decir que el error cometido en un momento determinado, i , no debe estar correlacionado con el error cometido en otro momento del tiempo, j , o dicho de otro modo, los errores no ejercen influencia unos sobre otros. En caso de existir este tipo de influencia o correlación, nos encontraríamos ante el problema de la autocorrelación en los residuos, el cual impide realizar una estimación por Mínimos Cuadrados válida.
3. La matriz de varianzas y covarianzas del término de error debe ser escalar, tal que $Var(e_i) = \sigma^2 I$, $i = 1, \dots, n$, donde I es la matriz identidad. Dado que siempre que medimos una variable, se produce un cierto error, resulta deseable que los errores que cometamos en momentos diferentes del tiempo sean similares en cuantía. Esta condición es lo que se conoce como supuesto de homocedasticidad que, en caso de no verificarse, impediría un uso correcto de la estimación lineal por Mínimos Cuadrados.

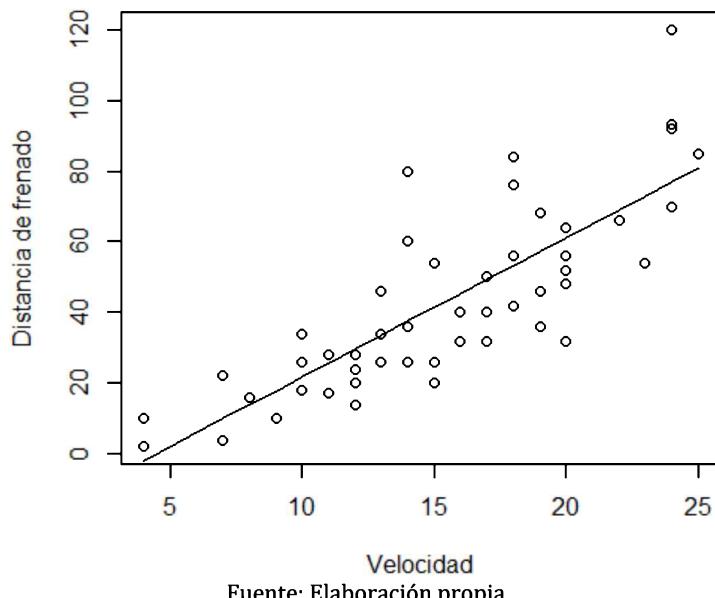
Estas hipótesis implican que los errores siguen una distribución Normal de media cero y varianza constante por lo que, dado su carácter aleatorio, hace que los errores sean por naturaleza impredecibles.

Asimismo, las variables incluidas en el modelo deben verificar que:

1. La variable dependiente Y_i se ajusta al modelo lineal durante todo el periodo muestral, es decir, no se produce un cambio importante en la estructura de comportamiento de Y_i a lo largo de la muestra considerada. Por tanto, se distribuirá como una normal con $E(Y_i) = \beta_0 + \beta_1 X_i$ y $Var(Y_i) = Var(e_i)$.
2. La variable explicativa, X_i , es no estocástica, es decir, es considerada fija en muestreos repetidos.

Si suponemos que se verifican los supuestos anteriores, la estimación mínimo cuadrática de los parámetros β_0 y β_1 , dará como resultado gráfico una recta que se ajusta lo máximo posible a la nube de puntos definida por todos los pares de valores muestrales (X_i, Y_i) , tal y como se puede apreciar en el Figura 1.

FIGURA 1. LÍNEA DE REGRESIÓN AJUSTADA. DISTANCIA DE FRENADO ~ VELOCIDAD



Fuente: Elaboración propia

El término de error, e_i , puede ser entendido, a la vista del gráfico anterior, como la distancia que existe entre el valor observado, Y_i , y el correspondiente valor estimado, que sería la imagen de X_i en el eje de ordenadas. El objetivo de la estimación por Mínimos Cuadrados Ordinarios es, precisamente, minimizar el sumatorio de todas esas distancias al cuadrado; es decir:

$$\text{Min} \sum(e_i)^2 = \sum(Y_i - \hat{Y}_i)^2 = \sum(Y_i - \hat{\beta}_0 - \hat{\beta}_1 X_i)^2 \quad [2]$$

Derivando esta expresión respecto a los coeficientes $\hat{\beta}_0$ y $\hat{\beta}_1$ e igualando a cero obtenemos el sistema de ecuaciones normales:

$$\begin{aligned} \sum Y_i &= n \hat{\beta}_0 + \sum \hat{\beta}_1 X_i \\ \sum (Y_i \cdot X_i) &= \sum \hat{\beta}_0 X_i + \sum \hat{\beta}_1 (X_i)^2 \end{aligned} \quad [3]$$

donde n representa el tamaño muestral.

Resolviendo dicho sistema de ecuaciones obtenemos la solución para los parámetros $\hat{\beta}_0$ y $\hat{\beta}_1$:

$$\begin{aligned} \hat{\beta}_0 &= \bar{Y} - \hat{\beta}_1 \bar{X} \\ \hat{\beta}_1 &= \frac{\sum((Y_i - \bar{Y}) \cdot (X_i - \bar{X}))}{\sum(X_i - \bar{X})^2} \end{aligned} \quad [4]$$

1.1.3. Modelo de Regresión Lineal Múltiple

Pasamos a continuación a generalizar el modelo anterior al caso de un modelo con varias variables exógenas, de tal forma que se trata de determinar la relación que existe entre la variable endógena Y y las variables exógenas: X_1, X_2, \dots, X_k . Dicho modelo se puede formular matricialmente de la siguiente manera:

$$y = \beta X + e \Rightarrow Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \dots + \beta_k X_{ki} + e_i, i = 1, 2, \dots, n \quad [5]$$

donde:

$$y = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{bmatrix} \quad [6]$$

es el vector de observaciones de la variable endógena

$$X = \begin{bmatrix} 1 & X_{11} & X_{21} & \dots & X_{k1} \\ 1 & X_{12} & X_{22} & \dots & X_{k2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & X_{1n} & X_{2n} & \dots & X_{kn} \end{bmatrix} \quad [7]$$

es el vector de observaciones de las variables exógenas

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix} \quad [8]$$

es el vector de los coeficientes que pretendemos estimar

$$e = \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{bmatrix} \quad [9]$$

es el vector de términos de error.

La primera columna de la matriz X , denominada *matriz de diseño* corresponde a un vector de n unos, siendo éste la variable que se considera asociada a la estimación del parámetro β_0 , y el resto, de la segunda en adelante, corresponde a cada una de las variables explicativas consideradas.

Suponiendo que se verifican las hipótesis que veíamos antes, el problema a resolver nuevamente es la minimización de la suma de los cuadrados de los términos de error tal que:

$$\text{Min} \sum (e_i)^2 = \sum (Y_i - \hat{Y}_i)^2 = \sum (Y_i - \hat{\beta} X)^2 \quad [10]$$

Desarrollando dicho cuadrado y derivando respecto a cada $\hat{\beta}_i$ obtenemos el siguiente sistema de ecuaciones normales expresado en notación matricial:

$$X' X \hat{\beta} = X' y \quad [11]$$

en donde basta con despejar $\hat{\beta}$ premultiplicando ambos miembros por la inversa de la matriz $(X' X)$ para obtener la estimación de los parámetros del modelo tal que:

$$\hat{\beta} = (X' X)^{-1} X' y \quad [12]$$

donde:

$$X'X = \begin{bmatrix} n & \sum X_{1i} & \sum X_{2i} & \dots & \sum X_{ki} \\ \sum X_{1i} & \sum X_{1i}^2 & \sum(X_{1i}X_{2i}) & \dots & \sum(X_{1i}X_{ki}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \sum X_{ki} & \sum(X_{ki}X_{1i}) & \sum(X_{ki}X_{2i}) & \dots & \sum X_{ki}^2 \end{bmatrix} \quad [13]$$

$$X'y = \begin{bmatrix} \sum Y_i \\ \sum(Y_i X_{1i}) \\ \vdots \\ \sum(Y_i X_{ki}) \end{bmatrix} \quad [14]$$

$$\hat{\beta} = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \vdots \\ \hat{\beta}_k \end{bmatrix} \quad [15]$$

Cada uno de los coeficientes estimados, $\hat{\beta}_i$, son una estimación insesgada del verdadero parámetro del modelo y representan la variación que experimenta la variable dependiente Y_i cuando una variable independiente X_i varía en una unidad y todas las demás permanecen constantes (supuesto ceteris paribus). Dichos coeficientes poseen propiedades estadísticas muy interesantes ya que, si se verifican los supuestos antes comentados, son insesgados, eficientes y óptimos.

Los valores estimados de la variable dependiente en expresión matricial vienen dados por:

$$\hat{y} = X\hat{\beta} = X(X'X)^{-1}X'y \quad [16]$$

Y la distribución estadística de la variable dependiente, se expresa en forma matricial:

$$E(y) = X\beta \quad [17]$$

$$\text{Var}(y) = \sigma^2 I \quad [18]$$

1.1.4. Propiedades estadísticas del estimador MCO

Si se cumplen las hipótesis de comportamiento sobre el término error, la distribución de probabilidad del estimador MCO $\hat{\beta}$ será una distribución Normal multivariante con vector de medias β y matriz de varianzas y covarianzas $\sigma^2(X'X)^{-1}$.

La esperanza matemática del estimador MCO se demuestra a partir de:

$$\begin{aligned} E(\hat{\beta}) &= E((X'X)^{-1}X'y) \\ &= (X'X)^{-1}X'E(y) \\ &= (X'X)^{-1}X'\beta = \beta \end{aligned} \quad [19]$$

De la definición de matriz de varianzas y covarianzas, se tiene que:

$$\begin{aligned} \text{Var}(\hat{\beta}) &= \text{Var}((X'X)^{-1}X'y) \\ &= (X'X)^{-1}X'\text{Var}(y)(X'X)^{-1}X' \\ &= (X'X)^{-1}X'\text{Var}(y)X(X'X)^{-1} \\ &= (X'X)^{-1}X'\sigma^2 IX(X'X)^{-1} \\ &= (X'X)^{-1}\sigma^2 \end{aligned} \quad [20]$$

El estimador $\hat{\beta}$ del parámetro β es insesgado porque su esperanza matemática coincide con el verdadero valor del parámetro $E(\hat{\beta}) = \beta$. Se dice que un estimador insesgado es más eficiente que otro estimador insesgado, si la varianza muestral de dicho estimador es menor que la varianza muestral de ese otro estimador. El *teorema de Gauss-Markov* demuestra que el estimador MCO es el más eficiente de la clase de estimadores lineales e insesgados de β .

1.1.5. Coeficiente de determinación

Una vez estimada la ecuación de regresión lineal tiene interés determinar la exactitud del ajuste realizado. Para ello hay que analizar la variación que experimenta esta variable dependiente y , dentro de esta variación, se estudia qué parte está siendo explicada por el modelo de regresión y qué parte es debida a los errores o residuos.

La forma de realizar dicho análisis es a partir de la siguiente expresión:

$$SCT = SCE + SCR \quad [21]$$

donde:

- SCT : es la Suma de Cuadrados Totales y representa una medida de la variación de la variable dependiente.
- SCE es la Suma de Cuadrados Explicados por el modelo de regresión.
- SCR es la Suma de Cuadrados de los Errores

Cuando el modelo tiene término independiente, cada una de estas sumas viene dada por:

$$SCT = y'y - n\bar{Y}^2 = \sum Y_i^2 - n\bar{Y}^2 \quad [22]$$

$$SCE = (\hat{\beta}'X')(X\hat{\beta}) - n\bar{Y}^2 = \sum \hat{Y}_i^2 - n\bar{Y}^2 \quad [23]$$

$$SCR = y'y - (\hat{\beta}'X')(X\hat{\beta}) = \sum Y_i^2 - \sum \hat{Y}_i^2 = \sum (Y_i - \hat{Y}_i)^2 = \sum \hat{e}_i^2 \quad [24]$$

A partir de las expresiones anteriores es posible obtener una medida estadística acerca de la bondad de ajuste del modelo mediante lo que se conoce como coeficiente de determinación (R^2), que se define como:

$$R^2 = \frac{SCE}{SCT} = 1 - \frac{SCR}{SCT}, \quad 0 \leq R^2 \leq 1 \quad [25]$$

Mediante este coeficiente es posible seleccionar el mejor modelo de entre varios que tengan el mismo número de variables exógenas, ya que la capacidad explicativa de un modelo es mayor cuanto más elevado sea el valor que tome este coeficiente. Sin embargo, hay que tener cierto cuidado a la hora de trabajar con modelos que presenten un R^2 muy cercano a 1 pues, aunque podría parecer que estamos ante el modelo “perfecto”, en realidad podría encubrir ciertos problemas de índole estadística como la multicolinealidad que veremos en el siguiente apartado.

Por otra parte, el valor del coeficiente de determinación aumenta con el número de variables exógenas del modelo por lo que, si los modelos que se comparan tienen distinto número de variables exógenas, no puede establecerse comparación entre sus R^2 . En este caso debe emplearse el coeficiente de determinación corregido (\bar{R}^2), el cual depura el incremento que experimenta el coeficiente de determinación cuando el número de variables exógenas es mayor.

La expresión analítica de la versión corregida es:

$$\bar{R}^2 = 1 - \frac{\frac{SCR}{SCT}}{\frac{n-k-1}{n-1}} \quad [26]$$

1.1.6. Inferencia acerca de los estimadores

Hasta el momento hemos visto como la estimación por MCO permite obtener estimaciones puntuales de los parámetros del modelo. La inferencia acerca de los mismos permite completar dicha estimación puntual, mediante la estimación por intervalos y los contrastes de hipótesis. Los primeros posibilitan la obtención de un intervalo dentro del cual, con un determinado nivel de confianza, oscilará el verdadero valor de un parámetro, mientras que los segundos nos permitirán extraer consecuencias del modelo, averiguando si existe o no, evidencia acerca de una serie de conjeturas que pueden plantearse sobre sus parámetros.

La inferencia estadística consiste en la estimación de los parámetros poblacionales a partir de la información extraída de una muestra de dicha población. El número de estimaciones que podemos realizar de una población, a través de la extracción de diferentes muestras de un mismo tamaño, es generalmente muy grande porque cada una de las muestras posibles que se pueden sacar de la población arrojaría una estimación.

Por esta razón, a la estimación que obtenemos en una investigación por muestreo la acompañamos con un intervalo de valores posibles. La amplitud de dicho intervalo dependerá del grado de confianza que establezcamos.

El grado o nivel de confianza nos expresa el número de veces que la media verdadera de la población está incluida en cien intervalos de cien muestras extraídas de una población dada. El nivel de confianza más utilizado es el 95%, lo que quiere decir que 95 de cada 100 intervalos construidos contendrán el verdadero valor de la media.

El intervalo de confianza para la media de una población normalmente distribuida se construye en base a la probabilidad de que dicha media esté comprendida entre dos valores \bar{X}_a y \bar{X}_b equidistantes a ella:

$$P(\bar{X}_a \leq \mu_{\bar{X}} \leq \bar{X}_b) = 1 - \alpha \quad [27]$$

siendo $1 - \alpha$ el nivel o grado de confianza asociado a dicho intervalo.

En términos generales, los intervalos de confianza para los estadísticos muestrales se expresan como:

$$\text{Estimador} \pm (\text{Factor de Fiabilidad}) * (\text{Error Típico del Estimador}) \quad [28]$$

1.1.6.1. Intervalos de Confianza

Para construir los intervalos de confianza de los parámetros β_i , se parte de que la estimación MCO proporciona el valor medio de los posibles valores que pudiera tener dicho parámetro, y de que la distribución de dichos valores sigue una distribución derivada de la Normal que se conoce como t de Student.

Dicha distribución es simétrica presentando mayor dispersión que la curva Normal estándar para un tamaño muestral n pequeño. A medida que n aumenta ($n > 100$) es prácticamente igual que la distribución Normal.

El cálculo del intervalo de confianza para $\hat{\beta}_i$ se realiza mediante la siguiente expresión:

$$\hat{\beta}_i \pm S_{\hat{\beta}_i} t_{n-k-1, 1-\frac{\alpha}{2}} \quad [29]$$

donde $S_{\hat{\beta}_i}$ es la desviación típica estimada para el coeficiente, que se obtiene de la matriz de varianzas y covarianzas de los estimadores MCO $\hat{V}ar(\hat{\beta}) = (X'X)^{-1} \hat{\sigma}^2$, calculado $\hat{\sigma}^2$ a partir de:

$$\hat{\sigma}^2 = \frac{\sum \hat{e}_i^2}{n-k} = \frac{e'e}{n-k-1} \quad [30]$$

1.1.6.2. Contraste de hipótesis

Una buena parte de las investigaciones estadísticas están orientadas al desarrollo de procesos encaminados a la contrastación de hipótesis que previamente se han establecido.

Una hipótesis es una afirmación que está sujeta a verificación o comprobación. Hay que tener presente que una hipótesis no es un hecho establecido o firme, las hipótesis están basadas en la experiencia, en la observación, en la experimentación o en la intuición del sujeto que las formula.

Cuando las hipótesis se plantean de tal modo que se pueden comprobar por medio de métodos estadísticos reciben el nombre de hipótesis estadísticas. Estas hipótesis son afirmaciones que se efectúan sobre uno o más parámetros de una o más poblaciones. Las hipótesis estadísticas son de dos tipos: hipótesis nula e hipótesis alternativa. La hipótesis nula, o que no se verifique dicha afirmación, simbolizada por H_0 , es la hipótesis que se debe comprobar.

Para contrastar una hipótesis nula examinamos los datos de la muestra tomados de la población y determinamos si son o no compatibles con dicha hipótesis. Si son compatibles entonces H_0 se acepta, en caso contrario se rechaza. Si se acepta la hipótesis nula afirmamos que los datos de esa muestra en concreto no dan suficiente evidencia para que concluyamos que la hipótesis nula sea falsa; si se rechaza decimos que los datos particulares de la muestra ponen de manifiesto que la hipótesis nula es falsa, entonces la hipótesis alternativa, H_1 , es considerada verdadera.

El criterio que permite decidir si rechazamos o no la hipótesis nula es siempre el mismo. Definimos un estadístico de prueba, y unos límites que dividen el espacio muestral en una región en donde se rechaza la hipótesis establecida, y otra región en la que no se rechaza, llamada región de aceptación. A la región donde se rechaza la hipótesis nula se le llama región crítica. Esta región es un subconjunto del espacio muestral, y si el valor del estadístico de prueba pertenece a él se rechaza la hipótesis nula.

El límite entre la región crítica y la región de aceptación viene determinado por la información previa relativa a la distribución del estadístico de prueba.

Señalar que un estadístico de prueba es una fórmula que nos dice como confrontar la hipótesis nula con la información de la muestra y es, por tanto, una variable aleatoria cuyo valor cambia de muestra a muestra.

Otra de las consideraciones a realizar en el contraste de hipótesis es fijar la probabilidad del error de rechazar la prueba siendo cierta. A este error se le denomina *nivel de significación*. Por ejemplo, si se utiliza un nivel de significación de $\alpha = 0,05$, equivale a decir que si para

realizar un contraste tomáramos infinitas muestras de la población, rechazaríamos la hipótesis nula de forma incorrecta un 5% de las veces.

En la formalización del procedimiento de contrastación podemos distinguir cinco pasos principales:

1. Formular las hipótesis H_0 y H_1 a contrastar.
2. Proponer un estadístico de contraste.
3. Elegir un nivel de significación aceptable.
4. Determinar la región crítica y/o calcular el p-valor.
5. Tomar la decisión de aceptar o rechazar la hipótesis nula.

Para contrastar la hipótesis de que el parámetro poblacional β_i tome el valor β_i^0 , necesitamos:

1. Formular las hipótesis:

$$H_0: \beta_i = \beta_i^0 \quad [31]$$

$$H_1: \beta_i \neq \beta_i^0 \quad [32]$$

2. Calcular el estadístico de contraste:

$$t_i = \frac{\hat{\beta}_i - \beta_i}{s_{\hat{\beta}_i}} \quad [33]$$

3. Elegir el nivel de significación: $\alpha = 0.05$ o $\alpha = 0.1$
4. Obtener en las tablas de la distribución t-student el valor crítico para un contraste bilateral (2 colas): $t_{n-k-1,0.975}$ o $t_{n-k-1,0.95}$

También se puede calcular el p-valor asociado al valor de la t de student correspondiente al estadístico de contraste. El p-valor se define como la probabilidad de obtener un estadístico de contraste al menos tan extremo como el que evalúo.

5. Establecer la regla de decisión para aceptar la hipótesis nula H_0 : si $|t_i| < t_{n-k-1,1-\frac{\alpha}{2}}$ acepto H_0 (dado que por la simetría de la función, $-t_{n-k-1,\frac{\alpha}{2}} = t_{n-k-1,1-\frac{\alpha}{2}}$).

En el caso de que obtenga el p-valor del estadístico de contraste, este debe ser inferior al nivel de significación α elegido.

1.1.6.3. Contraste de significación individual

Se denomina contraste de significación individual al contraste de hipótesis de que el parámetro $\beta_i = 0$. Es decir:

1. Formular las hipótesis:

$$H_0: \beta_i = 0 \quad [34]$$

$$H_1: \beta_i \neq 0 \quad [35]$$

2. Calcular el estadístico de contraste:

$$t_i = \frac{\hat{\beta}_i}{s_{\hat{\beta}_i}} \quad [36]$$

3. Elegir el nivel de significación: $\alpha = 0.05$ o $\alpha = 0.1$
4. Obtener en las tablas de la distribución t-student el valor crítico o p-valor para un contraste bilateral (2 colas): $t_{n-k-1,0.975}$ o $t_{n-k-1,0.95}$
5. Establecer la regla de decisión para aceptar la hipótesis nula H_0 : si $|t_i| < t_{n-k-1,1-\frac{\alpha}{2}}$ acepto H_0 .

Aceptar H_0 en este caso equivale a decir que el parámetro β_i es no significativo, ya que el valor cero entra dentro de los posibles valores que puede tomar el coeficiente en la población. En cambio rechazar H_0 , que ocurre cuando $|t_i| > t_{n-k-1,1-\frac{\alpha}{2}}$ equivale a decir que el coeficiente β_i es significativo, y por tanto admite algún tipo de influencia entre la variable X_i y la dependiente Y_i .

1.1.6.4. Contraste de significación global (Tabla ANOVA)

La hipótesis de no significación global $H_0: \beta_1 = \beta_2 = \dots = \beta_k = 0$ se rechaza al nivel de significación α construyendo el estadístico experimental:

$$F_{exp} = \frac{\frac{SCE}{k}}{\frac{SCR}{n-k-1}} \quad [37]$$

y la regla de decisión que rechaza la hipótesis H_0 ocurre cuando $F_{exp} > F_{k,n-k-1,\alpha}$.

El contraste de significación global se resume en el cuadro siguiente, en donde la variación total de la variable dependiente (SCT) se descompone en la explicada por la regresión (SCE) y en la no explicada (SCR). Los grados de libertad de estas tres sumas de cuadrados son $n - 1$, k y $n - k - 1$, respectivamente.

A partir de esta información muestral, podemos calcular el numerador y denominador del estadístico F_{exp} .

TABLA 1. TABLA ANOVA. MODELO DE REGRESIÓN LINEAL

Fuente de variación	Suma de cuadrados	Grados de libertad	Cuadrado medio	Estadístico F
Regresión	SCE	k	$\frac{SCE}{k}$	$\frac{SCE}{k}$
Residual	SCR	$n - k - 1$	$\frac{SCR}{n - k - 1}$	$\frac{SCR}{n - k - 1}$
Total	SCT	$n - 1$		

Fuente: Elaboración propia

1.1.7. Predicción

Una vez estimado y validado el modelo, una de sus aplicaciones más importantes consiste en poder realizar predicciones acerca del valor que tomaría la variable endógena en el futuro o para una unidad extramuestral. Esta predicción se puede realizar tanto para un valor individual como para un valor medio, o esperado, de la variable endógena, siendo posible efectuar una predicción puntual o por intervalos. Su cálculo se realiza mediante las expresiones que figuran a continuación:

- a) Predicción individual: se trata de hallar el valor estimado para la variable Y un periodo hacia delante. En este caso basta con sustituir el valor de las variables exógenas en el modelo en el siguiente periodo y calcular el nuevo valor de Y :

$$\hat{y}_{t+1} = \hat{\beta}_0 + \hat{\beta}_1 x_{1,t+1} + \dots + \hat{\beta}_k x_{k,t+1} = X_{t+1} \hat{\beta} \quad [38]$$

- b) Intervalo de predicción. Para hallar un intervalo de predicción debe utilizarse la siguiente expresión:

$$\hat{y}_{t+1} \pm t_{n-k-1,1-\frac{\alpha}{2}} \hat{\sigma} \sqrt{1 + X'_{t+1} (X'X)^{-1} X_{t+1}} \quad [39]$$

- c) Intervalos de predicción para un valor medio o esperado, y_i :

$$\hat{y}_i \pm t_{n-k-1,1-\frac{\alpha}{2}} \hat{\sigma} \sqrt{X'_{i\cdot} (X'X)^{-1} X_i} \quad [40]$$

1.1.8. Estimación del modelo de regresión con R

1.1.8.1. Modelo de Regresión Lineal Simple

Creamos en primer lugar un vector de reales mediante la función c y lo guardamos con el nombre “Cantidad”.

```
Cantidad <- c(2.456, 2.325, 2.250, 2.200, 2.100, 2.082, 2.045, 2.024)
```

Se crea ahora el vector de nombre “Precio”.

```
Precio<-c(82,92,94,99,106,108,112,115)
```

Para obtener los estadísticos básicos del vector (Cantidad): media, desviación estándar, varianza y mediana, se utilizan las siguientes funciones R:

```
mean(Cantidad)
[1] 2.18525
sd(Cantidad)
[1] 0.1515847
var(Cantidad)
[1] 0.02297793
median(Cantidad)
[1] 2.15
```

Si se quiere tener un resumen sumario de estadísticas de una variable:

```
summary(Cantidad)
   Min. 1st Qu. Median Mean 3rd Qu. Max.
2.024  2.073  2.150  2.185  2.269  2.456
```

La función de R que nos permite estimar un modelo de regresión lineal es la función lm. La forma de invocar a la función para estimar un modelo de regresión lineal simple es lm(y~x). Se puede consultar la ayuda de la función para ver todas las posibilidades que ofrece.

En nuestro ejemplo, obtenemos:

```
lm(Cantidad~Precio)

Call:
lm(formula = Cantidad ~ Precio)

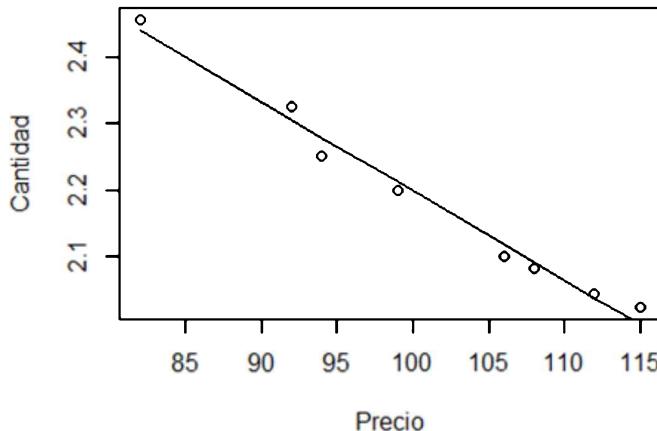
Coefficients:
(Intercept)      Precio
3.53427       -0.01336
```

En lugar de invocar simplemente la función podemos guardar su resultado en una variable y veremos así que obtenemos más información.

```
reg = lm(Cantidad~Precio)
```

Si queremos obtener una gráfica con los resultados de la regresión realizada:

```
plot(Cantidad~Precio)
lines(reg$fitted~Precio)
```



Para realizar el análisis del modelo estimado utilizaremos la función summary. Así:

```
summary(reg)

Call:
lm(formula = Cantidad ~ Precio)

Residuals:
    Min      1Q  Median      3Q     Max 
-0.02875 -0.01359 -0.00154  0.01762  0.02574 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 3.5342726  0.0734707 48.10 5.41e-09 ***
Precio      -0.0133567  0.0007235 -18.46 1.63e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.02154 on 6 degrees of freedom
Multiple R-squared:  0.9827, Adjusted R-squared:  0.9798 
F-statistic: 340.8 on 1 and 6 DF,  p-value: 1.629e-06
```

Como puede observarse, el test de significación global, con un valor de $F = 340.8$ y $p - value = 0.000001629$ nos indica que el coeficiente β_1 (precio) tiene un valor significativamente distinto de 0. Al tratarse de una regresión lineal simple, este contraste es equivalente al derivado de la t de Student con $H_0: \beta_1 = 0$, donde como vemos obtenemos el mismo p-value, siendo el valor del estadístico t la raíz cuadrada del estadístico F ($t^2 = -18.46^2 = 340.8$).

Por otra parte, el parámetro β_0 es también significativamente distinto de 0, con un valor del estadístico $t = 48.10$ y un $p - value = 5.41e - 09$.

Respecto a la bondad del ajuste, se obtiene un $R^2 = 0.9827$, indicando un alto grado de ajuste.

Finalmente, el modelo estimado ha sido:

$$\text{Cantidad} = 3.5342726 - 0.0133567 * \text{Precio}$$

1.1.8.2. Modelo de Regresión Lineal Múltiple

Utilizando la base de datos "mtcars", con datos sobre automóviles y sus características que incluye el programa R, cuya estructura se lista con la función "str":

```
data(mtcars)
str(mtcars)
'data.frame': 32 obs. of 11 variables:
 $ mpg : num 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
 $ cyl : num 6 6 4 6 8 6 8 4 4 6 ...
 $ disp: num 160 160 108 258 360 ...
 $ hp : num 110 110 93 110 175 105 245 62 95 123 ...
 $ drat: num 3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
 $ wt : num 2.62 2.88 2.32 3.21 3.44 ...
 $ qsec: num 16.5 17 18.6 19.4 17 ...
 $ vs : num 0 0 1 1 0 1 0 1 1 1 ...
 $ am : num 1 1 1 0 0 0 0 0 0 0 ...
 $ gear: num 4 4 4 3 3 3 3 4 4 4 ...
 $ carb: num 4 4 1 1 2 1 4 2 2 4 ...
```

Realizamos una regresión lineal multiple entre el consumo de gasolina y todas las características que la base de datos incorpora para cada coche.

```
summary(lm(mpg~.,data=mtcars))

Call:
lm(formula = mpg ~ ., data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.4506 -1.6044 -0.1196  1.2193  4.6271 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 12.30337  18.71788  0.657   0.5181    
cyl        -0.11144  1.04502  -0.107   0.9161    
disp        0.01334  0.01786  0.747   0.4635    
hp         -0.02148  0.02177  -0.987   0.3350    
drat        0.78711  1.63537  0.481   0.6353    
wt        -3.71530  1.89441  -1.961   0.0633 .  
qsec        0.82104  0.73084  1.123   0.2739    
vs          0.31776  2.10451  0.151   0.8814    
am          2.52023  2.05665  1.225   0.2340    
gear        0.65541  1.49326  0.439   0.6652    
carb       -0.19942  0.82875  -0.241   0.8122    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.65 on 21 degrees of freedom
Multiple R-squared:  0.869, Adjusted R-squared:  0.8066 
F-statistic: 13.93 on 10 and 21 DF,  p-value: 3.793e-07
```

El sumario de estadísticas sobre la regresión nos ofrece los resultados de los contrastes de significación individual sobre los parámetros y el de significación global. El hecho de que el contraste global sea significativo ($F = 13.93; p = 3.793e - 07$) y ninguno de los individuales lo sea, es debido a la existencia de multicolinealidad en las variables independientes. Este problema tiene distintas soluciones, como pueden ser la selección de variables o la aplicación de técnicas de reducción de dimensiones en las variables explicativas, cuestiones que veremos más adelante.

Para realizar una predicción del consumo de gasolina (mpg) en el modelo lineal que utiliza como variables explicativas el peso del vehículo (wt) y los caballos de vapor (hp), en un coche que pese 3 (1.000 lbs) y tenga 120 hp hay que utilizar la función R "predict".

```
summary(lm(mpg~wt+hp,data=mtcars))

Call:
lm(formula = mpg ~ wt + hp, data = mtcars)

Residuals:
    Min      1Q Median      3Q     Max 
-3.941 -1.600 -0.182  1.050  5.854 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 37.22727   1.59879  23.285 < 2e-16 ***
wt          -3.87783   0.63273  -6.129 1.12e-06 ***
hp          -0.03177   0.00903  -3.519  0.00145 **  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.593 on 29 degrees of freedom
Multiple R-squared:  0.8268, Adjusted R-squared:  0.8148 
F-statistic: 69.21 on 2 and 29 DF,  p-value: 9.109e-12

newdatos=data.frame(wt=3,hp=120)
predict(lm(mpg~wt+hp,data=mtcars),newdatos,interval="prediction",type="response")
       fit      lwr      upr
1 21.78102 16.38174 27.18031
```

1.2. EXTENSIONES AL MODELO DE REGRESIÓN LINEAL

1.2.1. Introducción

Como veíamos en el apartado anterior, el modelo de regresión lineal requiere que se cumplan las siguientes hipótesis sobre los términos de error:

- Media cero: $E(e_i) = 0 \quad i = 1, \dots, n$
- Varianza constante: $Var(e_i) = \sigma^2 I \quad i = 1, \dots, n$
- Residuos incorrelacionados: $Cov(e_i, e_j) = 0$

El incumplimiento de alguna de dichas hipótesis implica la no aleatoriedad de los residuos y, por tanto, la existencia de alguna estructura o relación de dependencia en los residuos que puede ser estimada, debiendo ser considerada en la especificación inicial del modelo. Los principales problemas asociados al incumplimiento de las hipótesis de normalidad de los residuos son, por un lado, la heterocedasticidad, cuando la varianza de los mismos no es constante, y la autocorrelación o existencia de relación de dependencia o correlación entre los diferentes residuos, lo que violaría el supuesto de términos de error incorrelacionados.

Si se construye una gráfica de los resultados de una estimación mínimo cuadrática (en ordenadas) frente al valor absoluto de los residuos (en abscisas), cuando éstos últimos presentan una distribución aleatoria, es decir, una distribución Normal de media cero y varianza constante, $N(0, \sigma^2)$, el resultado obtenido (Figura 2) muestra que el tamaño del error es independiente del tamaño de la variable estimada, ya que errores con valor elevado se corresponden con valores bajos y altos de la variable dependiente estimada; sin embargo,

una distribución de residuos con problemas de heterocedasticidad da lugar a una figura como la que puede observarse en la Figura 3, en donde se manifiesta una clara relación de dependencia entre la variable estimada y el tamaño del error. En este caso los errores de mayor tamaño se corresponden con los valores más altos de la variable estimada.

FIGURA 2. RESIDUOS HOMOCEDÁSTICOS

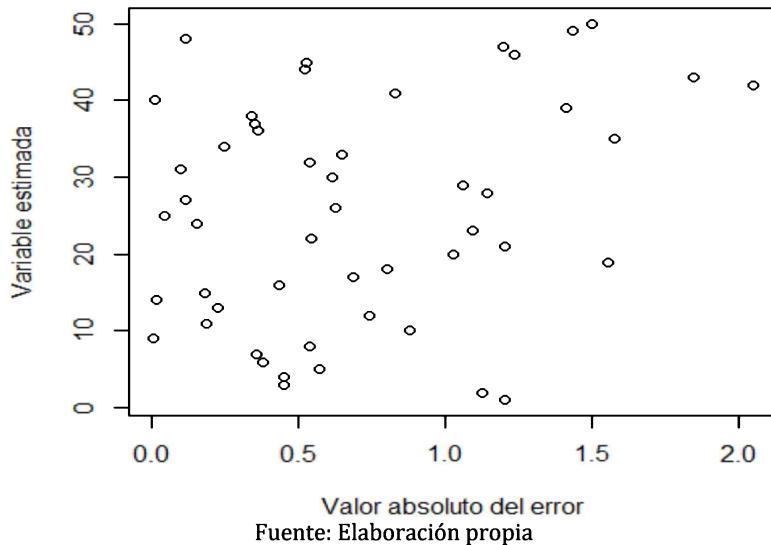
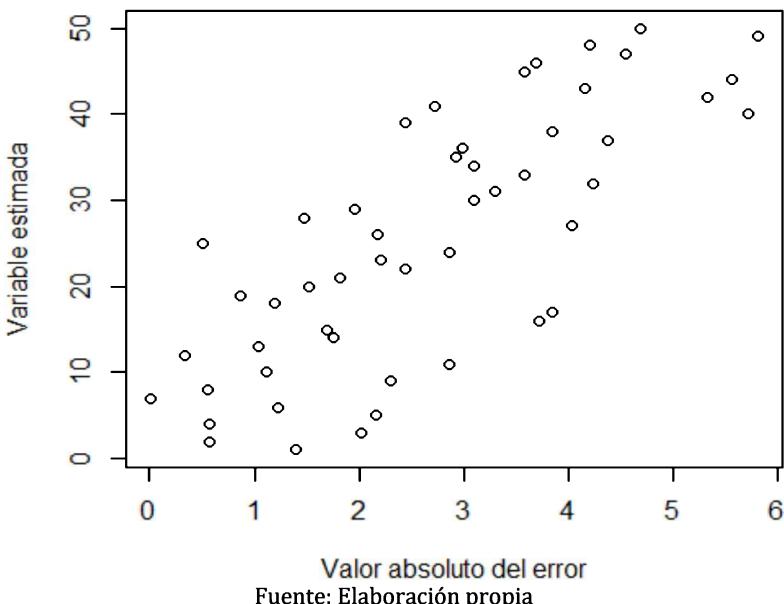
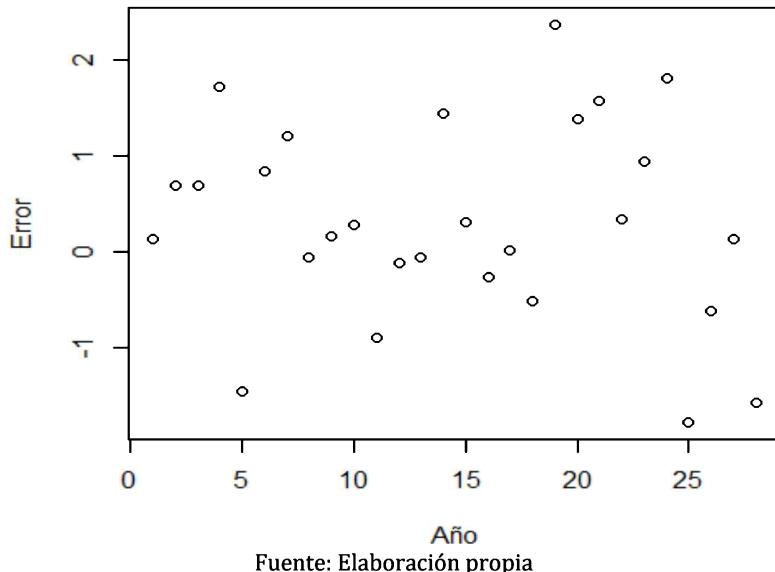


FIGURA 3. RESIDUOS HETEROCEDÁSTICOS



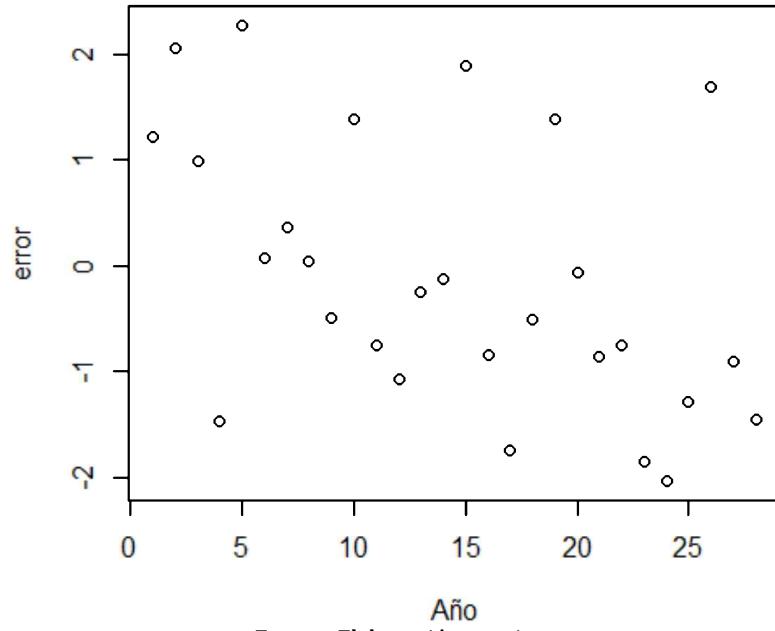
La representación gráfica de los errores en forma de serie temporal, es decir, poniendo en el eje de ordenadas los errores y en abscisas el periodo temporal en que están datados, permite apreciar la ausencia o presencia de correlación, ya que a los residuos no correlacionados (Figura 4) les corresponde una representación gráfica en la que no se aprecia pauta temporal alguna, sucediéndose de forma impredecible o aleatoria, mientras que en los residuos con problemas de autocorrelación la pauta temporal es evidente, evidenciándose que cada residuo podría ser previsto en función de la sucesión de los errores correspondientes a periodos temporales pasados (Figura 5).

FIGURA 4. RESIDUOS SIN AUTOCORRELACIÓN



Fuente: Elaboración propia

FIGURA 5. RESIDUOS CON AUTOCORRELACIÓN



Fuente: Elaboración propia

Estos problemas asociados a los errores pueden detectarse con tests estadísticos diseñados para ello.

1.2.2. Heterocedasticidad

Si existe heterocedasticidad en los residuos de nuestro modelo, esto implicará que la precisión de la explicación del modelo no es constante. En el campo de la economía, la heterocedasticidad puede provenir de cambios estructurales, eventos especiales, etc.

La consecuencia que lleva asociada es que la estimación de la varianza de los parámetros no es correcta, es decir, $\sigma^2(X'X)^{-1}$ no es una estimación consistente de $Var(\hat{\beta})$, pudiendo llevar a conclusiones erróneas sobre los β_k . Sin embargo, la expresión $\hat{\beta} = (X'X)^{-1}X'Y$ sigue siendo una estimación consistente de β .

Para tratar este problema podemos optar por:

- Mejorar la especificación del modelo (añadir, quitar o transformar variables).
- Utilizar estimadores consistentes para $Var(\hat{\beta})$, como por ejemplo los propuestos por White (1980) o por Newey y West (1987).

La detección de la heterocedasticidad se realiza a través de diversos contrastes paramétricos, entre los que cabe destacar el contraste de Bartlett (Mood, 1950), el contraste de Goldfeld-Quandt (1965) y el contraste de White (1980).

El contraste de White se basa en que, bajo la hipótesis nula de homocedasticidad, la matriz de varianzas y covarianzas de los estimadores MCO de $\hat{\beta}$ es: $\sigma^2(X'X)^{-1}$

Por el contrario, si existe heterocedasticidad, la matriz de varianzas y covarianzas viene dada por:

$$(X'X)^{-1}X'\Omega X(X'X)^{-1}, \Omega = (\sigma_1^2, \sigma_2^2, \dots, \sigma_n^2) \quad [41]$$

Por tanto, si tomamos la diferencia entre ambas queda:

$$(X'X)^{-1}X'\Omega X(X'X)^{-1} - \sigma^2(X'X)^{-1} \quad [42]$$

Por ello, basta con contrastar la hipótesis nula de que todas estas diferencias son iguales a cero, lo que equivale a contrastar que no hay heterocedasticidad.

Los pasos a seguir para realizar el contraste de White son los siguientes:

1. Estimar el modelo original y obtener la serie de residuos estimados.
2. Realizar una regresión del cuadrado de la serie de residuos obtenidos en el paso anterior sobre una constante, las variables exógenas del modelo original, sus cuadrados y los productos cruzados de segundo orden (los productos resultantes de multiplicar cada variable exógena por cada una de las restantes). Es decir, se trata de estimar por MCO la relación:

$$e_i^2 = \alpha + \varphi_1 X_{1i} + \dots + \varphi_k X_{ki} + \eta_1 X_{1i}^2 + \dots + \eta_k X_{ki}^2 + \varpi_1 X_{1i} X_{2i} + \dots + \varpi_k X_{1i} X_{ki} + \dots + \rho X_{k-1i} X_{ki} + u_i \quad [43]$$

3. Al aumentar el tamaño muestral, el producto nR^2 (donde n es el número de observaciones y R^2 es el coeficiente de determinación de la última regresión) sigue una distribución Chi-cuadrado con $p-1$ grados de libertad, donde p es el número de variables exógenas utilizadas en la segunda regresión. Se aceptará la hipótesis de existencia de heteroscedasticidad cuando el valor del estadístico supere el valor crítico de la distribución Chi-cuadrado (c) al nivel de significación estadística fijado ($nR^2 > c$).

El contraste de Breusch-Pagan es similar al de White, pero utilizando como la ecuación auxiliar en el punto 2 la siguiente: $e_i^2 = \alpha + \varphi_1 X_{1i} + \dots + \varphi_k X_{ki} + u_i$.

El contraste de heterocedasticidad de White en R se realiza a partir del de Breusch-Pagan, función bptest de la librería "lmtest", extendiendo la forma de la ecuación auxiliar:

```
library(lmtest)
m <- lm(mpg ~ hp+wt, data = mtcars)
bptest(m, ~ hp*wt + I(hp^2) + I(wt^2), data = mtcars)

studentized Breusch-Pagan test

data: m
BP = 6.5431, df = 5, p-value = 0.2569
```

Existe otra posibilidad que consiste en desarrollar el test estimando la regresión entre los residuos al cuadrado y las variables explicativas a través de redes neuronales. Para realizar en R el contraste de heterocedasticidad de White usando el test de red neuronal en el modelo estimado para las distancias y velocidades de la base de datos “cars” sobre distancias y velocidades de frenado de automóviles de los años 20 que incorpora R, primero hay que instalar (función “install”) y llamar al Package-R: “tseries”:

```
library(tseries)
y <- matrix(cars$dist, ncol=1)
x <- matrix(cars$speed, ncol=1)
white.test(x,y)

White Neural Network Test

data: x and y
X-squared = 2.4498, df = 2, p-value = 0.2938
```

1.2.3. Autocorrelación

Decimos que existe autocorrelación cuando el término de error de un modelo econométrico está correlacionado consigo mismo a través del tiempo, tal que $Cov(e_i, e_j) \neq 0$. Ello no significa que la correlación entre los errores se dé en todos los periodos, sino que puede darse tan solo entre algunos de ellos.

La razón más frecuente es que uno o varios de los parámetros β_j no son constantes durante la muestra. En el campo de la economía, puede provenir de cambios estructurales, eventos especiales, etc.

Al igual que en el caso anterior, en presencia de autocorrelación, los estimadores MCO siguen siendo insesgados pero no poseen mínima varianza, debiéndose utilizar en su lugar el método de estimación de los Mínimos Cuadrados Generalizados (MCG). Lógicamente, si la falta de constancia en el valor de los parámetros es grave, nuestra estimación no servirá para nada.

Para tratar este problema podemos optar por:

- Mejorar la especificación del modelo (añadir, quitar o transformar variables) o añadir términos de tipo MA para neutralizar la autocorrelación.
- Utilizar estimadores consistentes para $Var(\beta)$, como por ejemplo el de Newey y West (1987).

La existencia de autocorrelación en los residuos es fácilmente identificable obteniendo las funciones de autocorrelación (acf) y autocorrelación parcial (acp) de los errores mínimo-cuadráticos obtenidos en la estimación. Si dichas funciones corresponden a un ruido blanco, se constatará la ausencia de correlación entre los residuos. Sin embargo, el mero examen visual de las funciones anteriores puede resultar confuso y poco objetivo, por lo que en la práctica econométrica se utilizan diversos contrastes para la autocorrelación, siendo el más utilizado el de Durbin-Watson (1950), que pasamos a ver seguidamente.

1.2.3.1. Contraste de Durbin-Watson

Si se sospecha que el término de error del modelo econométrico tiene una estructura como la siguiente:

$$\hat{e}_t = \rho \hat{e}_{t-1} + u_t \quad [44]$$

entonces el contraste de Durbin-Watson permite contrastar la hipótesis nula de ausencia de autocorrelación.

Dicho contraste se basa en el cálculo del estadístico d , utilizando para ello los errores mínimo-cuadráticos resultantes de la estimación:

$$d = \frac{\sum_{t=2}^n (\hat{e}_t - \hat{e}_{t-1})^2}{\sum_{t=1}^n \hat{e}_t^2} \quad [45]$$

El valor del estadístico d oscila entre 0 y 4, siendo los valores cercanos a 2 los indicativos de ausencia de autocorrelación de primer orden. La interpretación exacta del test resulta compleja, ya que los valores críticos apropiados para contrastar la hipótesis nula de no autocorrelación requieren del conocimiento de la distribución de probabilidad bajo el supuesto de cumplimiento de dicha hipótesis nula, y dicha distribución depende a su vez de los valores de las variables explicativas, por lo que habría que calcularla en cada aplicación. Se ha demostrado que el valor esperado de d , cuando $\rho = 0$, está dado por la siguiente relación (Maddala, 1996).

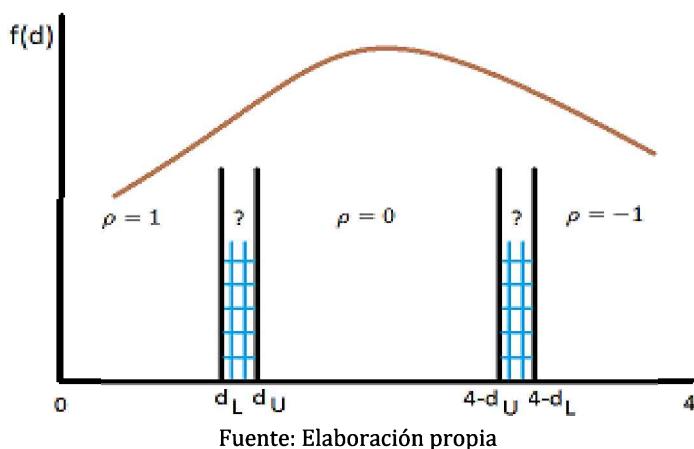
$$E(d) \approx 2 + \frac{2k}{n-k-1} \quad [46]$$

Para facilitar la interpretación del test, Durbin y Watson derivaron dos distribuciones: d_L y d_U , que no dependen de las variables explicativas y entre las cuales se encuentra la verdadera distribución de d , de forma que a partir de un determinado nivel de significación, se adoptan la siguientes reglas de decisión:

1. Si $d \leq d_L$ rechazamos la hipótesis nula de no autocorrelación frente a la hipótesis alternativa de autocorrelación positiva.
2. Si $d \geq (4 - d_L)$ rechazamos la hipótesis nula de no autocorrelación frente a la hipótesis alternativa de autocorrelación negativa.
3. Si $d_U \leq d \leq (4 - d_U)$ aceptamos la hipótesis nula de no autocorrelación.
4. Si $d_L \leq d \leq d_U$ o $(4 - d_U) \leq d \leq (4 - d_L)$ la prueba no es concluyente.

La Figura 6 muestra esta interpretación de forma gráfica.

FIGURA 6. INTERPRETACIÓN TEST DE DURBIN-WATSON



Fuente: Elaboración propia

El estadístico de Durbin-Watson es aproximadamente igual a $2(1 - r)$ en donde r es el coeficiente de autocorrelación simple muestral del retardo 1.

En R, el test de Durbin-Watson se encuentra en el Package-R “lmtest”, y su sintaxis es:

```
library(lmtest)
z <- cbind.data.frame(y, x)
dwtest(y~x, data = z)

Durbin-Watson test

data: y ~ x
DW = 1.6762, p-value = 0.09522
alternative hypothesis: true autocorrelation is greater than 0
```

1.2.3.2. Contraste de Breush-Godfrey

El test de correlación serial de Breusch-Godfrey es un test de autocorrelación en los errores y residuos estadísticos en un modelo de regresión. Hace uso de los errores generados en el modelo de regresión y un test de hipótesis derivado de éste. La hipótesis nula es que no existe correlación serial de cualquier orden de ρ .

El test es más general que el de Durbin-Watson, que solo es válido para regresores no-estocásticos y para testear la posibilidad de un modelo autorregresivo de primer orden para los errores de regresión. El test Breusch-Godfrey no tiene estas restricciones, y es estadísticamente más poderoso que el estadístico d .

Los pasos para realizar el contraste son los siguientes:

1. Estimar el modelo original y obtener la serie de residuos estimados.
2. Estimar la ecuación de regresión auxiliar:

$$\hat{e}_t = \alpha + \omega_1 X_1 + \omega_2 X_2 + \dots + \omega_k X_k + \delta_1 \hat{e}_{t-1} + \dots + \delta_p \hat{e}_{t-p} + \epsilon_t \quad [47]$$

3. Al aumentar el tamaño muestral, el producto $(n - p)R^2$ (donde n es el número de observaciones, p el número de retardos del error utilizados en la regresión auxiliar y R^2 es el coeficiente de determinación de dicha regresión) sigue una distribución Chi-cuadrado con p grados de libertad. Se aceptará la hipótesis de existencia de autocorrelación cuando el valor del estadístico supere el valor crítico de la distribución Chi-cuadrado (c) al nivel de significación estadística fijado $(n - p)R^2 > c$.

El test de Breusch-Godfrey también se realiza con la librería-R “lmtest”, y se programa para $p = 3$ del siguiente modo:

```
library(lmtest)
bgtest(y~x, order = 3, data = z)

Breusch-Godfrey test for serial correlation of order up to 3

data: y ~ x
LM test = 3.0936, df = 3, p-value = 0.3774
```

1.2.4. Deficiencias muestrales

El fenómeno de la multicolinealidad aparece cuando las variables exógenas de un modelo econométrico están correlacionadas entre sí, lo que tiene consecuencias negativas para la estimación por MCO, ya que la existencia de una relación lineal entre las variables exógenas implica que la matriz $(X'X)$ va a tener determinante cero, es decir será una matriz singular y, por tanto, no será invertible. Dado que $\hat{\beta} = (X'X)^{-1}X'y$, no será posible calcular la

estimación mínimo cuadrática de los parámetros del modelo ni, lógicamente, la varianza de los mismos. Esto es lo que se conoce por el nombre de *multicolinealidad exacta*.

1.2.5. Errores de especificación

Los errores de especificación hacen referencia a un conjunto de errores asociados a la especificación de un modelo econométrico. En concreto cabe referirse a:

- Omisión de variables relevantes.
- Inclusión de variables innecesarias.
- Adopción de formas funcionales equivocadas.

En Economía, la teoría no suele concretar la forma funcional de las relaciones que estudia. Así, por ejemplo, cuando se analiza la demanda se señala que la cantidad demandada es inversamente proporcional al precio; cuando se estudia el consumo agregado se apunta que la propensión marginal a consumir (relación entre renta y/o consumo) es mayor que cero y menor que uno. Por otro lado, es frecuente utilizar la condición “ceteris paribus” para aislar la información de otras variables relevantes que influyen y/o modifican la relación estudiada. Por esta razón, la existencia de errores de especificación en la relación estimada es un factor a considerar y a resolver en el proceso de la estimación econométrica.

Con independencia de la naturaleza de los errores de especificación, dado que en el proceso de estimación MCO deben de cumplirse determinadas hipótesis básicas (que los estimadores MCO deben de ser insesgados, eficientes y consistentes, y que el estimador de la varianza del término de error ha de ser insesgado), debemos preguntarnos: ¿qué ocurriría con estas propiedades ante errores de especificación?

La omisión de variables relevantes provoca que el estimador MCO sea sesgado en media (se obtiene un valor distinto que el que se obtendría al incorporar la variable relevante omitida) y en varianza. La incorporación de variables innecesarias determina que la varianza de los estimadores MCO sea más elevada (sesgo en la varianza), dificultando la interpretación del contraste de significación individual sobre los parámetros.

Si especificamos la forma funcional de una relación (ya sea lineal, cuadrática, cúbica, exponencial, logarítmica, etc.) y la verdadera relación presenta una forma diferente a la especificada tiene, en algunos casos, las mismas consecuencias que la omisión de variables relevantes, es decir, proporciona estimadores sesgados e inconsistentes. En general, una especificación funcional incorrecta lleva a obtener perturbaciones heteroscedásticas y/o autocorrelacionadas, o alejadas de los parámetros de la distribución del término de error del modelo correctamente especificado.

1.2.6. Métodos de selección de variables en el modelo lineal general

Una de las cuestiones más importantes a la hora de encontrar el modelo de ajuste más adecuado cuando se dispone de un amplio conjunto de variables explicativas, es la correcta especificación del modelo teórico, ya que como se ha visto la inclusión de una variable innecesaria o la omisión de una variable relevante, condiciona los estadísticos que resultan en la estimación MCO del modelo. Por otro lado, en un elevado número de explicativas no cabe descartar la existencia de correlaciones que originen un problema de multicolinealidad aproximada, y en estos casos hay que determinar cuál de ellas cabe incluir en la especificación del modelo.

En otras palabras, ante un conjunto elevado de variables explicativas debemos seleccionar de entre todas, un subconjunto de ellas que garanticen que el modelo esté lo mejor

especificado posible. Este análisis cabe hacerlo estudiando las características y propiedades de cada una de las variables independientes, a partir, por ejemplo, de los coeficientes de correlación de cada una de ellas y la dependiente, y de cada explicativa con las restantes, seleccionando modelos alternativos y observando los resultados estadísticos de la estimación MCO de cada uno de ellos. Sin embargo, en la práctica, la selección del subconjunto de variables explicativas de los modelos de regresión se deja en manos de procedimientos más o menos automáticos.

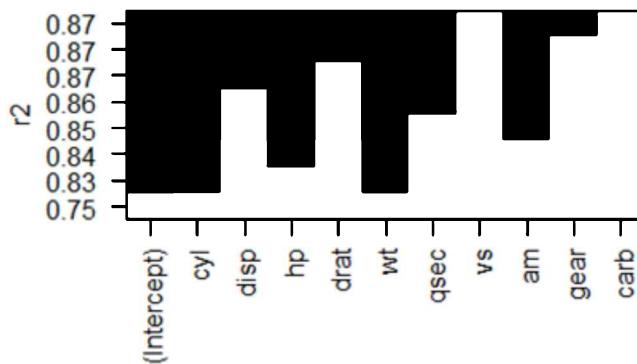
Los procedimientos más usuales son los siguientes:

- Método backward: se comienza por considerar incluidas en el modelo teórico a todas las variables disponibles y se van eliminando del modelo de una en una según su capacidad explicativa. En concreto, la primera variable que se elimina es aquella que presenta un menor coeficiente de correlación parcial con la variable dependiente –o lo que es equivalente, un menor valor del estadístico t – y así sucesivamente hasta llegar a una situación en la que la eliminación de una variable más suponga un descenso demasiado acusado en el coeficiente de determinación.
- Método forward: se comienza por un modelo que no contiene ninguna variable explicativa y se añade como primera de ellas a la que presente un mayor coeficiente de correlación –en valor absoluto– con la variable dependiente. En los pasos sucesivos se va incorporando al modelo aquella variable que presenta un mayor coeficiente de correlación parcial con la variable dependiente dadas las independientes ya incluidas en el modelo. El procedimiento se detiene cuando el incremento en el coeficiente de determinación debido a la inclusión de una nueva variable explicativa en el modelo ya no es importante.
- Método stepwise: es uno de los más empleados y consiste en una combinación de los dos anteriores. En el primer paso se procede como en el método forward pero a diferencia de éste, en el que cuando una variable entra en el modelo ya no vuelve a salir, en el procedimiento stepwise es posible que la inclusión de una nueva variable haga que otra que ya estaba en el modelo resulte redundante.

El modelo de ajuste al que se llega partiendo del mismo conjunto de variables explicativas es distinto según cuál sea el método de selección de variables elegido, por lo que la utilización de un procedimiento automático de selección de variables no significa que con él se llegue a obtener el mejor de los modelos a que da lugar el conjunto de datos con el que se trabaja.

Para realizar la selección de un modelo por cualquiera de los métodos descritos, necesitamos instalar la librería-R: "leaps". Una vez instalada ejecutamos el siguiente Chunk, para seleccionar según el método "forward":

```
library(leaps)
regfit.fwd = regsubsets(mpg~., data=mtcars, method="forward")
plot(regfit.fwd, scale="r2")
```



```
summary(regfit.fwd)
Subset selection object
Call: regsubsets.formula(mpg ~ ., data = mtcars, method = "forward")
10 Variables (and intercept)
  Forced in  Forced out
cyl      FALSE      FALSE
disp     FALSE      FALSE
hp      FALSE      FALSE
drat     FALSE      FALSE
wt      FALSE      FALSE
qsec     FALSE      FALSE
vs      FALSE      FALSE
am      FALSE      FALSE
gear     FALSE      FALSE
carb     FALSE      FALSE
1 subsets of each size up to 8
Selection Algorithm: forward
      cyl disp hp drat wt qsec vs am gear carb
1 ( 1 )   *   *   *   *   *   *   *   *   *   *
2 ( 1 )   *   *   *   *   *   *   *   *   *   *
3 ( 1 )   *   *   *   *   *   *   *   *   *   *
4 ( 1 )   *   *   *   *   *   *   *   *   *   *
5 ( 1 )   *   *   *   *   *   *   *   *   *   *
6 ( 1 )   *   *   *   *   *   *   *   *   *   *
7 ( 1 )   *   *   *   *   *   *   *   *   *   *
8 ( 1 )   *   *   *   *   *   *   *   *   *   *
```

Si queremos ver las estimaciones MCO de los parámetros del modelo 8:

```
coef(regfit.fwd, 8)
(Intercept)      cyl      disp        hp        drat        wt
12.56350226 -0.23126963  0.01611609 -0.02339020  0.70893592 -4.08155351
       qsec       am       gear
0.91812006  2.47759723  0.50403957
```

1.3. MODELOS CON VARIABLES CUALITATIVAS EXPLICATIVAS

1.3.1. Introducción

En un modelo econométrico, las variables representan a los conceptos u operaciones económicas que queremos analizar. Normalmente utilizamos variables cuantitativas, es decir, aquellas cuyos valores vienen expresados de forma numérica; sin embargo, también existe la posibilidad de incluir en el modelo econométrico información cualitativa, siempre que esta pueda expresarse de esa forma.

Las variables cualitativas expresan cualidades o atributos de los agentes o individuos (sexo, religión, nacionalidad, nivel de estudios, etc.) y también recogen acontecimientos

extraordinarios como guerras, terremotos, climatologías adversas, huelgas, cambios políticos, etc.

No cabe duda de que una forma de recoger factores de este tipo sería la utilización de variables proxy o aproximadas a las variables utilizadas. Por ejemplo, si quiero utilizar una variable que mida el nivel cultural de un país (variable cualitativa) puedo utilizar como variable proxy el número de bibliotecas existentes en un país, o representar una climatología adversa a partir de las temperaturas medias o precipitaciones. Sin embargo, no siempre es posible encontrar este tipo de variables y, en cualquier caso, debemos de ser conscientes de la posible existencia de errores en la definición de la variable.

Puesto que las variables cualitativas normalmente recogen aspectos de la presencia o no de determinado atributo (ser hombre o mujer, tener estudios universitarios o no tenerlos, etc.) se utilizan variables construidas artificialmente, llamadas también ficticias o dummy, que generalmente toman dos valores, 1 o 0, según se dé o no cierta cualidad o atributo. Habitualmente a la variable ficticia se le asigna el valor 1 en presencia de la cualidad y 0 en caso contrario. Las variables que toman valores 1 y 0, también reciben el nombre de variables dicotómicas o binarias.

Las variables dicotómicas pueden combinarse para caracterizar variables definidas por su pertenencia o no a un grupo. Si incluyo una variable cualitativa que me define la pertenencia o no de un país a un grupo, por ejemplo, renta alta, media y baja, introduciré tres variables cualitativas en el modelo asociadas a la pertenencia o no a cada grupo; la primera caracterizaría a los individuos con renta alta, la segunda a los individuos con renta media, y la tercera a los individuos con renta baja.

Los modelos que utilizan variables cualitativas como regresores se diferencian en dos grupos, los modelos de Análisis de la Varianza o modelos ANOVA, que únicamente incluyen variables cualitativas como regresores; y los modelos de Análisis de la Covarianza o modelos ANCOVA que incluyen tanto variables cualitativas como cuantitativas. Los modelos ANOVA son muy utilizados en Sociología, Psicología, Educación, etc.; en Economía son más comunes los modelos ANCOVA.

1.3.2. Modelos ANOVA: efectos fijos

Un problema estadístico clásico es la comparación de medias de dos distribuciones normales. Supongamos que las observaciones de la variable Y_i provienen de dos distribuciones normales con medias μ_1 y μ_2 y varianza común σ^2 . El tamaño de la primera distribución se circunscribe a las n_1 primeras observaciones, y el de la segunda a las $n - n_1$ restantes. Queremos contrastar la hipótesis $H_0: \mu_1 = \mu_2$ frente a la alternativa $H_1: \mu_1 \neq \mu_2$ al nivel de significación α .

Este contraste de igualdad de medias cabe formularlo en el marco del modelo lineal general. Así, bajo H_0 tenemos el siguiente modelo de regresión múltiple utilizando variables Dummy:

$$Y_i = \mu_1 D_{1i} + \mu_2 D_{2i} + u_i \quad [48]$$

donde D_{1i} toma valor uno en las n_1 primeras observaciones y cero en las restantes, y D_{2i} toma cero en las n_1 primeras observaciones y uno en las restantes, o lo que es lo mismo $D_{2i} = 1 - D_{1i}$.

Este modelo ANOVA puede formularse también a partir de las siguientes expresiones:

$$Y_i = \beta_0 + \mu_1 D_{1i} + u_i \quad [49]$$

$$Y_i = \beta_0 + \mu_2 D_{2i} + u_i \quad [50]$$

El contraste de igualdad de medias se realiza a través del contraste de significación global, para el que construimos el estadístico experimental $F_{exp} = \frac{\frac{R^2}{k-1}}{\frac{1-R^2}{n-k}}$, siendo el estadístico teórico F_{tco} . La hipótesis se rechazaría con la regla de decisión $F_{exp} > F_{tco}$.

Si se utiliza la especificación del modelo [48], el coeficiente asociado a la categoría D_1 vendrá dado por la suma $(\beta_0 + \mu_1)$, y para D_2 por β_1 . Si queremos contrastar la hipótesis de igualdad de medias en ambos grupos, equivaldría a contrastar la hipótesis nula $H_0: \mu_1 = 0$.

Si se utiliza la especificación del modelo [49], el coeficiente asociado a la categoría D_2 vendrá dado por la suma $(\beta_0 + \mu_2)$, y para D_1 por β_0 . Si queremos analizar la hipótesis de igualdad de medias en ambos grupos, equivaldría a contrastar la hipótesis nula $H_0: \mu_2 = 0$

El análisis de la varianza (ANOVA) se debe al estadístico-genético Sir Ronald Aylmer Fisher (1890-1962), autor del libro "Statistics Methods for Research Workers" publicado en 1925 y pionero de la aplicación de métodos estadísticos en el diseño de experimentos, introduciendo el concepto de aleatorización. Fue creado para resolver diversos problemas agrícolas y tiene por objetivo descomponer la variabilidad de los datos asociados a un experimento en componentes independientes, las cuales, son asignables a distintas causas. El problema que vamos a estudiar es, si disponemos de n elementos que se diferencian en un factor, por ejemplo, estudiantes de distintas aulas, vehículos de distinta marca, efectos de distintos medicamentos, productos sometidos a diferentes canales de distribución, fondos de inversión en distintos momentos temporales, etc. En cada elemento estudiado (fondos, productos, personas, vehículos,...) observamos cierta característica, variable respuesta, que varía aleatoriamente de un elemento a otro, como el consumo de combustible, la altura de cada persona, las ventas de un producto, etc. Se desea conocer si hay o no relación entre el valor medio esperado de la característica objeto de estudio y el factor.

Por tanto, este análisis surge en el ámbito del diseño de experimentos, cuya metodología estudia cómo realizar comparaciones lo más homogéneas posibles, para aumentar, en consecuencia, la probabilidad de detectar cambios o identificar variables influyentes.

Respecto a la aleatorización, se establece el siguiente principio: todos los factores no controlados por el experimentador y que pueden influir en los resultados se asignan al azar a las observaciones. La aleatorización es fundamental, ya que:

- a) Previene la existencia de sesgos.
- b) Evita la dependencia entre observaciones.
- c) Confirma la validez de los procedimientos estadísticos más comunes.

El problema que vamos a estudiar es, si disponemos de n elementos que se diferencian en un factor, por ejemplo, estudiantes de distintas aulas, vehículos de distinta marca, efectos de distintos medicamentos, productos sometidos a diferentes canales de distribución, fondos de inversión en distintos momentos temporales, etc. En cada elemento estudiado (fondos, productos, personas, vehículos,...) observamos cierta característica, variable respuesta, que varía aleatoriamente de un elemento a otro, como el consumo de

combustible, la altura de cada persona, las ventas de un producto, etc. Se desea conocer si hay o no relación entre el valor medio esperado de la característica objeto de estudio y el factor.

Por tanto podemos decir que el análisis de la varianza es un método estadístico para determinar si una variable determinada, llamada dependiente y medida en escala no métrica, toma valores medios iguales o distintos en los grupos que forma otra variable, llamada independiente y en escala métrica. Como ejemplo, partiendo de la base de datos "mtcars", vamos a analizar si la autonomía de los vehículos, medida en millas por galón de combustible, es la misma según su número de cilindros (4, 6 o 8).

1.3.2.1. Análisis de la varianza de un factor

El diseño que hemos planteado se conoce como análisis de la varianza de un factor, puesto que se considera la influencia de una sola variable (número de cilindros). En el siguiente apartado se verá el análisis de la varianza con dos o más factores, donde como ejemplo se considerará la influencia conjunta de dos variables independientes, el número de cilindros (4, 6 o 8) y el tipo de transmisión (automática [0] o manual [1]).

Planteamiento

El objetivo del análisis de la varianza es contrastar la homogeneidad simultánea de k poblaciones ($k \geq 2$) que siguen la ley de probabilidad $N(\mu_i, \sigma^2)$ para $i=1, \dots, L$, con varianza común desconocida. Es necesario suponer que las distribuciones tienen varianza común para plantear el modelo.

La homogeneidad de las poblaciones se contrastará mediante la igualdad simultánea de sus medias μ_i , es decir, trataremos de contrastar la hipótesis nula:

$$H_0: \mu_1 = \mu_2 = \dots = \mu_L = \mu \quad (\text{homogeneidad de las } k \text{ poblaciones}) \quad [51]$$

frente a la alternativa

$$H_1: \exists \mu_i \neq \mu_j \quad (\text{al menos existen dos medias que son diferentes}) \quad [52]$$

El objetivo del análisis es comprobar la hipótesis (nula) de que la media es igual en las distintas poblaciones. Si la hipótesis es cierta, entonces las diferencias entre los valores obtenidos y los valores medios son debidas al azar (variabilidad aleatoria) y no a un efecto sistemático.

Se denomina factor a la variable que supuestamente ejerce una influencia sobre la variable dependiente. En nuestro ejemplo, la variable dependiente sería la autonomía de los vehículos, medida en millas por galón, mientras que el factor el número de cilindros.

Los niveles del factor serán cada uno de los valores posibles del factor (4, 6, 8).

El error muestral es el error debido a la aleatoriedad en la selección de los elementos muestrales. Se representa por: $u_{ij} = y_{ij} - \mu_i \quad \forall i, j$

Un modelo es replicado si el experimento se repite varias veces para cada nivel de factor.

Un modelo de análisis de la varianza se llama equilibrado (balanced) cuando todos los grupos que forma la variable independiente son de igual tamaño, es decir cuando el número de observaciones para cada nivel del factor es siempre el mismo (el experimento se repite

para cada nivel del factor el mismo número de veces). En caso de no ser igual, se llama no equilibrado (unbalanced).

Modelo

El planteamiento anteriormente expuesto da lugar a la formulación de un modelo matemático. Dicho modelo es el siguiente:

$$y_{ij} = \mu + \alpha_i + u_{ij} \quad [53]$$

donde $\mu_i = \mu + \alpha_i$ serán las medias estimadas por el modelo, siendo, por tanto, μ la media general, α_i el efecto del factor en cada nivel y $\sum \alpha_i = 0$.

La estimación de estos parámetros teóricos se realiza del siguiente modo:

$$\hat{\mu} = \bar{y}_{..} \quad \hat{\alpha}_i = \bar{y}_{i..} - \bar{y}_{..} \quad \hat{\mu}_i = \hat{\mu} + \hat{\alpha}_i = \bar{y}_{i..} \quad \hat{u}_{ij} = e_{ij} = y_{ij} - \bar{y}_{i..} \quad [54]$$

Siendo $\bar{y}_{i..} = \frac{\sum_{j=1}^{n_i} y_{ij}}{n_i}$ la media muestral para cada grupo (en nuestro caso las autonomías medias de los vehículos según tengan 4, 6 o 8 cilindros), con n_i el número de observaciones o muestras, y $\bar{y}_{..} = \frac{\sum_{i=1}^L \sum_{j=1}^{n_i} y_{ij}}{n}$ la media global, o autonomía media del conjunto de vehículos analizados.

El modelo estimado queda, por tanto:

$$y_{ij} = \bar{y}_{..} + (\bar{y}_{i..} - \bar{y}_{..}) + (y_{ij} - \bar{y}_{i..}) \quad [55]$$

Tabla ANOVA

Partiendo de la expresión anterior, la variabilidad total (VT) de la variable respuesta será igual a:

$$VT = \sum_{i=1}^L \sum_{j=1}^{n_i} (y_{ij} - \bar{y}_{..})^2 = \sum_{i=1}^L \sum_{j=1}^{n_i} [(\bar{y}_{i..} - \bar{y}_{..}) + (y_{ij} - \bar{y}_{i..})]^2 \quad [56]$$

Es fácilmente demostrable que:

$$2 \sum_{i=1}^L \sum_{j=1}^{n_i} (\bar{y}_{i..} - \bar{y}_{..})(y_{ij} - \bar{y}_{i..}) = 2 \sum_{i=1}^L (\bar{y}_{i..} - \bar{y}_{..}) \sum_{j=1}^{n_i} (y_{ij} - \bar{y}_{i..}) = 0 \quad [57]$$

y por tanto:

$$VT = \sum_{i=1}^L \sum_{j=1}^{n_i} (\bar{y}_{i..} - \bar{y}_{..})^2 + \sum_{i=1}^L \sum_{j=1}^{n_i} (y_{ij} - \bar{y}_{i..})^2 = VE + VNE \quad [58]$$

El primer sumatorio de cuadrados constituye la variabilidad explicada (VE) por el modelo y el segundo la variabilidad no explicada (VNE), es decir:

$$VE = \sum_{i=1}^L \sum_{j=1}^{n_i} (\bar{y}_{i..} - \bar{y}_{..})^2 = \sum_{i=1}^L n_i (\bar{y}_{i..} - \bar{y}_{..})^2 = \sum_{i=1}^L n_i \hat{\alpha}_i^2 \quad [59]$$

$$VNE = \sum_{i=1}^L \sum_{j=1}^{n_i} (y_{ij} - \bar{y}_{i..})^2 = \sum_{i=1}^L \sum_{j=1}^{n_i} e_{ij}^2 \quad [60]$$

La tabla ANOVA queda, por tanto, como sigue:

TABLA 2. TABLA ANOVA. MODELO CON UN FACTOR

Fuente de variación	Suma de cuadrados	Grados de libertad	Cuadrado medio	Estadístico F
Entre grupos	VE	$L-1$	$\frac{VE}{L-1} = S_e^2$	$\frac{S_e^2}{S_R^2}$
Dentro de los grupos	VNE	$n-L$	$\frac{VNE}{n-L} = S_R^2$	
Total	VT	$n-1$	$\frac{VT}{n-1} = S_y^2$	

Fuente: Elaboración propia

Si las medias de la autonomía de los vehículos fueran muy distintas entre sí y, además, la varianza dentro de cada grupo fuera pequeña (los grupos muy distintos entre sí y, dentro de cada grupo un comportamiento muy homogéneo), la variabilidad total sería debida a la diferencia entre grupos. Sin embargo, si las medias en los grupos fueran muy parecidas entre sí, dado que se supone que la varianza dentro de cada grupo es la misma, la variabilidad total sería debida a la variabilidad dentro de los grupos.

Para contrastar la hipótesis nula de igualdad de medias se construye el estadístico F, que compara la variabilidad debida a las diferencias entre los grupos y la variabilidad debida a la diferencia dentro de los grupos. El estadístico F nos ha de decir si tenemos pruebas suficientes" para rechazar o aceptar la hipótesis nula, y se distribuye según una F de Snedecor con L-1 grados de libertad en el numerador y N-L en el denominador. Si el p-valor asociado a este estadístico es menor que el grado de significación fijado, rechazaremos la hipótesis de igualdad de medias.

En términos del modelo planteado, la hipótesis nula es: $H_0: \alpha_i = 0 \quad \forall i$.

En el siguiente "Chunk" construimos la tabla anova con la función "aov" (Analysis Of Variance) y estimamos un modelo ANOVA con la función "model.tables". Finalmente, representamos las tres distribuciones a través de un diagrama de cajas múltiple.

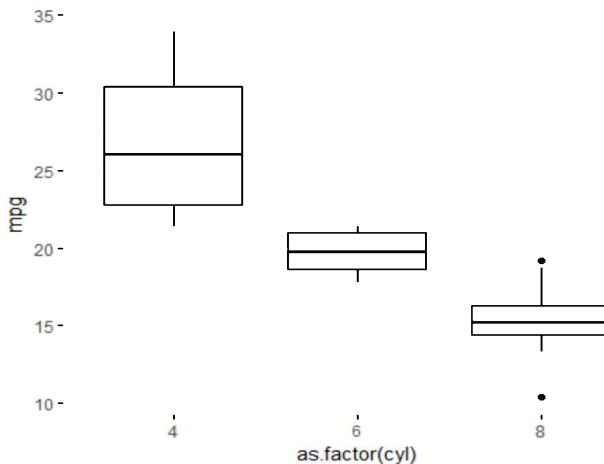
```
summary(aov(mpg~as.factor(cyl),data=mtcars))
      Df Sum Sq Mean Sq F value    Pr(>F)
as.factor(cyl)  2  824.8   412.4    39.7 4.98e-09 ***
Residuals     29  301.3    10.4
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

model.tables(aov(mpg~as.factor(cyl),data=mtcars),type="means")
Tables of means
Grand mean

20.09062

as.factor(cyl)
  4   6   8
 26.66 19.74 15.1
rep 11.00  7.00 14.0

library(ggplot2)
ggplot(mtcars, aes(as.factor(cyl), mpg)) + geom_boxplot()
```



La hipótesis más relevante que deben cumplir los datos para poder aplicar un análisis de la varianza es la de *homocedasticidad*, es decir, que la varianza de la variable dependiente (millas por galón) es constante en los grupos definidos por el factor (número de cilindros). Esta prueba puede realizarse a través de la función `leveneTest` de la librería `car`, y como vemos, existe heterocedasticidad, dado que el grupo de 4 cilindros presenta una variabilidad mayor al resto, tal como se puede intuir en el diagrama de cajas. También puede verse como a mayor número de cilindros la variabilidad disminuye.

```
library(car)
leveneTest(mtcars$mpg, as.factor(mtcars$cyl), center=mean)
Levene's Test for Homogeneity of Variance (center = mean)
  Df F value    Pr(>F)
group  2 6.4843 0.004703 **
29
---
Signif. codes:  0 '****' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Algunos autores (Uriel, 1995; Stevens, 1986) afirman, sin embargo, que el estadístico F no se ve muy afectado por el hecho de que no exista homocedasticidad siempre que las muestras de los diferentes grupos sean del mismo o similar tamaño. Se afirma que el estadístico se verá afectado cuando la razón entre el tamaño muestral del grupo de mayor tamaño y el más pequeño sea superior a 2.

Por otro lado, en el caso particular de que la hipótesis de homogeneidad de varianzas sea rechazada debido a que la varianza cambia con la media, existe una familia de transformaciones que proporciona, en general, homogeneidad en varianzas. Esta familia es de la forma:

$$T(X) = \begin{cases} X^p & \text{si } p \neq 0 \\ \ln(X) & \text{si } p = 0 \end{cases} \quad [61]$$

donde $p = 1 - \alpha$, siendo α la pendiente de la regresión lineal entre las medias y las desviaciones típicas de los grupos. Aplicando el logaritmo neperiano, es decir, con $p=0$ y $\alpha=1$, obtenemos ya un p-valor no significativo en el test de Levene, y por tanto homogeneidad de varianzas, obteniendo como resultado del ANOVA lo siguiente:

```
library(car)
lnmpg <- log(mtcars$mpg)
cylf <- as.factor(mtcars$cyl)
leveneTest(lnmpg, cylf, center=mean)
```

```

Levene's Test for Homogeneity of Variance (center = mean)
  Df F value Pr(>F)
group  2  1.6622 0.2073
      29

modelo.aov <- aov(lnmpg~cylf)
summary(modelo.aov)

      Df Sum Sq Mean Sq F value    Pr(>F)
cylf      2 2.0081  1.0040   39.31 5.52e-09 ***
Residuals 29 0.7407  0.0255
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

model.tables(modelo.aov)

Tables of effects

cylf
      4       6       8
  0.3129 0.02292 -0.2573
rep 11.0000 7.00000 14.0000

```

Existen en R diferentes pruebas de cara a verificar la hipótesis de normalidad. Usando la variable transformada, y dado que la muestra es pequeña, optamos por el test de Shapiro-Wilk, donde como vemos, al no ser significativo (p-valor mayor que 0,05), podemos asumir que las distribuciones son normales:

```

logmpg <- split(log(mtcars$mpg), as.factor(mtcars$cyl))
for (i in 1:length(logmpg)){
  logmpg.i <- as.vector(logmpg[i])
  print(shapiro.test(logmpg.i[[1]]))
}

Shapiro-Wilk normality test

data: logmpg.i[[1]]
W = 0.91722, p-value = 0.2961

Shapiro-Wilk normality test

data: logmpg.i[[1]]
W = 0.89736, p-value = 0.3153

Shapiro-Wilk normality test

data: logmpg.i[[1]]
W = 0.89341, p-value = 0.09052

```

Para muestras grandes, es preferible la prueba de Lilliefors (Kolmogorov-Smirnov), disponible en el paquete *nortest*. A pesar de que continuamente se alude al test Kolmogorov-Smirnov como un test válido para contrastar la normalidad, esto no es del todo cierto. Este test asume que se conoce la media y varianza poblacional, lo que en la mayoría de los casos no es posible. Esto hace que el test sea muy conservador y poco potente. Para solventar este problema, se desarrolló una modificación del Kolmogorov-Smirnov conocida como test Lilliefors. El test Lilliefors asume que la media y varianza son desconocidas, estando especialmente desarrollado para contrastar la normalidad. Es la alternativa al test de Shapiro-Wilk cuando el número de observaciones es mayor de 50.

Los resultados para nuestro ejemplo serían los siguientes:

```

library(nortest)
for (i in 1:length(logmpg)){
  logmpg.i <- as.vector(logmpg[i])
  print(lilliefors.test(logmpg.i[[1]]))
}

Lilliefors (Kolmogorov-Smirnov) normality test

data: logmpg.i[[1]]
D = 0.16847, p-value = 0.5164

Lilliefors (Kolmogorov-Smirnov) normality test

data: logmpg.i[[1]]
D = 0.2343, p-value = 0.2907

Lilliefors (Kolmogorov-Smirnov) normality test

data: logmpg.i[[1]]
D = 0.19852, p-value = 0.1412

```

El *test de Jarque-Bera* no requiere estimaciones de los parámetros que caracterizan la normal. El estadístico de Jarque-Bera cuantifica que tanto se desvían los coeficientes de asimetría y curtosis de los esperados en una distribución normal. Su formulación es la siguiente:

$$JB = \frac{n-k+1}{6} (S^2 + \frac{1}{4}(C - 3)^2) \quad [62]$$

donde n es el número de observaciones (o grados de libertad en general); S es la asimetría de la muestra, C la curtosis y k el número de regresores.

El estadístico de Jarque-Bera se distribuye asintóticamente como una distribución chi cuadrado con dos grados de libertad y puede usarse para probar la hipótesis nula de que los datos pertenecen a una distribución normal. Como hipótesis nula se considera que la asimetría y el exceso de curtosis, conjuntamente, son nulos, es decir:

$$H_0: C = 0; S = 3 \quad [63]$$

Puede calcularse mediante la función *jarque.bera.test()* del paquete *tseries*. Los resultados con este test serían:

```

library(tseries)
for (i in 1:length(logmpg)){
  logmpg.i <- as.vector(logmpg[i])
  print(jarque.bera.test(logmpg.i[[1]]))
}

Jarque Bera Test

data: logmpg.i[[1]]
X-squared = 0.99886, df = 2, p-value = 0.6069

Jarque Bera Test

data: logmpg.i[[1]]
X-squared = 0.71437, df = 2, p-value = 0.6996

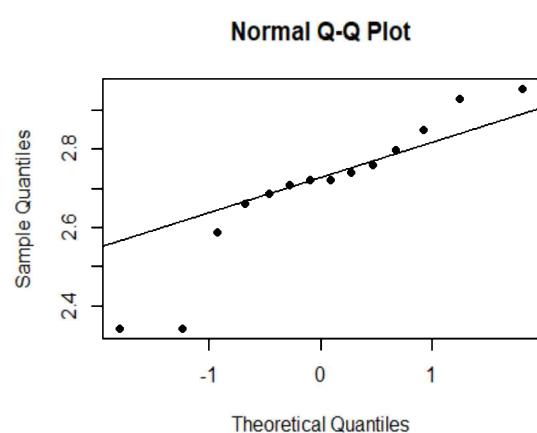
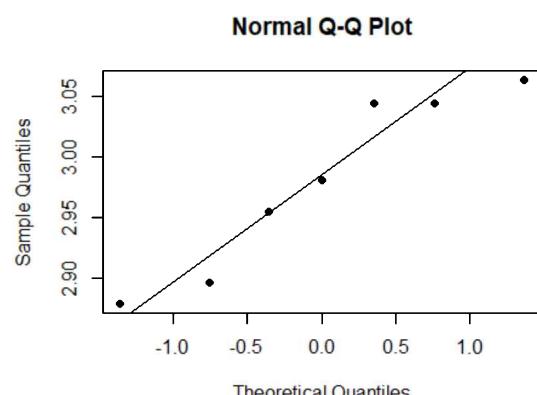
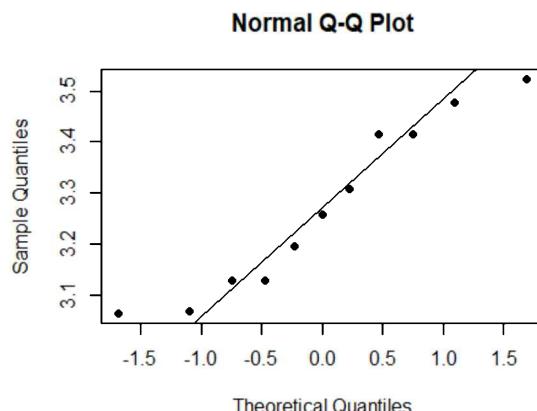
Jarque Bera Test

```

```
data: logmpg.i[[1]]
X-squared = 1.549, df = 2, p-value = 0.4609
```

También podemos utilizar métodos gráficos, representando el histograma junto a la curva normal teórica o el denominado gráfico de cuantiles teóricos (gráfico Q-Q). Este último, se realiza a partir de la función *qqnorm*.

```
for (i in 1:length(logmpg)){
  logmpg.i <- as.vector(logmpg[i])
  qqnorm(logmpg.i[[1]], pch = 19, col = "gray50")
  qqline(logmpg.i[[1]])
}
```



Con estos resultados, lo que sabemos es que alguna de las medias es diferente. Para averiguar que medias difieren de las otras, se pueden efectuar comparaciones de medias dos a dos y evaluar en cada caso las diferencias con el resto de los grupos.

Los dos procedimientos más utilizados para la comparación de medias por pares son los tests de Tukey y de Scheffe. El primero será preferible en modelos balanceados, en el caso de no balanceados, como el ejemplo que estamos realizando, optaremos por Scheffe.

El test de Scheffe está disponible en el paquete *agricolae*, disponiendo de dos opciones de presentación de resultados. Si queremos agrupar los niveles del factor con medias similares, especificamos la opción *group=TRUE*.

```
library(agricolae)
scheffe.test(modelo.aov, "cylf", group=TRUE, console=TRUE,
             main="Autonomía según cilindrada")
```

Study: Autonomía según cilindrada

Scheffe Test for lnmpg

Mean Square Error : 0.02553996

cylf, means

	lnmpg	std	r	Min	Max
4	3.270454	0.16762136	11	3.063391	3.523415
6	2.980439	0.07431982	7	2.879198	3.063391
8	2.700171	0.18113932	14	2.341806	2.954910

Alpha: 0.05 ; DF Error: 29

Critical Value of F: 3.327654

Groups according to probability of means differences and alpha level(0.05)

Means with the same letter are not significantly different.

	lnmpg	groups
4	3.270454	a
6	2.980439	b
8	2.700171	c

Si lo que buscamos es realizar todas las comparaciones de pares posibles, debemos especificar la opción *group=FALSE*.

```
library(agricolae)
scheffe.test(modelo.aov, "cylf", group=FALSE, console=TRUE,
             main="Autonomía según cilindrada")
```

Study: Autonomía según cilindrada

Scheffe Test for lnmpg

Mean Square Error : 0.02553996

cylf, means

	lnmpg	std	r	Min	Max
4	3.270454	0.16762136	11	3.063391	3.523415
6	2.980439	0.07431982	7	2.879198	3.063391
8	2.700171	0.18113932	14	2.341806	2.954910

Alpha: 0.05 ; DF Error: 29

Critical Value of F: 3.327654

Comparison between treatments means

	Difference	pvalue	sig	LCL	UCL
4 - 6	0.2900143	0.0032	**	0.1145018	0.4655268
4 - 8	0.5702825	0.0000	***	0.4240221	0.7165429
6 - 8	0.2802682	0.0029	**	0.1122278	0.4483086

Como puede observarse, a mayor número de cilindros se obtiene una menor autonomía, es decir, un mayor consumo de combustible, siendo todas las comparaciones estadísticamente significativas.

Para realizar el test de Tukey, se ha de utilizar la función `glht` del paquete `multcomp`. Los resultados para nuestro ejemplo serían los siguientes:

```
require(multcomp)
summary(glht(modelo.aov, linfct = mcp(cylf = "Tukey")))

Simultaneous Tests for General Linear Hypotheses

Multiple Comparisons of Means: Tukey Contrasts

Fit: aov(formula = lnmpg ~ cylf)

Linear Hypotheses:
Estimate Std. Error t value Pr(>|t|)
6 - 4 == 0 -0.29001   0.07727 -3.753  0.00234 **
8 - 4 == 0 -0.57028   0.06439 -8.857 < 0.001 ***
8 - 6 == 0 -0.28027   0.07398 -3.788  0.00207 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
(Adjusted p values reported -- single-step method)
```

1.3.2.2. Análisis de la varianza con dos factores

En el ejemplo anterior estudiábamos el posible efecto del número de cilindros del motor en el consumo de combustible de los vehículos. Imaginemos ahora que se desea estudiar además el posible efecto del tipo de transmisión (automática [0] o manual [1]).

Podríamos pensar que es necesario llevar a cabo un análisis de la varianza de un factor para cada uno de ellos, sin embargo es posible trabajar con las dos variables independientes de manera simultánea en un único estudio.

El diseño experimental que se sigue en estos casos, es conocido como diseño factorial, donde dos o más variables independientes son manipuladas en un único estudio de tal forma que en el análisis se representan todas las posibles combinaciones de los diversos niveles de las variables independientes. Teóricamente, aunque un diseño factorial puede incluir cualquier número de variables independientes, en la práctica resulta poco adecuado utilizar más de tres o cuatro.

En síntesis, en nuestro caso queremos constatar si:

- La autonomía de los vehículos varía dependiendo del número de cilindros del motor.
- La autonomía de los vehículos varía dependiendo del tipo de transmisión.
- La autonomía de los vehículos varía de forma conjunta según el número de cilindros del motor y el tipo de transmisión.

Modelo

El modelo matemático subyacente es el siguiente:

$$y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + u_{ijk} \quad [64]$$

siendo α_i el efecto del primer factor, β_j el del segundo y $(\alpha\beta)_{ij}$ la interacción entre ambos.

La estimación de estos parámetros teóricos se realiza, análogamente al caso anterior, del siguiente modo:

$$\begin{aligned} \hat{\mu} &= \bar{y}_{...} & \hat{\alpha}_i &= \bar{y}_{i..} - \bar{y}_{...} & \hat{\beta}_j &= \bar{y}_{.j.} - \bar{y}_{...} & \hat{(\alpha\beta)}_{ij} &= \bar{y}_{ij.} - \bar{y}_{i..} - \bar{y}_{.j.} + \bar{y}_{...} \\ \hat{\mu}_{ij} &= \hat{\mu} + \hat{\alpha}_i + \hat{\beta}_j + \hat{(\alpha\beta)}_{ij} & \hat{u}_{ijk} &= e_{ijk} = y_{ijk} - \bar{y}_{ij.} \end{aligned} \quad [65]$$

Siendo $\bar{y}_{i..}$, $\bar{y}_{.j.}$ y $\bar{y}_{ij.}$ las medias muestrales para el primer factor, para el segundo y para el cruce de ambos respectivamente..

El modelo estimado queda por tanto:

$$y_{ijk} = \bar{y}_{...} + (\bar{y}_{i..} - \bar{y}_{...}) + (\bar{y}_{.j.} - \bar{y}_{...}) + (\bar{y}_{ij.} - \bar{y}_{i..} - \bar{y}_{.j.} + \bar{y}_{...}) + (y_{ijk} - \bar{y}_{ij.}) \quad [66]$$

Tabla ANOVA

De una manera similar al caso de un factor, la varianza total puede descomponerse del siguiente modo (donde I es el número total de grupos del primer factor, en nuestro caso, los tres niveles del número de cilindros; J es el número total de grupos del segundo factor, en nuestro caso 2, transmisión automática y manual):

$$VT = \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^{n_{ij}} (y_{ijk} - \bar{y}_{...})^2 = \sum_{i=1}^I \sum_{j=1}^J n_{ij} (\bar{y}_{ij.} - \bar{y}_{...})^2 + \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^{n_{ij}} (y_{ijk} - \bar{y}_{ij.})^2 = VE + VNE \quad [67]$$

El primer término de la expresión anterior refleja la suma de los cuadrados con respecto a la media muestral global. El triple sumatorio se refiere: el primero a los niveles del primer factor (número de cilindros), el segundo a los del segundo factor (tipo de transmisión) y el tercero a los datos individuales del cruce de los dos anteriores.

El segundo término de la expresión se descompone en dos sumandos. El primero refleja las diferencias al cuadrado entre la media de cada celda y la media global, y constituye la variabilidad explicada por el modelo *VE*, mientras que el segundo sumando es la suma de cuadrados residual o variación no explicada *VNE*.

Lo interesante está en analizar qué ocasiona las diferencias del primer sumando, es decir, las diferencias al cuadrado entre la media de cada celda. Puede demostrarse que:

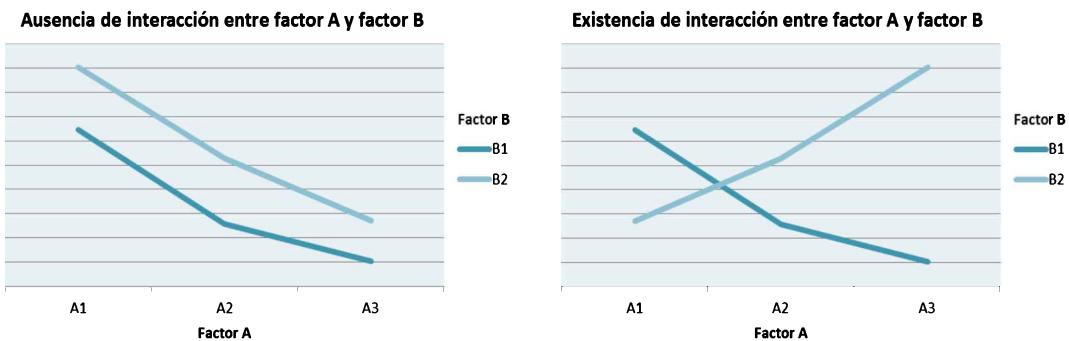
$$\begin{aligned} VE &= \sum_{i=1}^I \sum_{j=1}^J n_{ij} (\bar{y}_{ij.} - \bar{y}_{...})^2 = \sum_{i=1}^I \sum_{j=1}^J n_{ij} (\bar{y}_{i..} - \bar{y}_{...})^2 + \sum_{i=1}^I \sum_{j=1}^J n_{ij} (\bar{y}_{.j.} - \bar{y}_{...})^2 + \\ &+ \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^{n_{ij}} (\bar{y}_{ij.} - \bar{y}_{i..} - \bar{y}_{.j.} + \bar{y}_{...})^2 = VE_A + VE_B + VE_{AxB} \end{aligned} \quad [68]$$

Se ve claramente en esta expresión que los dos primeros sumandos corresponden a sumas; las denominaremos *VE_A* y *VE_B* y reflejan la variabilidad explicada por cada uno de los dos factores. El último término refleja la interacción de los factores A y B, es decir, el efecto de los dos factores que no es debido individualmente a ninguno de ellos. El efecto interacción, será denominado *VE_{AxB}*.

La interacción será significativa si el comportamiento de la variable respuesta en un factor es diferente en función de los niveles del otro factor. Así, partiendo de nuestro ejemplo, hemos visto que en media el consumo de combustible aumenta a medida que aumenta el número de los cilindros del motor. Si consideramos el tipo de transmisión, y resultara que esto ocurre en los coches automáticos, pero en los manuales disminuyera el consumo con el aumento del número de cilindros, existirá efecto de interacción.

Gráficamente, se puede observar a través de un gráfico de líneas múltiple, tal como muestra la Figura 7, existiendo interacción si las líneas se cruzan.

FIGURA 7. DETECCIÓN GRÁFICA DE INTERACCIÓN ENTRE DOS FACTORES



Fuente: Elaboración propia

En síntesis, la suma total de cuadrados puede descomponerse, en el caso de dos factores del siguiente modo:

$$VT = VE_A + VE_B + VEAxB + VNE \quad [69]$$

Al igual que en el caso de un factor, cada suma de cuadrados tiene sus propios grados de libertad, igual al número de niveles del factor menos 1. Los grados de libertad asociados a la interacción será la multiplicación de los asociados a ambos factores, es decir, $(I - 1) * (J - 1)$. Se muestra a continuación la tabla ANOVA donde se construye cada uno de los estadísticos F que, ahora, han de permitirnos determinar si el efecto de cada factor por separado y la interacción de ambos, son o no significativos.

TABLA 3. TABLA ANOVA. MODELO CON DOS FACTORES E INTERACCIÓN

Fuente de variación	Suma de cuadrados	Grados de libertad	Cuadrado medio (MC)	E(MC)	Estadístico F
Factor A	$VE_A = JK \sum_i \hat{\alpha}_i^2$	$I - 1$	$MCF_A = \frac{VE_A}{I - 1}$	$\sigma^2 + JK \frac{\sum \alpha_i^2}{I - 1}$	$F = \frac{MCF_A}{MCR}$
Factor B	$VE_B = IK \sum_j \hat{\beta}_j^2$	$J - 1$	$MCF_B = \frac{VE_B}{J - 1}$	$\sigma^2 + IK \frac{\sum \beta_j^2}{J - 1}$	$F = \frac{MCF_B}{MCR}$
Interacción	$VE_{AxB} = K \sum_{ij} (\hat{\alpha}\hat{\beta})_{ij}^2$	$(I - 1)(J - 1)$	$MCF_{AxB} = \frac{VE_{AxB}}{(I - 1)(J - 1)}$	$\sigma^2 + K \frac{\sum (\alpha\beta)_j^2}{(I - 1)(J - 1)}$	$F = \frac{MCF_{AxB}}{MCR}$
Error o residual	$VNE = \sum_{ijk} e_{ijk}$	$IJ(K - 1) = n - IJ$	$MCR = \frac{VNE}{n - IJ}$	σ^2	
Total	$VT = VE_A + VE_B + VNE$	$n - 1$	$MCT = \frac{VT}{n - 1}$		

Fuente: Elaboración propia

La columna E(MC) hace referencia al valor esperado o esperanza matemática de las varianzas o medias cuadráticas de las distintas fuentes de variación. Esta esperanza, como puede observarse, es igual a la suma de la varianza residual y la variabilidad debida al propio factor. Al construir los estadísticos F como el cociente entre la variabilidad estimada para cada factor y la variabilidad del error, en términos de esperanza, obtenemos (tomando como ejemplo el factor A):

$$\frac{E(MCF_A)}{E(MCR)} = \frac{\sigma^2 + JK \frac{\sum \alpha_i^2}{I-1}}{\sigma^2} \quad [70]$$

Por tanto, si consideramos válida la hipótesis nula correspondiente a $\alpha_i = 0 \forall i$, la esperanza matemática del estadístico F será igual a 1, aumentando su valor a medida que aumenta la variabilidad explicada por el factor.

Se muestra a continuación el ANOVA planteado en R, donde como vemos, tanto el tipo de transmisión como la interacción de éste con el número de cilindros resultan ser no significativos.

```
lnmpg <- log(mtcars$mpg)
cylf <- as.factor(mtcars$cyl)
amf <- as.factor(mtcars$am)

# Test de Levene para factor Número de cilindros
leveneTest(lnmpg,cylf,center=mean)
Levene's Test for Homogeneity of Variance (center = mean)
  Df F value Pr(>F)
group  2  1.6622 0.2073
      29

# Test de Levene para factor Tipo de transmisión
leveneTest(lnmpg,amf,center=mean)
Levene's Test for Homogeneity of Variance (center = mean)
  Df F value Pr(>F)
group  1  0.4653 0.5004
      30

# Test de Shapiro-Wilk para factor Número de cilindros
logmpg <- split(log(mtcars$mpg), as.factor(mtcars$cyl))
for (i in 1:length(logmpg)){
  logmpg.i <- as.vector(logmpg[i])
  print(shapiro.test(logmpg.i[[1]]))
}

Shapiro-Wilk normality test

data: logmpg.i[[1]]
W = 0.91722, p-value = 0.2961

Shapiro-Wilk normality test

data: logmpg.i[[1]]
W = 0.89736, p-value = 0.3153

Shapiro-Wilk normality test

data: logmpg.i[[1]]
W = 0.89341, p-value = 0.09052

# Test de Shapiro-Wilk para factor Tipo de transmisión
logmpg <- split(log(mtcars$mpg), as.factor(mtcars$am))
```

```

for (i in 1:length(logmpg)){
  logmpg.i <- as.vector(logmpg[i])
  print(shapiro.test(logmpg.i[[1]]))
}

Shapiro-Wilk normality test

data: logmpg.i[[1]]
W = 0.9557, p-value = 0.4909

Shapiro-Wilk normality test

data: logmpg.i[[1]]
W = 0.94212, p-value = 0.485

# Modelo ANOVA
modelo.aov <- aov(lnmpg~cylf+amf+cylf:amf)
summary(modelo.aov)

      Df Sum Sq Mean Sq F value    Pr(>F)
cylf     2 2.0081  1.0040  40.362 1.06e-08 ***
amf      1 0.0681  0.0681   2.737   0.110
cylf:amf  2 0.0258  0.0129   0.519   0.601
Residuals 26 0.6468  0.0249
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

model.tables(modelo.aov)

Tables of effects

cylf
      4       6       8
0.3129 0.02292 -0.2573
rep 11.0000 7.00000 14.0000

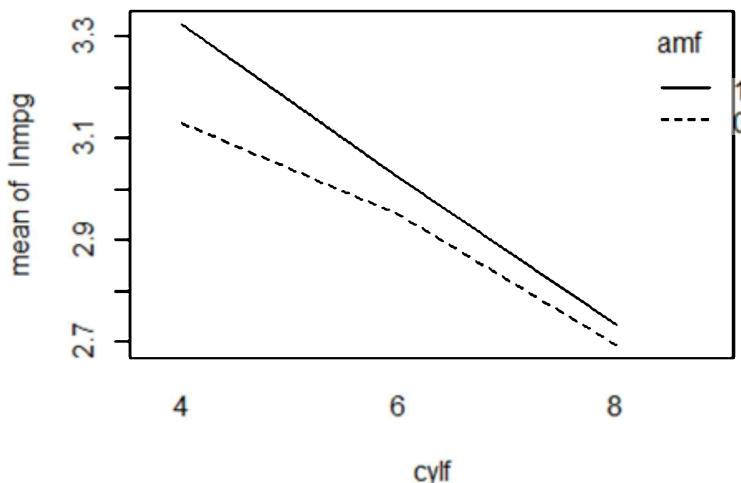
amf
      0       1
-0.03253 0.04754
rep 19.00000 13.00000

cylf:amf
      amf
cylf 0       1
4    -0.061  0.023
rep  3.000  8.000
6    0.015  -0.020
rep  4.000  3.000
8    0.010  -0.061
rep 12.000  2.000

```

Por último, el gráfico de interacción se construye del siguiente modo:

```
interaction.plot(cylf,amf,lnmpg)
```



donde si bien puede apreciarse un ligero mayor consumo de los vehículos automáticos (menor autonomía), dicha diferencia, eliminando el efecto del número de cilindros, no llega a ser estadísticamente significativa, tal y como hemos visto.

1.3.2.3. Diseño en bloques aleatorizados. El efecto Yule-Simpson

Si estuviéramos interesados en evaluar el impacto del tipo de transmisión, y no incluimos en nuestro modelo ANOVA la variable número de cilindros, obtendríamos lo siguiente:

```

lnmpg <- log(mtcars$mpg)
amf <- as.factor(mtcars$am)
modelo.aov <- aov(lnmpg~amf)
summary(modelo.aov)
      Df Sum Sq Mean Sq F value    Pr(>F)
amf      1 0.9275  0.9275   15.28 0.000491 ***
Residuals 30 1.8213  0.0607
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

model.tables(modelo.aov)
Tables of effects

  amf
    0          1
  -0.1408    0.2058
  rep     19.0000 13.0000

```

Como vemos, los resultados son ahora estadísticamente significativos, y llegaríamos a concluir, erróneamente, que los vehículos de transmisión manual consumen menos carburante. Este error se suele denominar *efecto Yule-Simpson* o *paradoja de Simpson*, y es debido a la influencia que ejerce la variable número de cilindros sobre el tipo de transmisión. Si analizamos la tabla de contingencia entre ambos factores, podemos observar, que los fabricantes de vehículos de la muestra a medida que aumentan el número de cilindros se decantan por incorporar la transmisión automática, vista como un extra. Así, el 63,2% de los vehículos automáticos tienen 8 cilindros, mientras que en el caso de la transmisión manual, el 61,6% son de 4 cilindros.

```

cylf <- as.factor(mtcars$cyl)
levels(cylf) <- c("4 cilindros", "6 cilindros", "8 cilindros")
amf <- as.factor(mtcars$am)
levels(amf) <- c("Automático", "Manual")
df <- data.frame(cylf, amf)

```

```

# Tabla de contingencia Cilindros - Tipo de transmisión
tabla <- table(df$cyl, df$amf)

# Frecuencias
tabla


|             | Automático | Manual |
|-------------|------------|--------|
| 4 cilindros | 3          | 8      |
| 6 cilindros | 4          | 3      |
| 8 cilindros | 12         | 2      |



# Porcentajes columna (si opción no se especifica calcularíamos porcentajes tabla y
# si es igual a 1 porcentajes fila)
prop.table(tabla,2)


|             | Automático | Manual    |
|-------------|------------|-----------|
| 4 cilindros | 0.1578947  | 0.6153846 |
| 6 cilindros | 0.2105263  | 0.2307692 |
| 8 cilindros | 0.6315789  | 0.1538462 |


```

La estrategia más adecuada en este caso sería agrupar los vehículos en bloques, definidos por el número de cilindros, y a continuación seleccionar en cada bloque igual número de vehículos de transmisión manual y automática.

Por tanto, un diseño en bloques aleatorizados es un diseño con aleatorización restringida, en el cual las unidades experimentales son primero clasificadas en grupos homogéneos, llamados bloques, y los tratamientos son entonces asignados aleatoriamente dentro de los bloques.

Esta estrategia de diseño mejora efectivamente la precisión en las comparaciones al reducir la variabilidad residual. Dicho diseño es quizás el diseño experimental más ampliamente utilizado.

El modelo planteado corresponde al de dos factores sin interacción, es decir:

$$y_{ijk} = \mu + \alpha_i + \beta_j + u_{ijk} \quad [71]$$

donde la variabilidad total se desglosa como: $VT = VE_A + VE_B + VNE$.

Bajo este esquema, el factor A (tipo de transmisión) será considerado como variable *tratamiento o factor principal* y el factor B (número de cilindros) como *factor bloque*. En el caso en que este factor sea no significativo, se eliminará del modelo, ganando con ello grados de libertad en la estimación del error.

1.3.2.4. Modelos con más de dos factores

La generalización del modelo de dos factores con interacción a uno con tres o más factores es inmediata.

Por ejemplo, es fácil comprobar que en el diseño factorial completo de tres factores con réplicas, incluyendo la interacción de tercer orden, quedaría:

$$y_{ijkl} = \mu + \alpha_i + \beta_j + \gamma_k + (\alpha\beta)_{ij} + (\alpha\gamma)_{ik} + (\beta\gamma)_{jk} + (\alpha\beta\gamma)_{ijk} + u_{ijkl} \quad [72]$$

la variabilidad total (VT) se desglosa en:

$$VT = VE_A + VE_B + VE_C + VE_{AxB} + VE_{AxC} + VE_{BxC} + VE_{AxBxC} + VNE \quad [73]$$

y la tabla ANOVA:

TABLA 4. TABLA ANOVA. MODELO CON TRES FACTORES E INTERACCIÓN

Fuente de variación	Suma de cuadrados	Grados de libertad	Cuadrado medio	Estadístico <i>F</i>
Factor A	VE_A	$I - 1$	$MCF_A = \frac{VE_A}{I - 1}$	$F = \frac{MCF_A}{MCR}$
Factor B	VE_B	$J - 1$	$MCF_B = \frac{VE_B}{J - 1}$	$F = \frac{MCF_B}{MCR}$
Factor C	VE_C	$K - 1$	$MCF_C = \frac{VE_C}{K - 1}$	$F = \frac{MCF_C}{MCR}$
Interacción AxB	VE_{AXB}	$(I - 1)(J - 1)$	$MCF_{AXB} = \frac{VE_{AXB}}{(I - 1)(J - 1)}$	$F = \frac{MCF_{AXB}}{MCR}$
Interacción AxC	VE_{AXC}	$(I - 1)(K - 1)$	$MCF_{AXC} = \frac{VE_{AXC}}{(I - 1)(K - 1)}$	$F = \frac{MCF_{AXC}}{MCR}$
Interacción BxC	VE_{BXC}	$(J - 1)(K - 1)$	$MCF_{BXC} = \frac{VE_{BXC}}{(J - 1)(K - 1)}$	$F = \frac{MCF_{BXC}}{MCR}$
Interacción AxBxC	VE_{AXBXC}	$(I - 1)(J - 1)(K - 1)$	$MCF_{AXBXC} = \frac{VE_{AXBXC}}{(I - 1)(J - 1)(K - 1)}$	$F = \frac{MCF_{AXBXC}}{MCR}$
Error o residual	VNE	$n - IJK$	$MCR = \frac{VNE}{n - IJK}$	
Total	VT	$n - 1$	$MCT = \frac{VT}{n - 1}$	

Fuente: Elaboración propia

Al igual que en el modelo de regresión, es conveniente la aplicación de técnicas de selección de factores (forward, backward, stepwise). Al eliminar los efectos no significativos del modelo, además de facilitar su interpretación, ganamos grados de libertad en el término error y reducimos su variabilidad.

1.3.2.5. Diseños experimentales: cuadrado latino, cuadrado latino replicado y cuadrado grecolatino

Cuadrado latino

El diseño de Cuadrado Latino es un plan experimental donde cada tratamiento solo aparece una vez en cada fila y en cada columna. Se consideran, por tanto, tres factores con K niveles cada uno y sin efectos de interacción.

Los Cuadrados Latinos se representan tradicionalmente por tablas $K \times K$, con letras latinas (A, B, C, ...) en las casillas para simbolizar los niveles de la variable de tratamiento, de ahí su nombre. Se trata, por tanto, de un diseño con K^2 observaciones.

Según Dowley y Wearden (1991), los diseños de Cuadrado Latino son formatos económicos porque no requieren todas las combinaciones posibles entre las tres dimensiones de variación.

FIGURA 8. EJEMPLOS DE CUADRADOS LATINOS CON 3 Y 4 NIVELES

$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \\ 3 & 1 & 2 \end{bmatrix}$	$\begin{bmatrix} a & b & d & c \\ b & c & a & d \\ c & d & b & a \\ d & a & c & b \end{bmatrix}$
---	--

Fuente: Elaboración propia

El popular rompecabezas Sudoku es un caso especial de cuadrado latino; toda solución de un Sudoku es un cuadrado latino. Un Sudoku impone una restricción adicional a los subgrupos de 3×3 , estos solo deben contener los dígitos del 1 al 9 (en la versión estándar).

El rompecabezas conocido como Diamante 16 ilustra un concepto generalizado de la ortogonalidad de los cuadrados latinos: el cuadrado ortogonal o "Matrices ortogonales" (ortogonal en el sentido combinatorio y no en un sentido algebraico-lineal).

El estadístico inglés Ronald Fisher se valió del uso de los cuadrados latinos para mejorar significativamente los métodos agrícolas, cuando se hallaba investigando la eficacia de los fertilizantes en el rendimiento de las cosechas. Buscó la manera de plantar cosechas en similares condiciones de suelo, de modo que la calidad de la tierra no fuese un factor indeseable que influyese en el rendimiento de la cosecha. Si bien la única manera de asegurarse de tener condiciones idénticas de tierra era utilizar siempre el mismo suelo, en la práctica esto es casi imposible, pues se deberían desenterrar y volver a plantar las cosechas varias veces.

Por otra parte, aunque sí se pudiera hacer esto último, las condiciones meteorológicas serían otro factor indeseable. Para evitar esto, por ejemplo, en un caso en que se tuviese un campo cuadrado dividido en 16 parcelas, se puede concebir un cuadrado latino en el que la descripción del campo sea tal que la calidad del suelo varíe «vertical» y «horizontalmente». Entonces se aplican al azar los 4 fertilizantes («a», «b», «c», y «d») con la única condición de que cada fertilizante aparezca una sola vez en cada fila y en cada columna. De esta manera se busca eliminar la variación de la calidad de la tierra. Si hubiese otro factor que pudiese influir en el rendimiento, por ejemplo, el momento del día (A, B, C, D) en que se aplica el tratamiento, entonces puede utilizarse un cuadrado latino ortogonal al anterior donde se identifiquen dichos momentos del día. De esta manera cada pareja momento-fertilizante se aplicará en una única parcela. Así, un plan podría ser:

FIGURA 9. PLAN CUADRADO LATINO

plan, MOMENTO

a, A	b, B	c, C	d, D
b, C	a, D	d, A	c, B
c, D	d, C	a, B	b, A
d, B	c, A	b, D	a, C

Fuente: Elaboración propia

Los pasos a seguir para la realización de un experimento de este tipo serán:

- Elegir un cuadrado latino de las tablas.
- Asignar aleatoriamente los tratamientos a las filas y columnas.

Modelo

Bajo el esquema comentado, el modelo a ajustar será el siguiente:

$$y_{ij(k)} = \mu + \alpha_i + \beta_j + \gamma_k + u_{ij(k)} \quad [74]$$

siendo los estimadores de los parámetros:

$$\begin{aligned} \hat{\mu} &= \bar{y}_{...} & \hat{\alpha}_i &= \bar{y}_{i..} - \bar{y}_{...} & \hat{\beta}_j &= \bar{y}_{.j.} - \bar{y}_{...} & \hat{\gamma}_k &= \bar{y}_{..k} - \bar{y}_{...} \\ e_{ij(k)} &= y_{ij(k)} - \bar{y}_{i..} - \bar{y}_{.j.} - \bar{y}_{..k} + 2\bar{y}_{...} \end{aligned} \quad [75]$$

Tabla ANOVA

Bajo las especificaciones de este modelo, la tabla ANOVA se construye de la siguiente manera:

TABLA 5. TABLA ANOVA

Fuente de variación	Suma de cuadrados	Grados de libertad	Cuadrado medio (MC)	E(MC)	Estadístico F
Fila	$VE_A = K \sum_i \hat{\alpha}_i^2$	$K - 1$	$MCF_A = \frac{VE_A}{K - 1}$	$\sigma^2 + \frac{K}{K - 1} \sum_i \alpha_i^2$	$F = \frac{MCF_A}{MCR}$
Columna	$VE_B = K \sum_j \hat{\beta}_j^2$	$K - 1$	$MCF_B = \frac{VE_B}{K - 1}$	$\sigma^2 + \frac{K}{K - 1} \sum_j \beta_j^2$	$F = \frac{MCF_B}{MCR}$
Tratamiento	$VE_C = K \sum_k \hat{\gamma}_k^2$	$K - 1$	$MCF_C = \frac{VE_C}{K - 1}$	$\sigma^2 + \frac{K}{K - 1} \sum_k \gamma_k^2$	$F = \frac{MCF_C}{MCR}$
Error o residual	$VNE = \sum_{ij} e_{ij(k)}^2$	$(K - 1)(K - 2)$	$MCR = \frac{VNE}{(K - 1)(K - 2)}$	σ^2	
Total	$VT = VE_A + VE_B + VE_C + VNE$	$n - 1$	$MCT = \frac{VT}{n - 1}$		

Fuente: Elaboración propia

siendo los efectos fila y columna efectos bloque.

Las principales formas estándar de los Cuadrados Latinos se encuentran en las tablas publicadas por Fisher y Yates (1953) y Cochran y Cox (1957).

Cuadrado latino replicado

Modelo

Supongamos ahora que en nuestro experimento realizamos R replicaciones de un cuadrado latino. En tal caso, la réplica se incorporará al modelo como efecto bloque, quedando por tanto como sigue.

$$y_{ij(k)r} = \mu + \alpha_i + \beta_j + \gamma_k + \delta_r + u_{ij(k)r} \quad [76]$$

siendo los estimadores de los parámetros:

$$\begin{aligned}\hat{\mu} &= \bar{y}_{...} & \hat{\alpha}_i &= \bar{y}_{i...} - \bar{y}_{...} & \hat{\beta}_j &= \bar{y}_{.j..} - \bar{y}_{...} & \hat{\gamma}_k &= \bar{y}_{..k.} - \bar{y}_{...} & \hat{\delta}_r &= \bar{y}_{...r} - \bar{y}_{...} \\ e_{ij(k)r} &= y_{ij(k)r} - \bar{y}_{i...} - \bar{y}_{.j..} - \bar{y}_{..k.} - \bar{y}_{...r} + 3\bar{y}_{...}\end{aligned}\quad [77]$$

Tabla ANOVA

Bajo las especificaciones de este modelo, la tabla ANOVA se construye de la siguiente manera:

TABLA 6. TABLA ANOVA

Fuente de variación	Suma de cuadrados	Grados de libertad	Cuadrado medio (MC)	E(MC)	Estadístico F
	$VE_A = KR \sum_i \hat{\alpha}_i^2$	$K - 1$	$MCF_A = \frac{VE_A}{K - 1}$	$\sigma^2 + KR \frac{\sum \alpha_i^2}{K - 1}$	$F = \frac{MCF_A}{MCR}$
	$VE_B = KR \sum_j \hat{\beta}_j^2$	$K - 1$	$MCF_B = \frac{VE_B}{K - 1}$	$\sigma^2 + KR \frac{\sum \beta_j^2}{K - 1}$	$F = \frac{MCF_B}{MCR}$
	$VE_C = KR \sum_k \hat{\gamma}_k^2$	$K - 1$	$MCF_C = \frac{VE_C}{K - 1}$	$\sigma^2 + KR \frac{\sum \gamma_k^2}{K - 1}$	$F = \frac{MCF_C}{MCR}$
	$VE_D = K^2 \sum_r \hat{\delta}_r^2$	$R - 1$	$MCF_D = \frac{VE_D}{R - 1}$	$\sigma^2 + K^2 \frac{\sum \delta_r^2}{R - 1}$	$F = \frac{MCF_D}{MCR}$
	$VNE = \sum_{ijr} e_{ij(k)r}^2$	$RK^2 - (3(K - 1) + (R - 1) + 1)$	$MCR = \frac{VNE}{RK^2 - 3(K - 1) - R}$	σ^2	
	$VT = VE_A + VE_B + VE_C + VE_D + VNE$	$RK^2 - 1 = n - 1$	$MCT = \frac{VT}{n - 1}$		

Fuente: Elaboración propia

Cuadrado grecolatino

En este caso tenemos 4 factores con K niveles cada uno y sin efectos de interacción. El número de observaciones es K^2 .

FIGURA 10. ESQUEMA DEL CUADRADO GRECOLATINO

		Factor 2 (columna)					
		1	2	.	.	.	K
Factor 1 (fila)	1						
	2						
	.			A α			
	.						
	.						
	K						

Fuente: Elaboración propia

Los niveles del primer tratamiento (Factor 3) se representan con letras latinas (A, B, C, ...) y los del segundo (Factor 4) con letras griegas ($\alpha, \beta, \gamma, \dots$), de ahí su nombre. En ambos casos, cada nivel aparecerá una sola vez por fila y columna.

Al igual que en el caso anterior, los pasos a seguir serán:

- Elegir un cuadrado grecolatino de las tablas.
- Asignar aleatoriamente los tratamientos a las filas y columnas.

El modelo, por tanto, sería el siguiente:

$$y_{ij(kh)} = \mu + \alpha_i + \beta_j + \gamma_k + \epsilon_h + u_{ij(kh)} \quad [78]$$

1.3.3. Modelos de componentes de la varianza: efectos aleatorios

Como hemos visto hasta ahora, el factor es una variable independiente o experimental controlada por el investigador. Puede tomar pocos o muchos valores o niveles, a cada uno de los cuales se asignan los grupos o muestras.

Si se toman L niveles y las inferencias se refieren exclusivamente a los L niveles y no a otros que podrían haber sido incluidos, el ANOVA se llama de *efectos fijos, sistemático o paramétrico*. El interés del diseño se centra en saber si esos niveles concretos difieren entre sí.

Cuando los niveles son muchos y se seleccionan al azar L niveles, pero las inferencias se desean hacer respecto al total de niveles, el análisis de varianza se denomina de *efectos aleatorios*. La idea básica es que el investigador no tiene interés en niveles particulares del factor.

Los modelos basados en factores aleatorios se denominan *Componentes de la varianza*, y lo que se persigue es evaluar la cantidad de variación de la variable dependiente que se asocia con una o más variables de efectos aleatorios.

1.3.3.1. Modelo de un factor aleatorio

En estos modelos se mide que parte de la variabilidad de la respuesta es debida al factor. Dado que el factor es una variable aleatoria, la variabilidad observada en el experimento será:

$$\sigma_y^2 = \sigma_\alpha^2 + \sigma^2 \quad [79]$$

Por tanto, lo que se pretende averiguar es si σ_α^2 es igual a 0 (hipótesis nula) o, por el contrario, el factor incorpora al experimento una variabilidad significativa.

El modelo queda, por tanto:

$$y_{ik} = \mu + \alpha_i + u_{ik} \quad [80]$$

con $\alpha_i \rightarrow N(0, \sigma_\alpha^2)$ y $u_{ik} \rightarrow N(0, \sigma^2)$, siendo la hipótesis nula del modelo:

$$H_0: \sigma_\alpha^2 = 0 \quad [81]$$

De forma análoga al modelo con efectos fijos, los estimadores de los parámetros del modelo serían:

$$\hat{\mu} = \bar{y}_{..} \quad \hat{\alpha}_i = \bar{y}_{i..} - \bar{y}_{..} \quad e_{ik} = y_{ik} - \bar{y}_{i..} \quad [82]$$

donde la descomposición de la variabilidad quedaría:

$$VE(\alpha) = K \sum_{i=1}^I (\bar{y}_{i..} - \bar{y}_{..})^2 \quad [83]$$

$$VNE = \sum_{i=1}^I \sum_{k=1}^K (y_{ik} - \bar{y}_{i..})^2 = \sum_{i=1}^I \sum_{k=1}^K e_{ik}^2 \quad [84]$$

$$VT = VE(\alpha) + VNE \quad [85]$$

quedando la tabla ANOVA de la siguiente manera:

TABLA 7. TABLA ANOVA

Fuente de variación	Suma de cuadrados	Grados de libertad	Cuadrado medio (MC)	E(MC)	Estadístico F
Entre grupos (VE)	$VE(\alpha)$	$I - 1$	$\hat{S}_\alpha^2 = \frac{VE(\alpha)}{I - 1}$	$\sigma^2 + K\sigma_\alpha^2$	$F = \frac{\hat{S}_\alpha^2}{\hat{S}_R^2}$
Interna (VNE)	VNE	$I(K - 1)$	$\hat{S}_R^2 = \frac{VNE}{n - I}$	σ^2	
Total	$VT = VE + VNE$	$(I - 1) + I(K - 1) = n - 1$	$S_y^2 = \frac{VT}{n - 1}$		

Fuente: Elaboración propia

Si α fuera un efecto fijo, recordar que la esperanza de la media cuadrática sería:

$$E(MS(\alpha)) = \sigma^2 + K \frac{\sum_{i=1}^I \alpha_i^2}{I-1} \quad [86]$$

es decir, cambian las esperanzas de las medias cuadráticas.

Teniendo en cuenta las esperanzas de las medias cuadráticas, las variabilidades del error (interna) y entre grupos se estimarían del siguiente modo:

$$\hat{\sigma}^2 = \hat{S}_R^2 \quad [87]$$

$$\hat{S}_\alpha^2 = \hat{\sigma}^2 + K \hat{\sigma}_\alpha^2 \Rightarrow \hat{\sigma}_\alpha^2 = \frac{\hat{S}_\alpha^2 - \hat{S}_R^2}{K} \quad [88]$$

1.3.3.2. Modelo de dos factores aleatorios con interacción

En este modelo la variabilidad observada en el experimento será:

$$\sigma_y^2 = \sigma_\alpha^2 + \sigma_\beta^2 + \sigma_{\alpha\beta}^2 + \sigma^2 \quad [89]$$

El modelo a estimar sería el siguiente:

$$y_{ijk} = \mu + \alpha_i + \beta_j + \alpha\beta_{ij} + u_{ijk} \quad [90]$$

con $\alpha_i \rightarrow N(0, \sigma_\alpha^2)$, $\beta_j \rightarrow N(0, \sigma_\beta^2)$ y $u_{ijk} \rightarrow N(0, \sigma^2)$, siendo las hipótesis nulas del modelo:

$$H_{01}: \sigma_\alpha^2 = 0 \quad [91]$$

$$H_{02}: \sigma_\beta^2 = 0 \quad [92]$$

$$H_{03}: \sigma_{\alpha\beta}^2 = 0 \quad [93]$$

Los estimadores de los parámetros del modelo se calculan del siguiente modo:

$$\begin{aligned} \hat{\mu} &= \bar{y}_{...} & \hat{\alpha}_i &= \bar{y}_{i..} - \bar{y}_{...} & \hat{\beta}_j &= \bar{y}_{.j.} - \bar{y}_{...} \\ \hat{\alpha}\hat{\beta}_j &= \bar{y}_{ij.} - \bar{y}_{i..} - \bar{y}_{.j.} + \bar{y}_{...} & e_{ijk} &= y_{ijk} - \bar{y}_{ij.} \end{aligned} \quad [94]$$

y la descomposición de la variabilidad quedaría:

$$VE(\alpha) = JK \sum_{i=1}^I \hat{\alpha}_i^2 \quad VE(\beta) = IK \sum_{i=1}^I \hat{\beta}_j^2 \quad VE(\alpha\beta) = K \sum_{i=1}^I \hat{\alpha}_i \hat{\beta}_j^2 \quad [95]$$

$$VNE = \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^K (y_{ijk} - \bar{y}_{ij.})^2 = \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^K e_{ijk}^2 \quad [96]$$

$$VT = VE(\alpha) + VE(\beta) + VE(\alpha\beta) + VNE \quad [97]$$

quedando la tabla ANOVA de la siguiente manera:

TABLA 8. TABLA ANOVA

Fuente de variación	Suma de cuadrados	Grados de libertad	Cuadrado medio (MC)	E(MC)	Estadístico F
Entre grupos (VE)	$VE(\alpha)$	$I - 1$	$\hat{S}_\alpha^2 = \frac{VE(\alpha)}{I - 1}$	$\sigma^2 + K\sigma_\alpha^2$	$F = \frac{S_\alpha^2}{S_R^2}$
Interna (VNE)	VNE	$I(K - 1)$	$\hat{S}_R^2 = \frac{VNE}{n - I}$	σ^2	
Total	$VT = VE + VNE$	$(I - 1) + I(K - 1) = n - 1$	$S_y^2 = \frac{VT}{n - 1}$		

Fuente: Elaboración propia

Como vemos, el denominador del test en los efectos principales es ahora la Media Cuadrática de la interacción.

1.3.3.3. Modelos con más de dos factores y modelos mixtos

Al igual que en el caso de los modelos ANOVA con efectos fijos, analizamos a continuación el modelo con tres efectos aleatorios y las correspondientes interacciones. La generalización a más de dos factores es también directa, si bien se complica la construcción de las pruebas F para los tres efectos principales.

Los grados de libertad y las esperanzas de las medias cuadráticas quedarán en este caso:

TABLA 9. GRADOS DE LIBERTAD Y ESPERANZAS

Fuente de variación	Grados de libertad	E(MC)
Tratamiento A	$I - 1$	$\sigma^2 + JKL\sigma_\alpha^2 + KL\sigma_{\alpha\beta}^2 + JL\sigma_{\alpha\gamma}^2 + L\sigma_{\alpha\beta\gamma}^2$
Tratamiento B	$J - 1$	$\sigma^2 + IKL\sigma_\beta^2 + KL\sigma_{\alpha\beta}^2 + IL\sigma_{\beta\gamma}^2 + L\sigma_{\alpha\beta\gamma}^2$
Tratamiento C	$K - 1$	$\sigma^2 + IJL\sigma_\gamma^2 + JL\sigma_{\alpha\gamma}^2 + IL\sigma_{\beta\gamma}^2 + L\sigma_{\alpha\beta\gamma}^2$
Interacción AB	$(I - 1)(J - 1)$	$\sigma^2 + KL\sigma_{\alpha\beta}^2 + L\sigma_{\alpha\beta\gamma}^2$
Interacción AC	$(I - 1)(K - 1)$	$\sigma^2 + JL\sigma_{\alpha\gamma}^2 + L\sigma_{\alpha\beta\gamma}^2$
Interacción BC	$(J - 1)(K - 1)$	$\sigma^2 + IL\sigma_{\beta\gamma}^2 + L\sigma_{\alpha\beta\gamma}^2$
Interacción ABC	$(I - 1)(J - 1)(K - 1)$	$\sigma^2 + L\sigma_{\alpha\beta\gamma}^2$
Error (VNE)	$IJK(L - 1)$	σ^2

Fuente: Elaboración propia

Como puede observarse, en el cálculo de la esperanza de la media cuadrática o varianza de cada fuente de variación o efecto, interviene la varianza del error, la del propio efecto y la de todas aquellas interacciones donde aparezca dicho efecto.

Teniendo en cuenta estas esperanzas, es inmediato el cálculo de los estadísticos F para contrastar los efectos de las interacciones:

$$F_{\alpha\beta\gamma} = \frac{\hat{S}_{\alpha\beta\gamma}^2}{\hat{S}_R^2} \quad F_{\alpha\beta} = \frac{\hat{S}_{\alpha\beta}^2}{\hat{S}_{\alpha\beta\gamma}^2} \quad F_{\alpha\gamma} = \frac{\hat{S}_{\alpha\gamma}^2}{\hat{S}_{\alpha\beta\gamma}^2} \quad F_{\beta\gamma} = \frac{\hat{S}_{\beta\gamma}^2}{\hat{S}_{\alpha\beta\gamma}^2} \quad [98]$$

Para los efectos principales (A, B, C) necesitamos estimar una media cuadrática para el denominador de la prueba. Suponiendo que estamos evaluando el efecto A, se pueden considerar.

$$F_\alpha = \frac{\hat{S}_\alpha^2}{\hat{S}_{\alpha\beta}^2 + \hat{S}_{\alpha\gamma}^2 - \hat{S}_{\alpha\beta\gamma}^2} \quad [99]$$

$$F'_{\alpha} = \frac{\hat{S}_\alpha^2 + \hat{S}_{\alpha\beta\gamma}^2}{\hat{S}_{\alpha\beta}^2 + \hat{S}_{\alpha\gamma}^2} \quad [100]$$

Los grados de libertad de las sumas de las medias cuadráticas (denominador de F_α y numerador y denominador de F'_{α}) se calculan en base al procedimiento de Satterthwaite.

Este procedimiento establece que dada una función lineal M definida como:

$$M = \alpha_1 \cdot (MC)_1 + \alpha_2 \cdot (MC)_2 + \cdots + \alpha_k \cdot (MC)_k \quad [101]$$

siendo $(MC)_1, (MC)_2, \dots, (MC)_k$ cuadrados medios con grados de libertad v_1, v_2, \dots, v_k .

Los grados de libertad para M son aproximadamente:

$$v = \frac{M^2}{\sum_{i=1}^k \frac{(\alpha_i \cdot (MC)_i)^2}{v_i}} \quad [102]$$

Cuando se utilizan dos o más factores, cada uno con varios niveles, unos de efectos fijos y otros de efectos aleatorios, el análisis de varianza es mixto.

1.3.4. Modelos anidados o jerárquicos

Cuando ciertos factores se hallan dentro de uno superior, el modelo se denomina ANOVA anidado o jerárquico. Los niveles del factor anidado, pongamos B, son similares pero no idénticos para los diferentes niveles del factor principal A. Se dice que B está anidado en A. Generalmente los factores que están anidados son aleatorios.

Como ejemplo, supóngase que se desea evaluar y comparar la atención recibida por los alumnos de enseñanza no universitaria de Madrid y Cataluña, seleccionando 3 colegios en cada caso y tomando datos de 5 aulas. La variable comunidad autónoma será un factor fijo, mientras que los colegios y las aulas serán factores aleatorios.

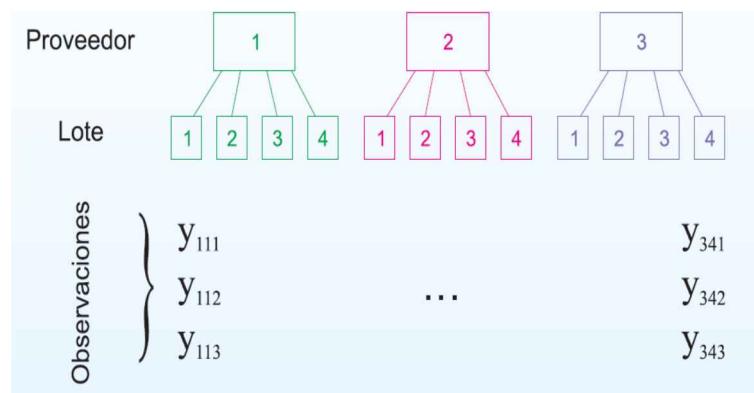
Además, el factor colegio estaría anidado dentro del factor comunidad autónoma y el factor aula dentro del factor colegio.

1.3.4.1. *Modelo de un efecto principal y otro anidado*

Como ejemplo, supongamos una compañía que compra su materia prima a tres proveedores. La compañía desea determinar si la calidad de la materia prima es la misma en cada proveedor.

Para ello, se seleccionan 4 lotes del producto de cada proveedor, y dentro de cada uno se realizan 3 mediciones. El esquema del diseño sería el siguiente:

FIGURA 11. DISEÑO DE UN EFECTO PRINCIPAL (PROVEEDOR) Y OTRO ANIDADO (LOTE)



Fuente: Elaboración propia

En este caso, el lote 1 del proveedor 1 no tiene nada que ver con el lote 1 de los otros proveedores, es solamente una etiqueta.

Este diseño está anidado en 2 etapas: lote anidado en proveedor y observación anidada en lote. El factor proveedor sería un efecto fijo, mientras que el factor lote un efecto aleatorio.

Modelo

El modelo estadístico para los diseños anidados de dos etapas es:

$$y_{ijk} = \mu + \alpha_i + \beta_{j(i)} + u_{ijk} \quad [103]$$

Las hipótesis nulas del modelo, serían las siguientes.

Efectos fijos

$$H_{01}: \alpha_i = 0; \forall i \quad [104]$$

$$H_{02}: \beta_j = 0; \forall j \quad [105]$$

Efectos aleatorios,

$$H_{01}: \sigma_\alpha^2 = 0 \quad [106]$$

$$H_{02}: \sigma_\beta^2 = 0 \quad [107]$$

$$\sigma_y^2 = \sigma_\alpha^2 + \sigma_\beta^2 + \sigma^2 \quad [108]$$

Los estimadores de los parámetros del modelo se calculan del siguiente modo:

$$\hat{\mu} = \bar{y}_{...} \quad \hat{\alpha}_i = \bar{y}_{i...} - \bar{y}_{...} \quad \hat{\beta}_{j(i)} = \bar{y}_{ij.} - \bar{y}_{i..} \quad e_{ijk} = y_{ijk} - \bar{y}_{ij.} \quad [109]$$

y la descomposición de la variabilidad quedaría:

$$VE(\alpha) = JK \sum_{i=1}^I \hat{\alpha}_i^2 \quad VE(\beta(\alpha)) = K \sum_{i=1}^I \sum_{j=1}^J \hat{\beta}_{j(i)}^2 \quad [110]$$

$$VNE = \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^K (y_{ijk} - \bar{y}_{ij.})^2 = \sum_{i=1}^I \sum_{j=1}^J \sum_{k=1}^K e_{ijk}^2 \quad [111]$$

$$VT = VE(\alpha) + VE(\beta(\alpha)) + VNE \quad [112]$$

quedando la tabla ANOVA de la siguiente manera:

TABLA 10. TABLA ANOVA

Fuente de variación	Suma de cuadrados	Grados de libertad	Cuadrado medio (MC)
α	$VE(\alpha)$	$I - 1$	$\hat{S}_\alpha^2 = \frac{VE(\alpha)}{I - 1}$
$\beta(\alpha)$	$VE(\beta(\alpha))$	$I(J - 1)$	$\hat{S}_{\beta(\alpha)}^2 = \frac{VE(\beta)}{I(J - 1)}$
Error (VNE)	VNE	$IJ(K - 1)$	$\hat{S}_R^2 = \frac{VNE}{IJ(K - 1)}$
Total	$VT = VE + VNE$	$n - 1$	$\hat{S}_y^2 = \frac{VT}{n - 1}$

Fuente: Elaboración propia

Según se ha visto, las esperanzas de las medias cuadráticas varían dependiendo de si los efectos son fijos o aleatorios, lo cual condiciona la construcción de las pruebas F. La siguiente tabla muestra las esperanzas y las pruebas F según el efecto sea fijo o aleatorio.

TABLA 11. ESPERANZAS Y ESTADÍSTICOS F

Fuente de variación	E(MC) efecto fijo	Estadístico F	E(MC) efecto aleatorio	Estadístico F
α	$\sigma^2 + \frac{JK}{I-1} \sum \alpha_i^2$	$F_\alpha = \frac{\hat{S}_\alpha^2}{\hat{S}_R^2}$	$\sigma^2 + JK\sigma_\alpha^2 + K\sigma_{\beta(\alpha)}^2$	$F_\alpha = \frac{\hat{S}_\alpha^2}{\hat{S}_{\beta(\alpha)}^2}$
$\beta(\alpha)$	$\sigma^2 + \frac{K}{I(J-1)} \sum \beta_{j(i)}^2$	$F_{\beta(\alpha)} = \frac{\hat{S}_{\beta(\alpha)}^2}{\hat{S}_R^2}$	$\sigma^2 + K\sigma_{\beta(\alpha)}^2$	$F_{\beta(\alpha)} = \frac{\hat{S}_{\beta(\alpha)}^2}{\hat{S}_R^2}$
Error (VNE)	σ^2		σ^2	

Fuente: Elaboración propia

Las configuraciones posibles serían en este caso:

- A fijo y B fijo.
- A fijo y B aleatorio.
- A aleatorio y B aleatorio.

1.3.4.2. Diseños partidos: modelo split-plot y modelo split-split-plot

Modelo Split-Plot

Este diseño se da cuando se tienen dos tipos de unidades experimentales, una para el primer factor y otra para el segundo.

- Factor A: Main Plot o Whole Plot.
- Factor B: Subplot.

Como ejemplo, supóngase que se desea comparar el rendimiento obtenido en la cosecha de un determinado producto agrario teniendo en cuenta el sistema de regadío y el tipo de fertilizante utilizado. Las tomas se realizan en 4 parcelas, en las cuales se sortea el sistema de regadío.

- Factor A: Sistema de regadío (2 niveles).
- Factor B: Tipo de fertilizante utilizado (4 niveles).
- Factor P: Parcela (4 niveles).

Los factores A y B son efectos fijos, mientras que el factor P es aleatorio, dado que las 4 parcelas utilizadas pueden considerarse una muestra aleatoria de las infinitas parcelas que pudieran utilizarse plantando este producto.

La variabilidad asociada a la parcela se desagrega en función de si ésta corresponde al Whole Plot o al Subplot.

- Whole Plot. Como puede verse en la Figura 12, el factor P está anidado en A, es decir, en cada parcela solo puede haber un sistema de regadío.
- Subplot. El factor P se cruza con el factor B, dado que en todas las parcelas se utilizan los mismos 4 tipos de fertilizantes.

FIGURA 12. DISEÑO SPLIT-PLOT. SISTEMA DE REGADÍO Y TIPO DE FERTILIZANTE

		FACTOR A			
		A ₁		A ₂	
		P ₁	P ₂	P ₃	P ₄
FACTOR B	B ₁	B ₂		B ₂	B ₄
	B ₃	B ₁		B ₃	B ₁
	B ₄	B ₃		B ₁	B ₂
	B ₂	B ₄		B ₄	B ₃

Fuente: Elaboración propia

La formulación del modelo sería, por tanto, la siguiente:

$$y_{ijk} = [\mu + \alpha_i + \pi_{j(i)}] + [\beta_k + \alpha\beta_{ik} + \pi'_{k(j(i))}] = WholePlot + Subplot \quad [113]$$

Estimando los parámetros de la forma habitual:

$$\begin{aligned} \hat{\mu} &= \bar{y}_{...} & \hat{\alpha}_i &= \bar{y}_{i..} - \bar{y}_{...} & \hat{\pi}_{j(i)} &= \bar{y}_{ij.} - \bar{y}_{...} & \hat{\beta}_k &= \bar{y}_{..k} - \bar{y}_{...} \\ \hat{\alpha}\beta_{ik} &= \bar{y}_{i.k} - \bar{y}_{i..} - \bar{y}_{..k} + \bar{y}_{...} & \hat{\pi}'_{k(j(i))} &= y_{ijk} - \bar{y}_{ij.} - \bar{y}_{i..} + \bar{y}_{i..} & & & & [114] \end{aligned}$$

Denotando por σ_W^2 a la variabilidad asociada al Whole Plot ($\pi_{j(i)}$), σ_S^2 a la asociada al Subplot ($\pi'_{k(j(i))}$) y Φ_A , Φ_B , Φ_{AB} las medias cuadráticas de los efectos asociados a los factores A, B y su interacción, las esperanzas de las medias cuadráticas asociadas a las distintas fuentes de variación quedarían del siguiente modo.

TABLA 12. ESPERANZAS Y GRADOS DE LIBERTAD

Fuente de variación	E(MC)	Grados de libertad
Whole Plot		
α_i	$JK\Phi_A + K\sigma_W^2$	$I - 1$
$\pi_{j(i)}$	$K\sigma_W^2$	$I(J - 1)$
Subplot		
β_k	$IJ\Phi_B + \sigma_S^2$	$K - 1$
$\alpha\beta_{ik}$	$J\Phi_{AB} + \sigma_S^2$	$(I - 1)(K - 1)$
$\pi'_{kj(i)}$	σ_S^2	$I(J - 1)(K - 1)$

Fuente: Elaboración propia

Si tenemos un modelo con H réplicas, podemos estimar la interacción $\pi\beta_{j(i)k}$ obteniendo las siguientes esperanzas:

TABLA 13. MODELO H RÉPLICAS. ESPERANZAS Y GRADOS DE LIBERTAD

Fuente de variación	E(MC)	Grados de libertad
Whole Plot		
α_i	$JKH\Phi_A + KH\sigma_W^2 + \sigma_S^2$	$I - 1$
$\pi_{j(i)}$	$KH\sigma_W^2 + \sigma_S^2$	$I(J - 1)$
Subplot		
β_k	$IJH\Phi_B + H\sigma_{\pi\beta}^2 + \sigma_S^2$	$K - 1$
$\alpha\beta_{ik}$	$JH\Phi_{AB} + H\sigma_{\pi\beta}^2 + \sigma_S^2$	$(I - 1)(K - 1)$
$\pi\beta_{j(i)k}$	$H\sigma_{\pi\beta}^2 + \sigma_S^2$	$I(J - 1)(K - 1)$
$\pi'_{kj(i)}$	σ_S^2	$I(J - 1)(K - 1)$

Fuente: Elaboración propia

El cálculo de estas esperanzas puede hacerse mediante una regla nemotécnica fácilmente generalizable a cualquier otro modelo.

Modelo Split-Split-Plot

Supóngase ahora que se quiere contrastar adicionalmente el rendimiento de la variedad actual frente a otras dos variedades nuevas.

En este caso, tendremos tres factores fijos y un factor aleatorio. Al igual que en el caso anterior, el factor P (parcela) aparece anidado al factor A (sistema de regadío). Por otra parte, el factor C (variedad de producto) aparece cruzado con el B (tipo de fertilizante) y con el P, es decir, dentro de cada parcela y para cada tipo de fertilizante se plantan, asignándolas al azar, las tres variedades del producto.

Gráficamente, el esquema del modelo es el siguiente.

FIGURA 13. DISEÑO SPLIT-SPLIT-PLOT

A ₁			A ₂		
P ₁		P ₂	P ₃		P ₄
B ₁ C ₁ C ₃	C ₂	B ₂ C ₃ C ₂	B ₁ C ₁ C ₃	B ₂ C ₁ C ₂	B ₁ C ₃ C ₂
B ₃ C ₂ C ₃	C ₁	B ₁ C ₃ C ₂	B ₃ C ₂ C ₁	B ₁ C ₁ C ₃	B ₃ C ₁ C ₂
B ₄ C ₃ C ₁	C ₂	B ₃ C ₁ C ₂	B ₄ C ₃ C ₁	B ₃ C ₁ C ₂	B ₄ C ₃ C ₁
B ₂ C ₁ C ₂	C ₃	B ₄ C ₂ C ₁	B ₂ C ₁ C ₃	B ₄ C ₁ C ₂	B ₂ C ₃ C ₁

Fuente: Elaboración propia

Y la ecuación del modelo resultante es:

$$y_{ijk} = [\mu + \alpha_i + \pi_{j(i)}] + [\beta_k + \gamma_h + \alpha\beta_{ik} + \alpha\gamma_{ih} + \beta\gamma_{kh} + \alpha\beta\gamma_{ikh} + \pi'_{khj(i)}] = WholePlot + Subplot [115]$$

1.3.5. Modelos ANCOVA

Los modelos ANCOVA son modelos ANOVA que incluyen ademas de las variables cualitativas o categóricas, variables cuantitativas entre las variables dependientes.

En determinadas ocasiones, en lugar de disponer de un factor bloque se tiene en su lugar una variable cuantitativa. Esta variable, si no está controlada por el investigador, se ha de incluir en el modelo con el fin de eliminar el efecto que pudiera tener sobre la estimación de los niveles del factor principal. Las variables explicativas cuantitativas se denominan *covariables*.

Partiendo de la especificación de un modelo ANOVA de un factor con dos niveles, al introducir la variable explicativa cuantitativa, X_i , tendríamos:

$$Y_i = \mu_1 D_{1i} + \mu_2 D_{2i} + \beta_1 X_i + u_i [116]$$

o sus equivalentes:

$$Y_i = \beta_0 + \mu_1 D_{1i} + \beta_1 X_i + u_i [117]$$

$$Y_i = \beta_0 + \mu_2 D_{2i} + \beta_1 X_i + u_i [118]$$

Para el análisis del comportamiento de cada grupo respecto a la pendiente, habría que especificar los siguientes modelos ANCOVA:

$$Y_i = \delta_1 D_{1i} X_i + \delta_2 D_{2i} X_i + u_i [119]$$

$$Y_i = \beta_0 + \beta_1 X_i + \delta_1 D_{1i} X_i + u_i [120]$$

$$Y_i = \beta_0 + \beta_1 X_i + \delta_2 D_{2i} X_i + u_i [121]$$

En los modelos ANOVA y ANCOVA se pueden incluir distintas variables categóricas. Consideremos una categórica E con tres categorías (1,2,3):

- E_{1i} que toma valor uno en los individuos pertenecientes a la categoría 1, y cero en los restantes.

- E_{2i} que toma valor uno en los individuos pertenecientes a la categoría 2, y cero en los restantes.
- E_{3i} que toma valor uno en los individuos pertenecientes a la categoría 3, y cero en los restantes.

Si bien a la hora de especificar el modelo hay que evitar los problemas de multicolinealidad entre todas las variables dummy incluidas y el término constante.

Una forma de evitar los problemas es no incluir alguna de las categorías en forma de variable dummy, y dejar que la constante recoja el efecto de la categoría no incluida.

Una especificación posible de este modelo ANCOVA sería entonces:

$$Y_i = \beta_0 + \beta_1 X_i + \mu_2 D_{2i} + \alpha_1 E_{1i} + \alpha_2 E_{2i} + \alpha_3 E_{3i} + u_i \quad [122]$$

Modelo

Desarrollamos a continuación el modelo más simple de ANCOVA, es decir, un factor y una covariante, que se supone relacionada linealmente con la variable dependiente:

$$y_{ij} = \mu + \alpha_i + \beta(x_{ij} - \bar{x}_{..}) + u_{ij} \quad i = 1, \dots, L; j = 1, \dots, n_i \quad [123]$$

siendo β el coeficiente de la regresión lineal entre X e Y . Las medias estimadas para los tratamientos bajo este modelo serían entonces:

$$\mu_i = \mu + \alpha_i + \beta(\bar{x}_i - \bar{x}_{..}) \quad [124]$$

Construyendo dos rectas de regresión con la misma pendiente, β , pero con distinta ordenada en el origen $\mu + \alpha_i$.

La diferencia con la media global μ se descompone en dos elementos, el efecto del factor α_i y el de la covariante $\beta(\bar{x}_i - \bar{x}_{..})$

La hipótesis nula de este modelo será por tanto:

$$H_0: \beta = 0; \quad \alpha_i = 0 \quad \forall i \quad [125]$$

Los parámetros se estiman:

$$\hat{\alpha}_i = \bar{y}_{i.} - \bar{y}_{..} \quad \hat{\beta} = \frac{s_{xy}}{s_x^2} \quad [126]$$

Y la descomposición de la variabilidad VT será:

$$VT = \sum_{i=1}^L \sum_{j=1}^{n_i} (y_{ij} - \bar{y}_{..})^2 = \sum_{i=1}^L n_i (\bar{y}_{i.} - \bar{y}_{..})^2 + \sum_{i=1}^L \sum_{j=1}^{n_i} [\hat{\beta}(x_{ij} - \bar{x}_{..})]^2 + VNE \quad [127]$$

Luego la variabilidad explicada por el factor será:

$$VE_A = \sum_{i=1}^L n_i (\bar{y}_{i.} - \bar{y}_{..})^2 = \sum_{i=1}^L n_i \hat{\alpha}_i^2 \quad [128]$$

La variabilidad explicada por la covariante será entonces:

$$VE_{XxY} = \hat{\beta}^2 \sum_{i=1}^L \sum_{j=1}^{n_i} (x_{ij} - \bar{x}_{..})^2 = \frac{[\sum_{i=1}^L \sum_{j=1}^{n_i} (x_{ij} - \bar{x}_{..})(y_{ij} - \bar{y}_{..})]^2}{\sum_{i=1}^L \sum_{j=1}^{n_i} (x_{ij} - \bar{x}_{..})^2} \quad [129]$$

y la variabilidad no explicada:

$$VNE = \sum_{i=1}^L \sum_{j=1}^{n_i} (y_{ij} - \bar{y}_i + \hat{\beta}(x_{ij} - \bar{x}_{..}))^2 \quad [130]$$

donde como vemos, puede descomponerse en aquella proveniente del factor y la asociada a la covariable.

Tabla ANOVA

La tabla ANOVA quedaría como sigue:

TABLA 14. TABLA ANOVA. MODELO CON UN FACTOR Y UNA COVARIABLE

Fuente de variación	Suma de cuadrados	Grados de libertad	Cuadrado medio	Estadístico F
Factor A	VE_A	$L - 1$	$MCF_A = \frac{VE_A}{L - 1}$	$F = \frac{MCF_A}{MCR}$
Covariable	VE_{XXY}	1	$MCF_{XXY} = \frac{VE_{XXY}}{VE_X}$	$F = \frac{MCF_{XXY}}{MCR}$
Error o residual	VNE	$n - L - 1$	$MCR = \frac{VNE}{n - L - 1}$	
Total	VT	$n - 1$	$MCT = \frac{VT}{n - 1}$	

Fuente: Elaboración propia

Podemos tener más de un factor y/o covariable, e incluso plantear efecto interacción entre un factor y una covariable, obteniendo en este caso dos rectas con distinta ordenada en el origen y distinta pendiente. Hay que tener en cuenta también las hipótesis del modelo sobre los residuos, la selección de variables, etc.

Tomando como covariable la variable peso (wt), en miles de libras, obtenemos de nuevo como resultado un contraste no significativo para el efecto del factor tipo de transmisión. Esta corrección es efectiva gracias a la relación existente entre el peso de los vehículos y el consumo de combustible, pues lógicamente a mayor peso, menos millas recorridas por galón de combustible.

```

lnmpg <- log(mtcars$mpg)
amf <- as.factor(mtcars$am)
modelo.lm <- lm(lnmpg~amf+mtcars$wt)
summary(modelo.lm)

Call:
lm(formula = lnmpg ~ amf + mtcars$wt)

Residuals:
    Min      1Q  Median      3Q     Max 
-0.21351 -0.08281  0.00304  0.04962  0.32349 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 3.89834   0.13567  28.734 < 2e-16 ***
amf1        -0.04307   0.06865  -0.627   0.535    
mtcars$wt   -0.28699   0.03501  -8.198 4.87e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1376 on 29 degrees of freedom

```

Multiple R-squared: 0.8003, Adjusted R-squared: 0.7865
 F-statistic: 58.1 on 2 and 29 DF, p-value: 7.186e-11

Otros modelos ANCOVA podrían ser:

```

Call:
lm(formula = mpg ~ 0 + as.factor(am) + wt, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.5295 -2.3619 -0.1317  1.4025  6.8782 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
as.factor(am)0 37.3216    3.0546 12.218 5.84e-13 ***
as.factor(am)1 37.2979    2.0857 17.883 < 2e-16 ***
wt             -5.3528    0.7882 -6.791 1.87e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.098 on 29 degrees of freedom
Multiple R-squared: 0.9802, Adjusted R-squared: 0.9781
F-statistic: 478.1 on 3 and 29 DF, p-value: < 2.2e-16

Call:
lm(formula = mpg ~ am + wt, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-4.5295 -2.3619 -0.1317  1.4025  6.8782 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 37.32155   3.05464 12.218 5.84e-13 ***
am          -0.02362   1.54565 -0.015  0.988    
wt          -5.35281   0.78824 -6.791 1.87e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.098 on 29 degrees of freedom
Multiple R-squared: 0.7528, Adjusted R-squared: 0.7358
F-statistic: 44.17 on 2 and 29 DF, p-value: 1.579e-09

Call:
lm(formula = mpg ~ am + wt + wt * am, data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.6004 -1.5446 -0.5325  0.9012  6.0909 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 31.4161    3.0201 10.402 4.00e-11 ***
am          14.8784    4.2640  3.489  0.00162 **  
wt          -3.7859    0.7856 -4.819 4.55e-05 ***
am:wt       -5.2984    1.4447 -3.667  0.00102 **  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.591 on 28 degrees of freedom
Multiple R-squared: 0.833, Adjusted R-squared: 0.8151
F-statistic: 46.57 on 3 and 28 DF, p-value: 5.209e-11

Call:

```

```

lm(formula = mpg ~ 0 + as.factor(am) + as.factor(gear) + wt,
  data = mtcars)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.5798 -2.4056 -0.3692  1.8198  5.7713 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
as.factor(am)0   35.0955   3.1862  11.015 1.72e-11 ***
as.factor(am)1   35.2838   3.0091  11.726 4.20e-12 ***
as.factor(gear)4   2.0769   1.7343   1.198   0.242    
as.factor(gear)5  -1.0615   2.3845  -0.445   0.660    
wt                 -4.8782   0.7945  -6.140 1.46e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.968 on 27 degrees of freedom
Multiple R-squared:  0.9831, Adjusted R-squared:  0.9799 
F-statistic: 313.4 on 5 and 27 DF,  p-value: < 2.2e-16

```

La función R "lm", presenta siempre una especificación que evita la multicolinealidad perfecta.

Realizar ejercicio libre con base de datos "npk".

```

'data.frame': 24 obs. of 5 variables:
$ block: Factor w/ 6 levels "1","2","3","4",... : 1 1 1 1 2 2 2 2 3 3 ...
$ N    : Factor w/ 2 levels "0","1": 1 2 1 2 2 2 1 1 1 2 ...
$ P    : Factor w/ 2 levels "0","1": 2 2 1 1 1 2 1 2 2 2 ...
$ K    : Factor w/ 2 levels "0","1": 2 1 1 2 1 2 2 1 1 2 ...
$ yield: num 49.5 62.8 46.8 57 59.8 58.5 55.5 56 62.8 55.8 ...

```

1.4. MODELOS CON VARIABLE DEPENDIENTE MULTIVARIANTE: MANOVA Y MANCOVA

1.4.1. Definición del contraste

En la situación en la que existen múltiples variables respuesta, es posible la realización de un test simultáneo utilizando el *Análisis de la Varianza Multivariante (MANOVA)*.

Por ejemplo, podemos llevar a cabo un experimento en el que aplicamos dos tratamientos (A y B) a dos grupos de ratones, y estamos interesados en el peso y la altura de los ratones.

En este caso, el peso y la altura de los ratones son dos variables dependientes, y nuestra hipótesis es que se ven afectadas conjuntamente por la diferencia en el tratamiento. El MANOVA contrasta esta hipótesis.

Si nuestro modelo incluyera variables explicativas cuantitativas (covariables), estaríamos hablando de un *Análisis de la Covarianza Multivariante (MANCOVA)*.

1.4.2. Supuestos para su aplicación

MANOVA puede ser usado en ciertas condiciones:

1. Las variables dependientes deben distribuirse normalmente dentro de los grupos.

La función R mshapiro.test() [en el paquete mvnormtest] se puede utilizar para realizar la prueba de Shapiro-Wilk para la normalidad multivariante. Esto es útil en el caso de MANOVA, que asume normalidad multivariante.

2. Homogeneidad de varianzas en todo el rango de predictores.
3. Linealidad entre todos los pares de variables dependientes, todos los pares de covariables y todos los pares de variable dependiente-covariable en cada celda.

1.4.3. Estadísticos

De cara a determinar la significación del contraste, se dispone de los siguientes estadísticos:

Lambda de Wilks

$$\frac{\det(E)}{\det(E+H)} \quad [131]$$

Traza de Pillai

$$\text{traza}(H(E + H)^{-1}) \quad [132]$$

Traza de Hotelling-Lawley

$$\text{traza}(E^{-1}H) \quad [133]$$

Raiz máxima de Roy

$$\text{máximo autovalor de } E^{-1}H \quad [134]$$

donde la matriz E corresponde a la suma de cuadrados y productos cruzados del modelo y H a la suma de cuadrados y productos cruzados de los residuos.

1.4.4. Interpretación del test

Si el test multivariante global es significativo, se concluye que el correspondiente efecto (tratamiento) es significativo. En este caso, la siguiente cuestión será determinar si el tratamiento afecta solamente en el peso, solamente en la altura, o en ambos. En otras palabras, queremos identificar las variables dependientes específicas que contribuyeron a la significación del efecto global.

Para responder a esta pregunta, podemos construir modelos ANOVA para examinar por separado cada variable dependiente.

1.4.5. Cálculo en R

Para el ejemplo vamos a utilizar el famoso data.frame iris (de Fisher o Anderson), el cual proporciona las medidas en centímetros de las variables longitud y anchura del sépalo y longitud y anchura del pétalo, respectivamente, para 50 flores de cada una de las 3 especies de iris. Las especies son Iris setosa, versicolor y virginica.

Pregunta: Queremos saber si hay alguna diferencia significativa, en la longitud del sépalo y el pétalo, entre las diferentes especies.

La función manova() se puede utilizar de la siguiente manera:

```
# test MANOVA
res.man <- manova(cbind(Sepal.Length, Petal.Length) ~ Species, data = iris)
summary(res.man)
      Df Pillai approx F num Df den Df Pr(>F)
Species     2 0.9885    71.829      4     294 < 2.2e-16 ***
Residuals 147
```

```

---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

# Mira cuál difiere
summary.aov(res.man)
  Response Sepal.Length :
    Df Sum Sq Mean Sq F value    Pr(>F)
Species      2 63.212 31.606 119.26 < 2.2e-16 ***
Residuals   147 38.956  0.265
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

  Response Petal.Length :
    Df Sum Sq Mean Sq F value    Pr(>F)
Species      2 437.10 218.551 1180.2 < 2.2e-16 ***
Residuals   147  27.22  0.185
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

```

De la salida anterior, se puede ver que las dos variables son muy diferentes entre las especies.

La función R `mancova()` presenta la siguiente sintaxis:

```
mancova(data, deps, factors = NULL, covs = NULL, multivar = list("pillai", "wilks", "hotel", "roy"), boxM = FALSE, shapiro = FALSE, qqPlot = FALSE)
```

Argumentos

- `data`: el conjunto de datos a utilizar, como un `data frame`.
- `deps`: una cadena que nombra la variable dependiente, la cual debe ser numérica.
- `factors`: Un vector de cadenas nombrando los factores.
- `covs`: Un vector de cadenas nombrando las covariables.
- `multivar` (uno o más): '`pillai`', '`wilks`', '`hotel`' o '`roy`'; estadísticas multivariantes: Traza de Pillai, Lambda de Wilks, Traza de Hotelling y la Raíz Mayor de Roy, respectivamente.
- `boxM`: `TRUE` o `FALSE` (predeterminado), para el test M de Box.
- `shapiro`: `TRUE` o `FALSE` (predeterminado), para el test de Shapiro-Wilk.
- `qqPlot`: `TRUE` o `FALSE` (predeterminado), para un plot Q-Q de normalidad multivariate.

Valor

Un objeto de resultados que contiene tablas que se pueden a su vez convertir en `data.frame` con `asDF` o `as.data.frame`. Por ejemplo:

- `results$multivar$asDF`
- `as.data.frame(results$multivar)`

Considerando las 4 variables dependientes del `data.frame iris`, se muestra a continuación el uso de la función `mancova()`.

```
library(jmv)
mancova(data = iris, deps = c('Sepal.Length', 'Sepal.Width', 'Petal.Length', 'Petal.Width'), factors = 'Species')
```

MANCOVA**Multivariate Tests**

		value	F	df1	df2	p
Species	Pillai's Trace	1.19	53.5	8	290	< .001
	Wilks' Lambda	0.0234	199	8	288	< .001
	Hotelling's Trace	32.5	581	8	286	< .001
	Roy's Largest Root	32.2	1167	4	145	< .001

Univariate Tests

	Dependent Variable	Sum of Squares	df	Mean Square	F	p
Species	Sepal.Length	63.21	2	31.6061	119.3	< .001
	Sepal.Width	11.34	2	5.6725	49.2	< .001
	Petal.Length	437.10	2	218.5514	1180.2	< .001
	Petal.Width	80.41	2	40.2067	960.0	< .001
Residuals	Sepal.Length	38.96	147	0.2650		
	Sepal.Width	16.96	147	0.1154		
	Petal.Length	27.22	147	0.1852		
	Petal.Width	6.16	147	0.0419		

1.5. ESTIMACIÓN POR MÁXIMA VERO SIMILITUD RESTRINGIDA (REML) EN MODELOS MIXTOS

Uno de los objetivos del análisis de los modelos mixtos es el de estimar las varianzas de cada uno de los efectos aleatorios, así como sus covarianzas, conocidas como componentes de varianza y covarianza.

La estimación de componentes de varianza y covarianza se puede realizar por los métodos de momentos (ANOVA), estimación cuadrática insesgada de mínima varianza (MIVQUE), máxima verosimilitud (ML) y máxima verosimilitud restringida (REML).

El método de análisis de varianza de mínimos cuadrados (ANOVA) para estimar los componentes de varianza es relativamente sencillo y bueno cuando se tienen experimentos balanceados, presentando las siguientes propiedades: son insesgados, de mínima varianza entre todos los estimadores insesgados que son funciones cuadráticas de las observaciones y, bajo normalidad, de mínima varianza entre todos los estimadores insesgados.

En el caso de diseños desbalanceados, a veces este método de estimar las componentes de varianza ofrece estimaciones negativas. Algunos autores dicen que es evidencia de que la componente es cero, aunque otros dicen que puede ser evidencia de que el modelo es incorrecto, perdiendo todas sus propiedades excepto la de insesgamiento.

Para realizar la estimación por el método MIVQUE tienen que suministrarse valores para los componentes de varianza. Los estimadores son entonces funciones de los datos y de los valores previos, y son de varianza mínima solamente cuando los valores previos son iguales a los valores verdaderos de los componentes de varianza.

Los estimadores de los componentes de varianza y covarianza obtenidos por los métodos de ML y REML tienen propiedades asintóticas que los hacen preferibles sobre los estimadores obtenidos con otros métodos. Cuando el tamaño de muestra es considerablemente grande son:

- Consistentes.
- Normalmente asintóticos.
- Eficientes.
- No requieren de conjuntos de datos balanceados para mantener estas propiedades.
- La matriz asintótica de varianzas y covarianzas de los estimadores es conocida, lo que permite el establecimiento de intervalos de confianza y pruebas de hipótesis acerca de los componentes de varianza y covarianza.

Para tamaños de muestra “pequeños” los estimadores de máxima verosimilitud son generalmente sesgados, puesto que al estimar los efectos aleatorios no se toma en cuenta la pérdida en grados de libertad que resulta de la estimación de los efectos fijos.

Para solucionar este problema, el método REML utiliza combinaciones lineales de los elementos del vector de datos, de tal manera que esas combinaciones no contienen efectos fijos, generando una función de verosimilitud que no depende de los efectos fijos. Posteriormente la función de verosimilitud restringida se maximiza con respecto a cada uno de los componentes de varianza y covarianza. Los valores que maximizan la función son los respectivos estimadores REML de los componentes de varianza y covarianza.

Generalmente, las ecuaciones generadas por los métodos basados en máxima verosimilitud no pueden ser resueltas analíticamente para los diferentes componentes de varianza, por lo que es necesario obtener las soluciones por medio de métodos iterativos altamente demandantes de recursos computacionales. Con el actual desarrollo de la tecnología y de nuevos algoritmos computacionales, esto ya no es un impedimento para realizar dicha estimación.

Respecto al análisis multivariante, la estimación de componentes de covarianza entre múltiples características se ha realizado tradicionalmente a través del análisis de varianza y covarianza multivariante (modelos MANOVA Y MANCOVA). Sin embargo, los estimadores REML de covarianza entre características tienen las mismas propiedades asintóticas que los estimadores de componentes de varianza basados en máxima verosimilitud.

1.6. AJUSTE DE MODELOS MIXTOS CON R

En R hay dos paquetes que nos van a permitir ajustar modelos mixtos.

El primero que apareció fue el paquete *nlme()*. El código de este paquete se escribió para que fuera compatible con S y S-Plus (las versiones comerciales de R).

El otro paquete, que apareció posteriormente, es el *lme4*. El código de este paquete no es compatible con S y S-Plus y la sintaxis cambia ligeramente con respecto al paquete nlme.

1.6.1. Función *lme()* del paquete *nlme*

La especificación de los efectos fijos y aleatorios del modelo en la función *lme()* se realiza con dos argumentos distintos, *fixed* y *random* respectivamente.

En el caso de que no haya efectos fijos en el modelo (todos los efectos son aleatorios), la componente fija del modelo estimaría solo la constante o intercept:

```
fixed = y ~ 1
```

El argumento *fixed* no es totalmente necesario y se puede omitir en el caso de que no haya efectos fijos. En este caso, basta con especificar una fórmula como en el caso de la función

`lm()` y R entiende que todos los factores explicativos son aleatorios. La formulación entonces sería así:

```
y ~ a + b + c
```

donde a , b y c serían factores aleatorios. En el caso de que haya factores fijos y aleatorios, ambos componentes deben ser definidos de manera individual. El componente aleatorio en este caso se especificaría de la siguiente forma:

```
random = ~ 1|a/b/c
```

Un detalle importante es que el nombre de la variable respuesta y no está repetido en la fórmula de los efectos aleatorios: en su lugar se deja un espacio en blanco a la izquierda del símbolo \sim .

En la mayoría de los modelos mixtos se asume que los efectos aleatorios tienen una media de cero y que lo que interesa cuantificar es la variación en la constante (esto es lo que significa el 1) causada por las diferencias entre los niveles del factor de los efectos aleatorios.

Después de la constante viene una barra vertical $|$ que significa "dada la siguiente distribución de las variables aleatorias". En este ejemplo concreto hay tres efectos aleatorios con c anidado dentro de b , que a su vez está anidado dentro de a . Los factores están separados por la barra oblicua $/$ y las variables se enumeran de izquierda a derecha en orden decreciente siguiendo una jerarquía espacial o temporal. La formulación completa del modelo utilizando la función `lme()` sería:

```
lme(fixed = y ~ 1, random = ~ 1|a/b/c)
```

1.6.2. Función `lmer()` del paquete `lme4`

En la función `lmer()` la fórmula se especifica en un único argumento incluyendo ambos tipos de efectos. Los efectos fijos se especifican en primer lugar a la derecha del símbolo \sim en la forma habitual. A continuación, se especifican los efectos aleatorios precedidos por un $+$ y entre paréntesis. R identifica los efectos aleatorios a través de la barra vertical $|$.

La especificación de factores anidados se haría con $:$ en vez de con la barra $/$. Al especificar la anidación de unos factores dentro de otros, no se está asumiendo que todos los factores incluidos en la fórmula son efectos aleatorios, cosa que sí se asume en el caso de la función `lme()`.

Para el ejemplo anterior, el modelo quedaría especificado de la siguiente forma:

```
lmer(y ~ 1 + (1|a) + (1|a:b) + (1|a:b:c))
```

1.6.3. Ejemplos de modelos con R

1.6.3.1. Diseños con factores aleatorios y covariante

El archivo `RIKZ.txt`, contiene datos recogidos por el Netherlands Institute for Coastal and Marine Management/RIKZ en verano de 2002, correspondientes a una muestra de 9 áreas intermareales (en realidad playas, por eso se denominan como Beach en el archivo) situadas a lo largo de la costa norte del país.

En cada área se fijaron 5 estaciones a distintas alturas sobre el nivel medio de marea (la variable `NAP` recoge el valor de la altura), que permitieron medir la macrofauna y varias

variables abióticas. A partir de estos datos se calculó un índice de riqueza biológica (número de especies diferentes) en cada estación.

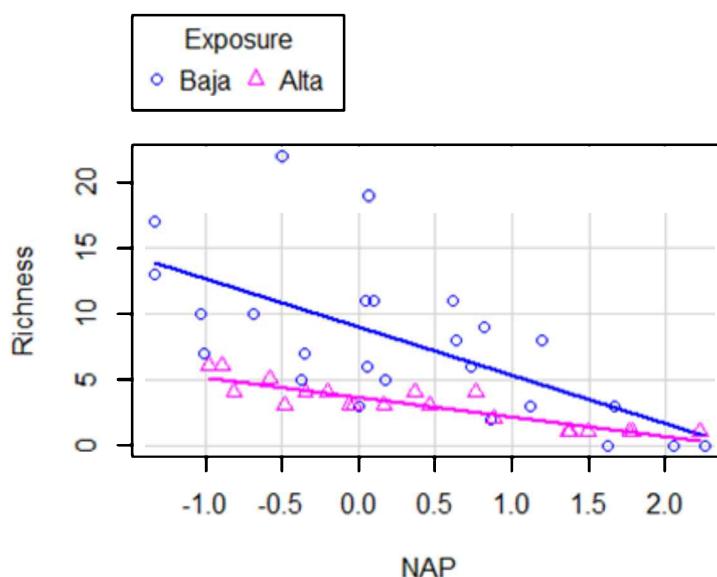
Asimismo, también para cada estación se calculó un índice de exposición que combinaba la acción del oleaje, la longitud de la zona de rompientes, la pendiente de la misma, el tamaño del grano y la profundidad de la capa anaeróbica. Este índice toma solo dos valores, alto y bajo.

La cuestión de fondo consiste en decidir si existe asociación entre la riqueza biológica, la exposición y la altura sobre el nivel medio de la marea. Un primer modelo para el estudio de esta relación es el siguiente:

En primer lugar, leemos los datos y realizamos una exploración gráfica inicial.

```
library(car)
RIKZ = read.table("RIKZ.txt", header = T)
fBeach = factor(RIKZ$Beach)
fExposure <- factor(ifelse(RIKZ$Exposure == 11, 2, 1))
levels(fExposure) <- c("Baja", "Alta")
datos <- cbind.data.frame(Sample = RIKZ$Sample, Richness = RIKZ$Richness, Exposure =
fExposure, NAP = RIKZ$NAP, Beach = fBeach)

scatterplot(Richness ~ NAP | Exposure, data = datos, regLine = TRUE, smooth = F)
```



En esta gráfica observamos que tanto cuando la exposición es alta como cuando es baja, la riqueza biológica disminuye a medida que aumenta la altura de la estación NAP con respecto al nivel medio de marea; asimismo el gráfico parece indicar que la tasa con que se produce dicha disminución es mayor a niveles de exposición bajos (pendiente negativa más acusada).

Si denotamos como:

- R_{ij} a la Riqueza biológica en la estación j de la playa i ,
- A_{ij} el correspondiente valor de la Altura sobre el nivel medio de marea,
- E_i el nivel de Exposición en la playa i , y
- u_{ij} el término residual no explicado con distribución $N(0, \sigma_e^2)$.

Podemos contemplar 3 posibilidades:

1. No hay diferencias en la relación Riqueza-NAP según la exposición.

$$R_{ij} = \beta_0 + \beta_1 * A_{ij} + u_{ij} \quad [135]$$

2. La relación entre Riqueza y NAP es lineal, con la misma pendiente cualquiera que sea la exposición, y con ordenada dependiente del nivel de la misma.

$$R_{ij} = \beta_0 + \beta_1 * A_{ij} + \beta_2 * E_i + u_{ij} \quad [136]$$

3. La relación Riqueza-NAP es lineal y diferente tanto en pendiente como en ordenada en el origen según el nivel de exposición.

$$R_{ij} = \beta_0 + \beta_1 * A_{ij} + \beta_2 * E_i + \beta_3 * A_{ij} * E_i + u_{ij} \quad [137]$$

El código R que almacena estos 3 modelos sería el siguiente:

```
lm1 = lm(Richness ~ NAP, data = datos)
lm2 = lm(Richness ~ NAP + Exposure, data = datos)
lm3 = lm(Richness ~ NAP * Exposure, data = datos)
```

Si comparamos los modelos 1 y 2, estaríamos evaluando la significación de la ordenada en el origen en ambos niveles de exposición.

```
# Modelo sin diferencias en la relación Riqueza-NAP según la exposición
summary(lm1)
```

```
Call:
lm(formula = Richness ~ NAP, data = datos)

Residuals:
    Min      1Q  Median      3Q      Max 
-5.0675 -2.7607 -0.8029  1.3534 13.8723 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  6.6857    0.6578 10.164 5.25e-13 ***
NAP         -2.8669    0.6307 -4.545 4.42e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 4.16 on 43 degrees of freedom
Multiple R-squared:  0.3245, Adjusted R-squared:  0.3088 
F-statistic: 20.66 on 1 and 43 DF,  p-value: 4.418e-05
```

```
# Modelo con relación lineal entre Riqueza y NAP, con la misma pendiente cualquiera
que sea la exposición, y con ordenada dependiente del nivel de la misma
summary(lm2)
```

```
Call:
lm(formula = Richness ~ NAP + Exposure, data = datos)

Residuals:
    Min      1Q  Median      3Q      Max 
-5.6501 -2.4452 -0.6548  1.6964 11.9823 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  8.6447    0.7201 12.006 3.65e-15 ***
NAP         -2.7296    0.5336 -5.115 7.32e-06 ***
ExposureAlta -4.5152    1.0559 -4.276 0.000107 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 3.513 on 42 degrees of freedom
Multiple R-squared:  0.5294, Adjusted R-squared:  0.507
F-statistic: 23.63 on 2 and 42 DF,  p-value: 1.335e-07
```

```
# Modelo con relación lineal Riqueza-NAP diferente tanto en pendiente como en
ordenada en el origen según el nivel de exposición
summary(lm3)
```

```
Call:
lm(formula = Richness ~ NAP * Exposure, data = datos)

Residuals:
    Min      1Q  Median      3Q     Max 
-5.9289 -1.3930 -0.3001  0.9302 11.2329 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)   8.9216    0.7049 12.657 9.50e-16 ***
NAP          -3.6689    0.6806 -5.391 3.17e-06 ***
ExposureAlta -5.3043    1.0827 -4.899 1.55e-05 ***
NAP:ExposureAlta  2.1771    1.0362  2.101  0.0418 *  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 3.379 on 41 degrees of freedom
Multiple R-squared:  0.5752, Adjusted R-squared:  0.5441
F-statistic: 18.5 on 3 and 41 DF,  p-value: 9.591e-08
```

Dado que todos los efectos son significativos, optaríamos por el modelo 3.

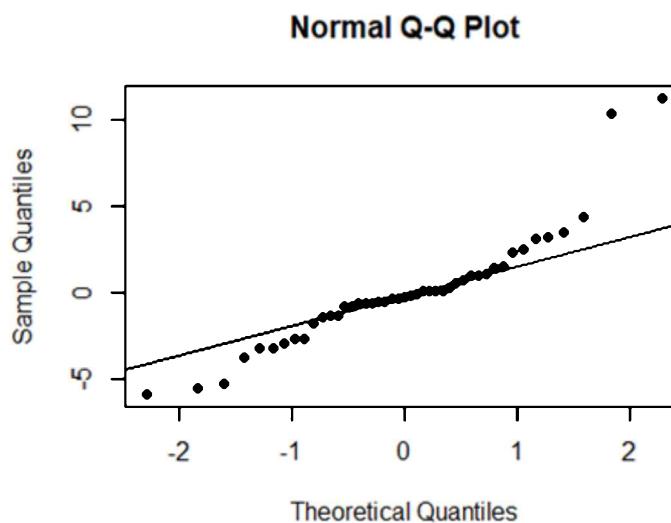
El siguiente paso sería validar el modelo. Si analizamos la normalidad de los residuos.

```
shapiro.test(residuals(lm3))

Shapiro-Wilk normality test

data: residuals(lm3)
W = 0.87878, p-value = 0.0002206

qqnorm(residuals(lm3), pch = 19, col = "gray50")
qqline(residuals(lm3))
```

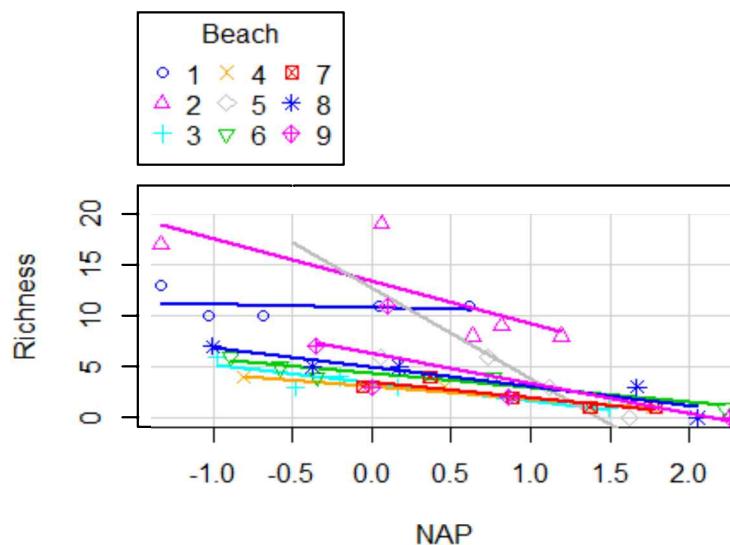


Vemos que no es posible aceptar dicha hipótesis debido a la dispersión irregular de los puntos en torno a la recta cuando el nivel de exposición es bajo. La pregunta natural es

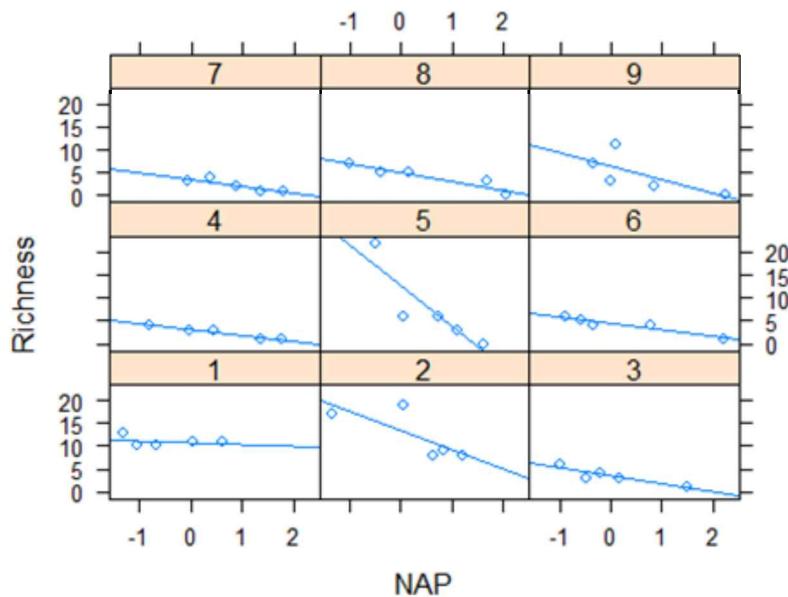
entonces ¿hay alguna otra variable que no hayamos tenido en cuenta que pudiera dar cuenta de esta dispersión tan irregular? Una posible causa de tal dispersión podría ser el efecto de la playa.

Haciendo una nueva comparación considerando la playa como variable de agrupamiento obtenemos:

```
library(car)
library(lattice)
scatterplot(Richness ~ NAP | Beach, data = datos, regLine = TRUE, smooth = F)
  Warning in scatterplot.default(X[, 2], X[, 1], groups = X[, 3], xlab = xlab, :
  number of groups exceeds number of available colors
  colors are recycled
```



```
xyplot(Richness ~ NAP | Beach, data = datos, type = c("p", "r"))
```



Estas figuras nos indican que en la relación Riqueza-NAP existe también un efecto específico de la playa. El comando lmList() nos muestra la regresión Riqueza-NAP para cada playa:

```

library(nlme)
lmRAP = lmList(Richness ~ NAP | fBeach, data = datos)
lmRAP
Call:
Model: Richness ~ NAP | fBeach
Data: datos

Coefficients:
(Intercept)      NAP
1   10.821894 -0.3718279
2   13.345694 -4.1752712
3   3.400702 -1.7553529
4   3.087716 -1.2485766
5   12.782828 -8.9001779
6   4.324634 -1.3885120
7   3.520626 -1.5176126
8   4.951455 -1.8930665
9   6.295053 -2.9675304

Degrees of freedom: 45 total; 27 residual
Residual standard error: 2.478969

```

Tratar de ajustar un modelo a cada una de las playas resultaría excesivamente costoso, pues tendríamos que estimar 3 parámetros (los coeficientes de la regresión más el error estándar) en cada playa, por tanto, un total de 27 parámetros, cuando solo disponemos de 45 observaciones para hacerlo.

Además, las playas fueron elegidas al azar entre todas las de la costa norte de Holanda, por lo que en realidad no tiene demasiado interés determinar qué es lo que pasa particularmente en cada una de ellas.

Modelo de un factor aleatorio

Si evaluamos únicamente el efecto playa, el modelo sería:

$$R_{ij} = \beta_0 + P_i + u_{ij} \quad [138]$$

Los efectos asociados a la playa P_i se suponen con distribución $N(0, \sigma_P^2)$, o en otras palabras, no nos interesa aquí estimar el efecto individual de cada playa, sino la variabilidad en R_i que puede explicarse por el efecto de la playa.

```

mod1fa = lme(Richness ~ 1, random = ~1 | Beach, data = datos)
summary(mod1fa)
Linear mixed-effects model fit by REML
Data: datos
      AIC      BIC      logLik
267.1142 272.4668 -130.5571

Random effects:
Formula: ~1 | Beach
          (Intercept) Residual
StdDev:    3.237112 3.938415

Fixed effects: Richness ~ 1
Value Std.Error DF t-value p-value
(Intercept) 5.688889 1.228419 36 4.631066     0

Standardized Within-Group Residuals:
Min           Q1           Med           Q3           Max
-1.77968689 -0.50704111 -0.09795286  0.25468670  3.80631705

```

Number of Observations: 45
Number of Groups: 9

Según se observa en los resultados:

$$\hat{\beta}_0 = 5.689 \quad \hat{\sigma}_P^2 = 3.237 \quad \hat{\sigma}_e^2 = 3.938$$

Los intervalos de confianza para estos parámetros se obtienen del siguiente modo:

```
intervals(mod1fa)
Approximate 95% confidence intervals

Fixed effects:
      lower     est.     upper
(Intercept) 3.19754 5.688889 8.180238
attr(,"label")
[1] "Fixed effects:"

Random Effects:
  Level: Beach
      lower     est.     upper
sd((Intercept)) 1.70907 3.237112 6.131343

Within-group standard error:
      lower     est.     upper
3.126123 3.938415 4.961773
```

Modelo de ordenada aleatoria

Podemos introducir ahora el efecto de la altura de la estación, NAP, considerando la existencia de una asociación lineal entre esta variable y la riqueza biológica.

Ahora bien, en lugar de estimar una regresión distinta para cada playa podemos optar por modelar esta asociación considerando que la pendiente es la misma para todas las playas, y que la ordenada varía aleatoriamente de una playa a otra. Es decir:

$$R_{ij} = \beta_{0i} + \beta_1 A_{ij} + u_{ij} \quad [139]$$

donde $\beta_{0i} \approx N(\beta_0, \sigma_{\beta_0}^2)$ es la ordenada en la playa i y β_1 es la pendiente común a todas las playas. Transcrito a R, este modelo sería:

```
modOrdA1 = lme(Richness ~ NAP, random = ~1 | Beach, data = datos)
summary(modOrdA1)
```

```
Linear mixed-effects model fit by REML
Data: datos
      AIC      BIC      logLik
247.4802 254.525 -119.7401
```

```
Random effects:
Formula: ~1 | Beach
      (Intercept) Residual
StdDev: 2.944065 3.05977
```

```
Fixed effects: Richness ~ NAP
      Value Std.Error DF t-value p-value
(Intercept) 6.581893 1.0957618 35 6.006682 0
NAP        -2.568400 0.4947246 35 -5.191574 0
Correlation:
  (Intr)
NAP -0.157
```

Standardized Within-Group Residuals:

Min	Q1	Med	Q3	Max
-1.4227495	-0.4848006	-0.1576462	0.2518966	3.9793918

```

Number of Observations: 45
Number of Groups: 9
intervals(modOrdAl)
Approximate 95% confidence intervals

```

Fixed effects:

	lower	est.	upper
(Intercept)	4.357378	6.581893	8.806407
NAP	-3.572744	-2.568400	-1.564055
attr(,"label")			
[1]	"Fixed effects:"		

Random Effects:

Level: Beach

	lower	est.	upper
sd((Intercept))	1.616183	2.944065	5.362956

Within-group standard error:

	lower	est.	upper
2.420870	3.059770	3.867284	

El modelo final resultante arroja las siguientes estimaciones:

$$\hat{\beta}_0 = 6.582 \quad \sigma_{\beta_0}^2 = 2.944 \quad \hat{\beta}_1 = -2.568 \quad \hat{\sigma}_e^2 = 3.06$$

donde $\beta_0 \approx N(6.582, 2.944)$.

Modelo con ordenada y pendiente aleatorias

Considerando que tanto la ordenada como la pendiente varían aleatoriamente entre las playas, el modelo sería entonces:

$$R_{ij} = \beta_{0i} + \beta_{1i}A_{ij} + u_{ij} \quad [140]$$

donde $\beta_{0i} \approx N(\beta_0, \sigma_{\beta_0}^2)$ es la ordenada y $\beta_{1i} \approx N(\beta_1, \sigma_{\beta_1}^2)$ la pendiente en la playa i.

El modelo estimado con R sería el siguiente:

```

modPendOrdAl = lme(Richness ~ NAP, random = ~1 + NAP | fBeach, data = datos)
summary(modPendOrdAl)
  Linear mixed-effects model fit by REML
  Data: datos
      AIC      BIC      logLik
  244.3839 254.9511 -116.1919

  Random effects:
  Formula: ~1 + NAP | fBeach
  Structure: General positive-definite, Log-Cholesky parametrization
            StdDev   Corr
  (Intercept) 3.549069 (Intr)
  NAP         1.714956 -0.99
  Residual    2.702824

  Fixed effects: Richness ~ NAP
                  Value Std.Error DF t-value p-value
  (Intercept) 6.588703 1.2647624 35 5.209439 0e+00
  NAP        -2.830027 0.7229382 35 -3.914617 4e-04
  Correlation:
  (Intr)
  NAP -0.819

```

Standardized Within-Group Residuals:

Min	Q1	Med	Q3	Max
-1.8213267	-0.3411044	-0.1674620	0.1921292	3.0397140

Number of Observations: 45**Number of Groups:** 9

Según podemos observar, los resultados arrojan las siguientes estimaciones:

$$\hat{\beta}_0 = 6.589 \quad \sigma_{\beta_0}^2 = 3.549 \rightarrow \beta_0 \approx N(6.589, 3.549)$$

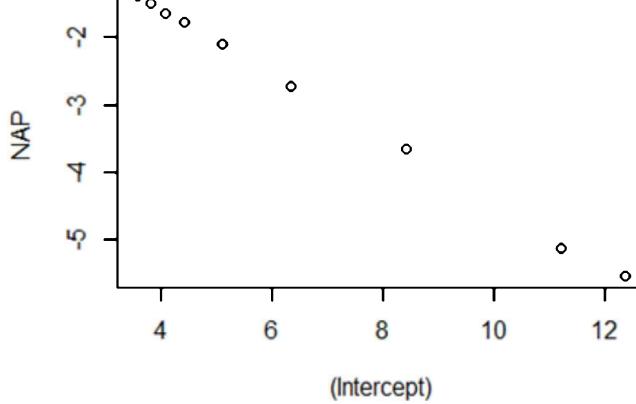
$$\hat{\beta}_1 = -2.830 \quad \sigma_{\beta_1}^2 = 1.715 \rightarrow \beta_1 \approx N(-2.830, 1.715)$$

$$\sigma_e^2 = 2.703$$

Como también puede observarse, la correlación entre las estimaciones de las pendientes y ordenadas en distintas playas vale -0.99. Este valor significa que las pendientes y las ordenadas son muy dependientes entre sí, y que al aumentar la pendiente disminuye la ordenada (lo cual es esperable). Podemos ver estos coeficientes y una representación gráfica de los mismos a través de la función *coef()*.

```
coef(modPendOrdA1)
  (Intercept)      NAP
1   8.421048 -3.656282
2  12.363496 -5.536830
3   3.806642 -1.505716
4   3.562421 -1.385962
5  11.200138 -5.137276
6   4.426276 -1.775683
7   4.082942 -1.644325
8   5.099893 -2.106857
9   6.335470 -2.721310
```

```
plot(as.data.frame(coef(modPendOrdA1)))
```



Por último, se muestran a continuación los intervalos de confianza asociados a las estimaciones de los parámetros.

```
intervals(modPendOrdA1)
  Approximate 95% confidence intervals
```

```
Fixed effects:
  lower      est.      upper
(Intercept) 4.021099 6.588703 9.156307
NAP         -4.297669 -2.830027 -1.362384
```

```

attr("label")
[1] "Fixed effects:"


Random Effects:
Level: fBeach
      lower      est.      upper
sd((Intercept)) 1.9869898 3.5490685 6.339181
sd(NAP)          0.6586315 1.7149559 4.465431
cor((Intercept),NAP) -1.0000000 -0.9901979 1.000000

Within-group standard error:
      lower      est.      upper
1.948475 2.702824 3.749218

```

Selección del mejor modelo

Comenzamos por el modelo más completo posible para los efectos fijos. Elegimos un modelo que contenga como variables explicativas NAP y Exposure, así como sus interacciones:

```

# Modelo de efectos fijos: NAP y Exposure
modelo1 = gls(Richness ~ 1 + NAP * Exposure, data = datos)
modelo1
Generalized least squares fit by REML
Model: Richness ~ 1 + NAP * Exposure
Data: datos
Log-restricted-likelihood: -114.2665

Coefficients:
(Intercept)           NAP       ExposureAlta NAP:ExposureAlta
8.921601            -3.668932        -5.304301         2.177147

Degrees of freedom: 45 total; 41 residual
Residual standard error: 3.37862

```

Procedemos a continuación a buscar la estructura óptima para la componente aleatoria. En principio, en este caso cabe pensar en dos posibilidades:

- considerar como aleatoria la variación entre las ordenadas en cada playa, manteniendo fija la pendiente (modelo2a);
- o considerar que tanto las ordenadas como las pendientes se encuentran sujetas a efectos aleatorios (modelo2b).

```

modelo2a = lme(Richness ~ 1 + NAP * Exposure, random = ~1 | Beach, data = datos)
modelo2b = lme(Richness ~ 1 + NAP * Exposure, random = ~1 + NAP | Beach, data = datos)

```

La función **anova.lme()** permite comparar el ajuste entre dos modelos. Los estadísticos AIC, BIC y logLik serán estudiados en detalle en el apartado correspondiente a la evaluación de modelos, baste aquí decir que si el p-valor obtenido es inferior a 0.05, podremos decir que existen evidencias para suponer una calidad del ajuste distinta. Es habitual tomar como criterio el modelo con menor AIC.

```

# Modelo completo efectos fijos vs modelo con ordenadas aleatorias y pendiente fija
anova.lme(modelo1, modelo2a)
      Model df      AIC      BIC    logLik   Test L.Ratio p-value
modelo1     1 5 238.5329 247.1008 -114.2665
modelo2a    2 6 236.4925 246.7739 -112.2462 1 vs 2 4.040477 0.0444

# Modelo con ordenadas aleatorias y pendiente fija vs modelo con ordenadas y
# pendientes aleatorias
anova.lme(modelo2a, modelo2b)

```

	Model	df	AIC	BIC	logLik	Test	L.Ratio	p-value
modelo2a		1	6	236.4925	246.7739	-112.2462		
modelo2b		2	8	237.1331	250.8416	-110.5665	1 vs 2	3.359399 0.1864

Dado que el AIC del modelo2a (236.4925) es menor que el del modelo1 (238.5329) y el p-valor es significativo (0.0444), podemos concluir que el modelo 2 presenta un mejor ajuste que el modelo 1. Por otra parte, no existen diferencias en el ajuste entre el modelo2a y el modelo2b (p-valor = 0.1864), así que optaríamos en este caso por el modelo2a, por ser más sencillo y presentar un menor valor de AIC.

El modelo finalmente considerado sería:

```
summary(modelo2a)
Linear mixed-effects model fit by REML
  Data: datos
      AIC      BIC      logLik 
  236.4925 246.7739 -112.2462 

Random effects:
 Formula: ~1 | Beach
        (Intercept) Residual
 StdDev:    1.818578 2.942872 

Fixed effects: Richness ~ 1 + NAP * Exposure
                Value Std.Error DF   t-value p-value
(Intercept)     8.861084 1.0208449 34  8.680147 0.0000
NAP            -3.463651 0.6278583 34 -5.516613 0.0000
ExposureAlta   -5.255617 1.5452292  7 -3.401190 0.0114
NAP:ExposureAlta 2.000464 0.9461260 34  2.114374 0.0419

Correlation:
          (Intr) NAP   ExpsrA
NAP       -0.181
ExposureAlta -0.661  0.120
NAP:ExposureAlta  0.120 -0.664 -0.221

Standardized Within-Group Residuals:
      Min        Q1        Med        Q3        Max
-1.48492635 -0.41608772 -0.07703731  0.15206678  3.73131777

Number of Observations: 45
Number of Groups: 9
```

Como vemos, todos los efectos son significativos, pero dado que la interacción presenta un p-valor (0.0419) relativamente alto, vamos a considerar el modelo sin interacción para ver si al incluirla mejoramos el ajuste.

Si queremos comparar los modelos con y sin interacción, al ser ésta una interacción de dos efectos fijos, debemos realizar la estimación por el método de máxima verosimilitud ML. Como podemos comprobar, p-valor = 0.0311, si eliminamos del modelo la interacción la calidad del ajuste disminuiría significativamente, luego esto confirmaría la elección del modelo2a.

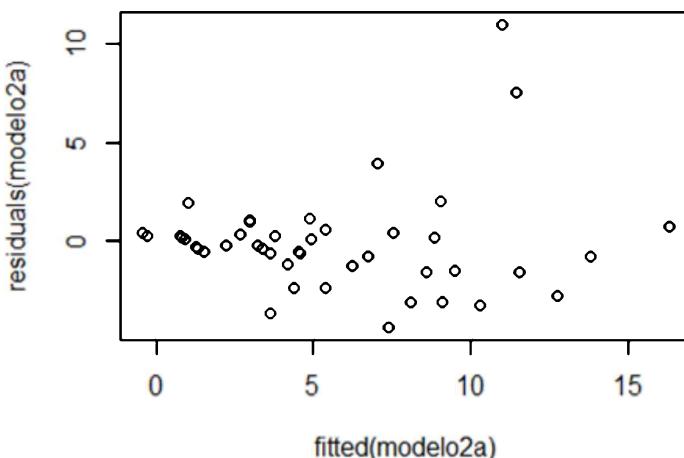
```
modelo3 = lme(Richness ~ 1 + NAP + Exposure, random = ~1 | Beach, data = datos,
method = "ML")
modelo2a.ml = lme(Richness ~ 1 + NAP * Exposure, random = ~1 | Beach, data = datos,
method = "ML")
anova.lme(modelo2a.ml, modelo3)
Model df      AIC      BIC      logLik  Test  L.Ratio p-value
modelo2a.ml     1  6 242.1135 252.9535 -115.0567
modelo3        2  5 244.7589 253.7922 -117.3795 1 vs 2 4.645431 0.0311
```

Por último, hay que señalar que los residuos siguen sin cumplir las hipótesis de normalidad.

```
shapiro.test(residuals(modelo2a))
```

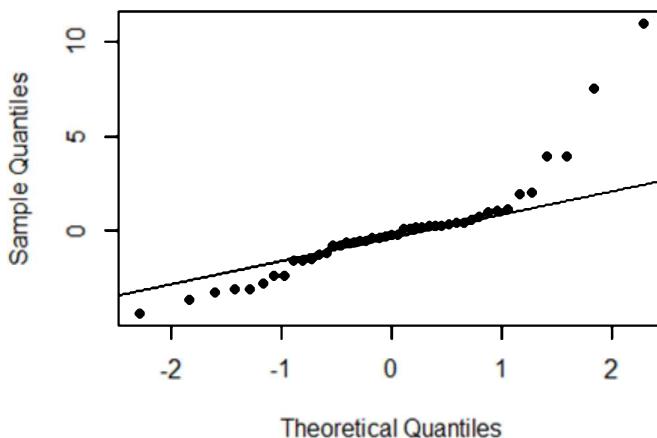
```
Shapiro-Wilk normality test

data: residuals(modelo2a)
W = 0.8239, p-value = 8.328e-06
plot(fitted(modelo2a), residuals(modelo2a))
```



```
qqnorm(residuals(modelo2a), pch = 19, col = "gray50")
qqline(residuals(modelo2a))
```

Normal Q-Q Plot



1.6.3.2. Diseño anidado de dos etapas

Retomando el ejemplo expuesto en la teoría, donde una empresa desea contrastar la calidad de la materia prima suministrada por tres proveedores, seleccionando para ello cuatro lotes de cada proveedor y tomando tres mediciones en cada lote.

Estamos por tanto ante un diseño balanceado con 36 observaciones, las cuales están almacenadas en el fichero "Anidado 2 etapas.xlsx".

```
library(xlsx)
setwd("C:/Trabajo/Libro Metodos/Capítulo 4")
calidad_df <- read.xlsx("Anidado 2 etapas.xlsx", 1)
calidad_df$Proveedor <- as.factor(calidad_df$Proveedor)
calidad_df$Lote <- as.factor(calidad_df$Lote)
str(calidad_df)
```

```
'data.frame': 36 obs. of 3 variables:
 $ Proveedor: Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 1 1 ...
 $ Lote      : Factor w/ 4 levels "1","2","3","4": 1 1 1 2 2 2 3 3 3 4 ...
 $ Calidad   : num 7 5 6 4 3 2 4 6 7 7 ...

# Calidad media por proveedor
aggregate(calidad_df$Calidad, by = list(calidad_df$Proveedor), FUN = mean)
  Group.1      x
1      1 5.583333
2      2 6.333333
3      3 7.166667
```

El modelo a ajustar lo podemos definir como:

$$C_{ijk} = \mu + P_i + L(P)_{j(i)} + u_{ijk} \quad [141]$$

donde C_{ijk} representa la calidad de la unidad k del lote j y el proveedor i, P_i el efecto (fijo) del proveedor i y $L(P)_{j(i)}$ la variabilidad asociada al factor lote (efecto aleatorio y anidado en efecto proveedor).

Se muestra a continuación el código en R para obtener el análisis de la varianza y la configuración del modelo, en este caso utilizando la función *lmer()*.

```
library(lme4)
# Análisis de la varianza
modelo_anidado.aov <- aov(Calidad ~ 1 + Proveedor + Error(Proveedor/Lote), data =
calidad_df)

# Modelo
modelo_anidado = lmer(Calidad ~ 1 + Proveedor + (1 | Proveedor:Lote), data =
calidad_df)
```

Según puede observarse en los resultados del análisis de la varianza, la media cuadrática para el efecto proveedor y el efecto lote dentro de proveedor serían:

$$MC(P) = 7,528$$

$$MC(L(P)) = 7,769$$

$$MC(Error) = 2,639$$

Luego concluimos que no hay efecto significativo del proveedor en la calidad de la materia prima.

$$F_P = \frac{MC(P)}{MC(L(P))} = \frac{7,528}{7,769} = 0,97$$

Ahora bien, la calidad de los lotes de materia prima del mismo proveedor difiere significativamente, por lo tanto, hay que trabajar con los proveedores para que reduzcan su variabilidad de lote a lote.

$$F_{L(P)} = \frac{MC(L(P))}{MC(Error)} = \frac{7,769}{2,639} = 2,94$$

```
summary(modelo_anidado.aov)
```

Error: Proveedor	Df	Sum Sq	Mean Sq
Proveedor	2	15.06	7.528

```
Error: Proveedor:Lote
      Df Sum Sq Mean Sq F value Pr(>F)
Residuals 9 69.92 7.769
```

```
Error: Within
      Df Sum Sq Mean Sq F value Pr(>F)
Residuals 24 63.33 2.639
```

Las estimaciones de los componentes de varianza son:

Componente	Estimación	Porcentaje
Lote(Proveedor)	1,710	39,32
Error	2,639	60,68
Total	4,349	100,00

```
summary(modelo_anidado)
Linear mixed model fit by REML ['lmerMod']
Formula: Calidad ~ 1 + Proveedor + (1 | Proveedor:Lote)
Data: calidad_df

REML criterion at convergence: 142.8

Scaled residuals:
    Min      1Q  Median      3Q     Max 
-1.47513 -0.75010  0.08126  0.70609  1.87201 

Random effects:
 Groups           Name        Variance Std.Dev. 
 Proveedor:Lote (Intercept) 1.710     1.308  
 Residual          2.639     1.624  
 Number of obs: 36, groups: Proveedor:Lote, 12

Fixed effects:
            Estimate Std. Error t value
(Intercept)  5.5833    0.8046   6.939
Proveedor2    0.7500    1.1379   0.659
Proveedor3    1.5833    1.1379   1.391

Correlation of Fixed Effects:
            (Intr) Prvdr2
Proveedor2 -0.707
Proveedor3 -0.707  0.500
```

Los resultados, sin tener en cuenta la estructura del diseño (cada 3 observaciones, se tiene en común que provienen de un mismo lote donde hay diferencias, según el análisis anterior), serían los siguientes:

```
summary(lm(Calidad ~ 1 + Proveedor, data = calidad_df))

Call:
lm(formula = Calidad ~ 1 + Proveedor, data = calidad_df)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.583 -1.396  0.125  1.417  4.417 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  5.5833    0.5801   9.625 4.18e-11 ***
Proveedor2    0.7500    0.8204   0.914   0.3672    
Proveedor3    1.5833    0.8204   1.930   0.0622 .  

```

```

---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.009 on 33 degrees of freedom
Multiple R-squared:  0.1015, Adjusted R-squared:  0.04706
F-statistic: 1.864 on 2 and 33 DF,  p-value: 0.171

```

1.6.3.3. Diseño Split-Plot

Para el ejemplo vamos a utilizar el data.frame *Oats*, que contiene los datos asociados a un experimento de este tipo sobre el rendimiento de la avena.

Se dispone de tres variedades y cuatro niveles de tratamiento manurial. El experimento se organizó en 6 bloques de 3 parcelas principales, cada una dividida en 4 subparcelas. Las variedades se aplicaron a las parcelas principales y los tratamientos manuriales a las subparcelas.

```

oats_df <- Oats

str(oats_df)
Classes 'nfnGroupedData', 'nfGroupedData', 'groupedData' and 'data.frame':  72
obs. of 4 variables:
 $ Block : Ord.factor w/ 6 levels "VI"<"V"<"III"<...: 6 6 6 6 6 6 6 6 ...
 $ Variety: Factor w/ 3 levels "Golden Rain",...: 3 3 3 3 1 1 1 1 2 2 ...
 $ nitro : num  0 0.2 0.4 0.6 0 0.2 0.4 0.6 0 0.2 ...
 $ yield : num  111 130 157 174 117 114 161 141 105 140 ...
 - attr(*, "formula")=Class 'formula' language yield ~ nitro | Block
 ... ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
 - attr(*, "labels")=List of 2
   ..$ y: chr "Yield"
   ..$ x: chr "Nitrogen concentration"
 - attr(*, "units")=List of 2
   ..$ y: chr "(bushels/acre)"
   ..$ x: chr "(cwt/acre)"
 - attr(*, "inner")=Class 'formula' language ~Variety
 ... ..- attr(*, ".Environment")=<environment: R_GlobalEnv>

head(oats_df)
Grouped Data: yield ~ nitro | Block
  Block      Variety nitro yield
1     I        Victory  0.0    111
2     I        Victory  0.2    130
3     I        Victory  0.4    157
4     I        Victory  0.6    174
5     I  Golden Rain  0.0    117
6     I  Golden Rain  0.2    114

```

Por tanto, las parcelas están anidadas en el efecto variedad y el tratamiento cruzado con el efecto bloque. El modelo a ajustar sería:

$$Y_{ijk} = [\mu + V_i + B_{j(i)}] + [N_k + VN_{ik} + B'_{kj(i)}] = WholePlot + Subplot \quad [142]$$

El análisis de la varianza lo obtenemos:

```

oats_df$N <- factor(Oats$nitro)
oats_df$N <- ordered(oats_df$N, levels = sort(levels(oats_df$N)))
oats_df <- cbind.data.frame(B = Oats$Block, V = Oats$Variety, N = oats_df$N, Y = Oats$yield)
oats.aov <- aov(Y ~ N*V + Error(V/B), data = oats_df)
summary(oats.aov)

Error: V
  Df Sum Sq Mean Sq

```

```
V 2 1786 893.2

Error: V:B
      Df Sum Sq Mean Sq F value Pr(>F)
Residuals 15 21889   1459

Error: Within
      Df Sum Sq Mean Sq F value Pr(>F)
N         3 20020   6674  37.686 2.46e-12 ***
N:V       6    322     54   0.303   0.932
Residuals 45  7969   177
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

La configuración del modelo usando `lmer()`, se haría del siguiente modo.

```
library(lme4)
modelo_split_plot = lmer(Y ~ 1 + N * V + (1 | V:B), data = oats_df)
modelo_split_plot
  Linear mixed model fit by REML ['lmerMod']
  Formula: Y ~ 1 + N * V + (1 | V:B)
  Data: oats_df
  REML criterion at convergence: 538.1656
  Random effects:
  Groups   Name        Std.Dev.
  V:B      (Intercept) 17.90
  Residual           13.31
  Number of obs: 72, groups: V:B, 18
  Fixed Effects:
            (Intercept)          N.L          N.Q          N.C
            104.5000    33.6901   -4.1667   -0.8199
  VMarvellous    VVictory  N.L:VMarvellous  N.Q:VMarvellous
            5.2917    -6.8750   -4.8075   -1.9167
  N.C:VMarvellous  N.L:VVictory  N.Q:VVictory  N.C:VVictory
            3.9877    2.5715   -1.0833   -2.8696

summary(modelo_split_plot)
  Linear mixed model fit by REML ['lmerMod']
  Formula: Y ~ 1 + N * V + (1 | V:B)
  Data: oats_df
  REML criterion at convergence: 538.2
  Scaled residuals:
  Min    1Q   Median    3Q   Max
-1.83746 -0.64204  0.01107  0.61254  1.61722
  Random effects:
  Groups   Name        Variance Std.Dev.
  V:B      (Intercept) 320.5    17.90
  Residual           177.1    13.31
  Number of obs: 72, groups: V:B, 18
  Fixed effects:
            Estimate Std. Error t value
  (Intercept) 104.5000   7.7975 13.402
  N.L          33.6901   5.4327  6.201
  N.Q          -4.1667   5.4327 -0.767
  N.C          -0.8199   5.4327 -0.151
  VMarvellous   5.2917  11.0274  0.480
  VVictory     -6.8750  11.0274 -0.623
  N.L:VMarvellous -4.8075  7.6830 -0.626
  N.Q:VMarvellous -1.9167  7.6830 -0.249
  N.C:VMarvellous  3.9877  7.6830  0.519
  N.L:VVictory    2.5715  7.6830  0.335
```

```

N.Q:VVictory    -1.0833    7.6830  -0.141
N.C:VVictory     -2.8696    7.6830  -0.374

Correlation of Fixed Effects:
              (Intr) N.L   N.Q   N.C   VMrvll1 VVctry N.L:VM N.Q:VM N.C:VM
N.L            0.000
N.Q            0.000  0.000
N.C            0.000  0.000  0.000
VMarvellous  -0.707  0.000  0.000  0.000
VVictory      -0.707  0.000  0.000  0.000  0.500
N.L:VMrvlls   0.000 -0.707  0.000  0.000  0.000  0.000
N.Q:VMrvlls   0.000  0.000 -0.707  0.000  0.000  0.000  0.000
N.C:VMrvlls   0.000  0.000  0.000 -0.707  0.000  0.000  0.000  0.000
N.L:VVictry   0.000 -0.707  0.000  0.000  0.000  0.000  0.500  0.000  0.000
N.Q:VVictry   0.000  0.000 -0.707  0.000  0.000  0.000  0.000  0.500  0.000
N.C:VVictry   0.000  0.000  0.000 -0.707  0.000  0.000  0.000  0.000  0.500
N.L:VV       N.Q:VV

N.L
N.Q
N.C
VMarvellous
VVictory
N.L:VMrvlls
N.Q:VMrvlls
N.C:VMrvlls
N.L:VVictry
N.Q:VVictry  0.000
N.C:VVictry  0.000  0.000

```

2. MODELO LINEAL GENERALIZADO

2.1. FORMULACIÓN GENERAL

Los modelos lineales (regresión, ANOVA, ANCOVA), se basan en los siguientes supuestos:

1. Los errores se distribuyen normalmente.
2. La varianza es constante.
3. La variable dependiente se relaciona linealmente con las variables independientes.

De manera analítica tendríamos:

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \dots + \beta_k X_{ki} + u_i, \quad i = 1, 2, \dots, n \quad [143]$$

donde e_i está distribuida de cómo una normal de media cero, varianza constante (homocedástica), y donde la covarianza entre e_i y e_j es nula para $e_i \neq e_j$ (ausencia de autocorrelación). Es decir, estos supuestos llevan implícito que la distribución de la variable dependiente Y_i sea también una normal $Y_i \sim (\mu, \sigma^2)$, donde $\mu = \beta_0 + \beta_1 \bar{X}_1 + \beta_2 \bar{X}_2 + \dots + \beta_k \bar{X}_k$.

En muchas ocasiones, sin embargo, nos encontramos con que uno o varios de estos supuestos no se cumplen por la naturaleza de la información. En algunos casos, estos problemas se pueden llegar a solucionar mediante la transformación de la variable respuesta (por ejemplo tomando logaritmos). Sin embargo, estas transformaciones no siempre consiguen corregir la falta de normalidad, la heterocedasticidad (varianza no constante) o la no linealidad de nuestros datos.

Una alternativa a la transformación de la variable dependiente/respuesta y a la falta de normalidad es el uso de los modelos lineales generalizados (*MLG*).

Los *MLG* fueron formulados por John Nelder y Robert Wedderburn (1989) como una manera de unificar varios modelos estadísticos, incluyendo la regresión lineal, regresión logística y regresión de Poisson, bajo un solo marco teórico.

Los *MLG* son, por tanto, una extensión de los modelos lineales que permiten utilizar distribuciones no normales de los errores (binomiales, Poisson, gamma, etc.) y varianzas no constantes.

Los *MLG* permiten especificar distintos tipos de distribución de errores. Cayuela (2010) expone los siguientes ejemplos:

- Poisson, muy útiles para conteos de acontecimientos, por ejemplo: número de heridos por accidentes de tráfico; número de hogares asegurados que dan parte de siniestro al día.
- Binomiales, de gran utilidad para proporciones y datos de presencia/ausencia, por ejemplo: tasas de mortalidad; tasas de infección; porcentaje de siniestros mortales.
- Gamma, muy útiles con datos que muestran un coeficiente de variación constante, esto es, en donde la varianza aumenta según aumenta la media de la muestra de manera constante, por ejemplo: número de heridos en función del número de siniestros.
- Exponencial, muy útiles para los análisis de supervivencia.

Otra razón por la que un modelo lineal puede no ser adecuado para describir un fenómeno determinado es que la relación entre la variable respuesta y las variables independientes no es siempre lineal.

La función de vínculo se encarga de linealizar la relación entre la variable dependiente y las variables independientes mediante la transformación de la variable respuesta:

TABLA 15. FUNCIONES DE LIGADURA-VÍNCULO MÁS USADAS

Función Vínculo	Formula	Uso
Identidad	μ	Datos continuos con errores normales (regresión y ANOVA)
Logaritmica	$\log(\mu)$	Conteos con errores de tipo de Poisson
Logit	$\log\left(\frac{\mu}{n-\mu}\right)$	Proporciones (datos entre 0 y 1) con errores binomiales
Recíproca	$\frac{1}{\mu}$	Datos continuos con errores Gamma
Cuadrada	$\sqrt{\mu}$	Conteos
Exponencial	μ^n	Funciones de potencia

Fuente: (Cayuela, 2016)

TABLA 16. MODELOS MLG MÁS COMUNES

Tipo de análisis	Variable respuesta	Variable explicativa	Función vínculo	Distribución de errores
Regresión	Continua	Continua	Identidad	Normal
ANOVA	Continua	Factor	Identidad	Normal
Regresión	Continua	Continua	Identidad	Gamma
Regresión	Conteo	Continua	Recíproca	Poisson
Tabla de contingencia	Conteo	Factor	Logarítmica	Poisson
Proporciones	Proporción	Continua	Logarítmica	Binomial
Regresión logística	Binaria	Continua	Logit	Binomial
Análisis de supervivencia	Tiempo	Recíproca	Identidad	Exponencial

Fuente: (Cayuela, 2016)

La estimación de los parámetros β se realiza por máximo verosimilitud, y los ajustes de $\hat{\mu}_i$ se calculan como $g^{-1}(x'_i\beta)$, una vez estimados los parámetros del vector β .

La especificación de un MLG se realiza en tres partes:

- La componente aleatoria correspondiente a la variable Y_i que sigue una distribución de la familia exponencial (normal, log-normal, poisson, gamma,...)
- La componente sistemática, o predictor, que se denota η y corresponde al vector de n componentes $\eta_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \dots + \beta_k X_{ki}$
- La función de ligadura (o función link $g(\bullet)$) que relaciona la esperanza matemática de la variable con el predictor lineal $\eta_i = g(\mu_i)$, que debe de ser monótona y diferenciable.

Para ilustrar el procedimiento en R, vamos a trabajar con una selección de variables de la Encuesta de Presupuestos Familiares del año 2014.

Las variables seleccionadas han sido las siguientes: - nmiemb (Número de miembros del hogar) - nmiem10 (Número de miembros del hogar de 25 a 34 años) - nmiem11 (Número de miembros del hogar de 35 a 64 años) - nmiem12 (Número de miembros del hogar de 65 a 84 años) - nmiem13 (Número de miembros del hogar de 85 o más años) - numacti (Número de miembros activos en el hogar) - numinacti (Número de miembros no activos en el hogar) - numocu (Número de miembros ocupados en el hogar) - numnoci (Número de miembros no ocupados en el hogar) - numestu (Número de estudiantes en el hogar) - numnoestu (Número de no estudiantes en el hogar) - tiphogar1 (Tipo de hogar -primera clasificación-) - situocuhog (Situación del hogar respecto a la ocupación) - situacthog (Situación del hogar respecto a la actividad) - impexac (ingresos corrientes del hogar)

Todas las variables excepto tiphogar1, situocuhog y situacthog son numéricas.

Las categorías de tiphogar1 son:

Hogar de un solo adulto.

1. Una persona de 65 o más años.
2. Una persona de 30 a 64 años.
3. Una persona de menos de 30 años.
4. Un adulto con niños menores de 16 años.

Pareja sin hijos.

5. Pareja sin hijos teniendo al menos uno de los miembros 65 años o más.
6. Pareja sin hijos teniendo los dos miembros menos de 65 años.

Pareja con hijos menores de 16 años.

7. Pareja con un hijo menor de 16 años.
8. Pareja con dos hijos menores de 16 años.
9. Pareja con tres o más hijos menores de 16 años.

Otras familias nucleares.

10. Padre o madre solo, con al menos un hijo de 16 o más años.
11. Pareja con al menos un hijo de 16 o más años.
12. Otros hogares.

Las categorías de situocuhog son:

1. El sustentador principal y el cónyuge ocupados, al menos otro de los miembros también ocupado.
2. El sustentador principal y el cónyuge ocupados, ninguno de los otros miembros ocupados (si es que los hay).
3. El sustentador principal o el cónyuge ocupado, otro de los miembros ocupado.
4. El sustentador principal o el cónyuge ocupado, al menos otros dos miembros ocupados.
5. El sustentador principal o el cónyuge ocupado, ninguno de los otros miembros ocupado (si es que los hay).
6. Ni el sustentador principal ni su cónyuge ocupado, otro miembro ocupado.
7. Ni el sustentador principal ni su cónyuge ocupados, al menos otros dos miembros ocupados.
8. Ningún ocupado en el hogar. -9. No consta.

En tanto que las categorías de situacthog son:

1. El sustentador principal y el cónyuge activos, al menos otro de los miembros también activo.
2. El sustentador principal y el cónyuge activos, ninguno de los otros miembros activos (si es que los hay).
3. El sustentador principal o el cónyuge activo, otro de los miembros también activo.
4. El sustentador principal o el cónyuge activo, al menos otros dos miembros activos.
5. El sustentador principal o el cónyuge activo, ninguno de los otros miembros activos (si es que los hay).
6. Ni el sustentador principal ni su cónyuge activos, otro miembro activo.
7. Ni el sustentador principal ni su cónyuge activos, al menos otros dos miembros activos.
8. Ningún activo en el hogar. -9. No consta.

Suponiendo que la información está contenida en un fichero denominado *datos.csv*, para cargarlo especificamos en primer lugar el directorio de trabajo con *setwd* y a continuación lo asignamos a un *data.frame* mediante la instrucción *read.csv*.

```
setwd("C:/Trabajo/Libro Metodos/Capítulo 4")
datos <- read.csv('datos.csv',sep=',',header=TRUE)
head(datos)
```

```

id nmiemb nmiem10 nmiem11 nmiem12 nmiem13 numacti numinacti numocu
1 55      4      0      2      0      0      2      1      2
2 56      2      2      0      0      0      2      0      1
3 57      1      0      0      1      0      0      1      0
4 58      3      0      2      0      0      2      0      2
5 59      3      0      2      0      0      2      0      1
6 60      1      0      1      0      0      0      1      0
numnoci numestu numnoestu
1      1      1      2
2      1      0      2
3      1      0      1
4      0      0      2
5      1      0      2
6      1      0      1

```

tiphogar1

- 1 Pareja con al menos un hijo de 16 o más años
- 2 Pareja sin hijos teniendo los dos miembros menos de 65 años
- 3 Una persona de 65 o más años
- 4 Pareja con un hijo menor de 16 años
- 5 Pareja con un hijo menor de 16 años
- 6 Una persona de 30 a 64 años

situocuhog

- 1 El sustentador principal y el cónyuge ocupados, ninguno de los otros miembros ocupados (si es que los hay)
- 2 El sustentador principal o el cónyuge ocupado, ninguno de los otros miembros ocupado (si es que los hay)
- 3 Ningún ocupado en el hogar
- 4 El sustentador principal y el cónyuge ocupados, ninguno de los otros miembros ocupados (si es que los hay)
- 5 El sustentador principal o el cónyuge ocupado, ninguno de los otros miembros ocupado (si es que los hay)
- 6 Ningún ocupado en el hogar

situacthog

- 1 El sustentador principal y el cónyuge activos, ninguno de los otros miembros activos (si es que los hay)
- 2 El sustentador principal y el cónyuge activos, ninguno de los otros miembros activos (si es que los hay)
- 3 Ningún activo en el hogar
- 4 El sustentador principal y el cónyuge activos, ninguno de los otros miembros activos (si es que los hay)
- 5 El sustentador principal y el cónyuge activos, ninguno de los otros miembros activos (si es que los hay)
- 6 Ningún activo en el hogar

impexac

```

1   2142
2   1550
3    300
4   2100
5   1305
6    826

```

Utilizando estas variables vamos a estimar un modelo para predecir los ingresos del hogar utilizando la distribución de poisson y un tipo de función vinculo logarítmica. Para ello ejecutamos el siguiente "Chunk":

```

est2 <- glm(impexac ~ nmiemb + nmiem11 + nmiem13 + numacti + tiphogar1 + situocuhog
+ situacthog + numnoestu, data=datos,family=poisson (link = "log"))
coef(est2)

```

```

(Intercept)
7.56483558

```

nmiemb
-0.04558776

nmiem11
0.06870008

nmiem13
-0.01786729

numinacti
0.09262404

tiphogar1Padre o madre solo, con al menos un hijo de 16 o más años
-0.04584300

tiphogar1Pareja con al menos un hijo de 16 o más años
-0.02912191

tiphogar1Pareja con dos hijos menores de 16 años
0.05740392

tiphogar1Pareja con tres o más hijos menores de 16 años
0.13250932

tiphogar1Pareja con un hijo menor de 16 años
-0.07656341

tiphogar1Pareja sin hijos teniendo al menos uno de los miembros 65 años o más
0.03865313

tiphogar1Pareja sin hijos teniendo los dos miembros menos de 65 años
-0.05638398

tiphogar1Un adulto con niños menores de 16 años
-0.30609531

tiphogar1Una persona de 30 a 64 años
-0.33412036

tiphogar1Una persona de 65 o más años
-0.32429945

tiphogar1Una persona de menos de 30 años
-0.74317174

situocuhogEl sustentador principal o el cónyugeocupado, al menos otros dos miembros ocupados
0.22591676

situocuhogEl sustentador principal o el cónyugeocupado, ninguno de los otros miembros ocupado (si es que los hay)
-0.38042030

situocuhogEl sustentador principal y el cónyuge ocupados, ninguno de los otros miembros ocupados (si es que los hay)
0.17888241

situocuhogEl sustentador principal y el cónyugeocupados, al menos otro de los miembros también ocupado
0.52414177

situocuhogNi el sustentador principal ni su cónyuge ocupado, otro miembro ocupado
-0.44280173

situocuhogNi el sustentador principal ni su cónyuge ocupados, al menos otros dos miembros ocupados
-0.01427409

situocuhogNingún ocupado en el hogar
-1.02238143

situacthogEl sustentador principal o el cónyuge activo, ninguno de los otros miembros activos (si es que los hay)
0.16955273

situacthogEl sustentador principal o el cónyuge activo, otro de los miembros también activo
0.03442693

situacthogEl sustentador principal y el cónyuge activos, al menos otro de los miembros también activo
-0.13855351

situacthogEl sustentador principal y el cónyuge activos, ninguno de los otros miembros activos (si es que los hay)
0.06609112

situacthogNi el sustentador principal ni su cónyuge activos, al menos otros dos miembros activos
0.25906870

situacthogNi el sustentador principal ni su cónyuge activos, otro miembro activo
0.44059994

situacthogNingún activo en el hogar
0.58468108

numnoestu
0.07785444

De forma análoga al modelo lineal, R dispone de la función *glmer()* (Generalized Linear Mixed-Effects Models) en el paquete *lme4*, para tratar el caso de los modelos mixtos.

2.2. MODELOS CON VARIABLES CUALITATIVAS ENDÓGENAS

2.2.1. *Modelo probabilístico lineal*

El modelo de probabilidad lineal se caracteriza por tener la variable endógena Y dicotómica o binaria, es decir toma el valor $Y_i = 1$ si un determinado suceso ocurre y el valor $Y_i = 0$ en caso contrario. Estos modelos están muy extendidos en el análisis estadístico pero encuentran una difícil aplicación en Economía debido a las dificultades de interpretación económica de los resultados que ofrecen este tipo de investigaciones. A este respecto, hay que considerar que estos modelos lo que realmente investigan es la probabilidad de que se dé una opción (valores $Y_i = 1$) o no se dé ($Y_i = 0$).

A pesar del carácter dicotómico de la variable endógena, el modelo de probabilidad lineal se especifica de la forma habitual, teniendo presente que las variables exógenas no son dicotómicas sino continuas:

$$Y_i = \beta_0 + \beta_1 X_i + e_i \quad [144]$$

De acuerdo con esta expresión, y dado el hecho de que la variable endógena tome valores discretos (1 o 0), el término de perturbación e_i , puede tomar también dos valores únicamente:

- Si $Y_i = 0 \Rightarrow e_i = -\beta_0 - \beta_1 X_i$ con probabilidad p .
- Si $Y_i = 1 \Rightarrow e_i = 1 - \beta_0 - \beta_1 X_i$ con probabilidad $(1 - p)$.

Dado que la esperanza del término de error ha de ser nula $E(e_i) = 0$, entonces se demuestra que $p = 1 - \beta_0 - \beta_1 X_i$ y $(1 - p) = \beta_0 + \beta_1 X_i$, lo que permite evaluar la probabilidad de que la variable endógena tome el valor correspondiente:

- $Prob(Y_i = 0) = Prob(e_i = -\beta_0 - \beta_1 X_i) = p = 1 - \beta_0 - \beta_1 X_i$.
- $Prob(Y_i = 1) = Prob(e_i = 1 - \beta_0 - \beta_1 X_i) = (1 - p) = \beta_0 + \beta_1 X_i$.

A su vez, la varianza del término de perturbación se calcularía a partir de:

$$Var(e_i) = p(1 - p) = (1 - \beta_0 - \beta_1 X_i)(\beta_0 + \beta_1 X_i) \quad [145]$$

Una problemática inherente a los estimadores MCO de estos modelos es la siguiente:

- La perturbación aleatoria e_i no sigue una distribución Normal. Es sencillo observar este hecho ya que el carácter binario (1 o 0) de la variable endógena afecta a la distribución de la perturbación, teniendo ésta una distribución Binomial. Este problema se atenúa cuando se utilizan tamaños de muestra (N) grandes en donde la distribución Binomial es susceptible de aproximarse a una Normal.
- La perturbación aleatoria no tiene una varianza constante (es heteroscedástica), lo cual supone una falta de eficiencia. Para solucionarlo habría que realizar transformaciones que nos diesen una perturbación homocedástica; esta transformación consiste en multiplicar todas las variables por una cierta cantidad que elimine el problema de la heteroscedasticidad. Dicha cantidad es:

$$\frac{1}{\sqrt{(1 - \hat{\beta}_0 - \hat{\beta}_1 X_i)(\hat{\beta}_0 + \hat{\beta}_1 X_i)}} \quad [146]$$

siendo $\hat{\beta}_0$ y $\hat{\beta}_1$ las estimaciones MCO del modelo.

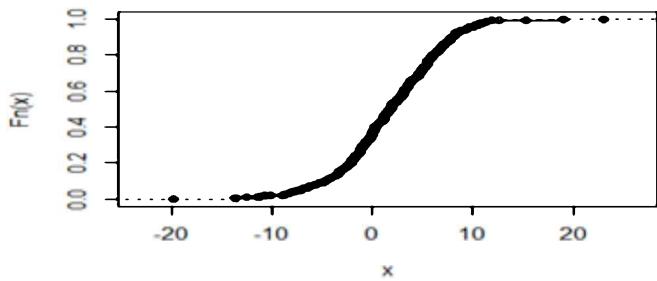
- No obstante, el mayor problema que plantean estos modelos es que las predicciones realizadas sobre la variable endógena no siempre se encuentran en el intervalo [0,1], ya que pueden ser menores que cero y mayores que uno. Este problema tiene dos soluciones, una es tomar como valor cero todas las estimaciones de la variable endógena con valores negativos, y uno cuando estas resulten mayores que uno; la segunda solución es utilizar funciones de distribución que estén acotadas entre cero y uno como son la Logística y la Normal; de éstas se derivan los modelos Logit y Probit que pasamos a ver a continuación.

2.2.2. Modelo Logit

Planteamiento del modelo

El problema que presentan los modelos probabilísticos lineales en cuanto a la existencia de predicciones establecidas fuera de rango (negativas o mayores que uno), es debido a que utilizan una función de probabilidad que depende linealmente de las variables explicativas (X_i), que se resolverían acotando dicha distribución de probabilidad. El modelo Logit en concreto utiliza, para ello, la función de distribución logística:

FIGURA 14. FUNCIÓN DE DISTRIBUCIÓN LOGÍSTICA



Fuente: Elaboración propia

Debido a que la función de distribución logística no tiene forma lineal, el modelo Logit se estima de forma diferente, así en vez de minimizar las sumas de las diferencias al cuadrado entre los valores observados y los estimados por el modelo, el carácter no lineal de los modelos Logit requiere la utilización del método de Máxima Verosimilitud para ser estimado, maximizando la verosimilitud de que un suceso tenga lugar, aunque se podría estimar por MCO mediante una transformación logarítmica de los datos (Gujarati, 1997).

La probabilidad de que $Y_i = 0$ (p_i) se define ahora mediante la siguiente expresión:

$$p_i = \frac{1}{1+e^{-z_i}} \quad [147]$$

donde $z_i = \beta_0 + \sum_j \beta_j X_{ji}$.

La probabilidad de que $Y_i = 1$ ($1 - p_i$) sería:

$$1 - p_i = \frac{1}{1+e^{z_i}} \quad [148]$$

En consecuencia, la razón entre ambas será igual a:

$$\frac{p_i}{1-p_i} = \frac{1+e^{z_i}}{1+e^{-z_i}} = e^{z_i} \quad [149]$$

Tomando el logaritmo de la expresión anterior se obtiene:

$$L_i = \log\left(\frac{p_i}{1-p_i}\right) = \log(e^{z_i}) = \beta_0 + \sum_{j=1}^k \beta_j X_{ji} + u_i \quad [150]$$

donde L_i es denominado Logit.

Los coeficientes β indican el cambio en el Logit causado por el cambio en una unidad en el valor de X_i , mientras que los e^β definen el cambio en la razón de probabilidades $\frac{p}{1-p}$ causado por el cambio en una unidad en el valor de X_i . Si β es positivo, e^β será mayor que 1, es decir, $\frac{p}{1-p}$ se incrementará; si β es negativo, e^β será menor que 1, y $\frac{p}{1-p}$ disminuirá. Adicionalmente, puede demostrarse que el cambio en la probabilidad (p) causado por el cambio en una unidad en el valor de X_i es $\beta\left(\frac{p}{1-p}\right)$, es decir, depende no solo del coeficiente, sino también del nivel de probabilidad a partir del cual se mide el cambio.

A la hora de estimar un modelo Logit, hay que tener presente que además de los valores X_i , se necesitan los valores del Logit (L_i). Por otro lado, la estimación de los coeficientes del modelo ha de realizarse utilizando el método de Máxima Verosimilitud, a través del algoritmo de Newton-Raphson, es decir, eligiendo como estimadores de los coeficientes a aquellos que maximizan la función de verosimilitud, construida sobre la base de $p = \frac{1}{1+e^{-z}}$.

Pero si tenemos la posibilidad de agrupar los datos individuales, entonces podría estimarse el modelo por MCO.

Contrastes sobre los parámetros

Para evaluar si cada variable individualmente contribuye significativamente al modelo se utiliza el estadístico de Wald, definido como el cociente entre el valor del parámetro estimado al cuadrado dividido entre su varianza:

$$W(\beta_j) = \frac{\hat{\beta}_j^2}{Var(\hat{\beta}_j)} \quad [151]$$

Este estadístico, bajo la hipótesis nula $H_0: \beta_j = 0$, se distribuye como una Chi-Cuadrado con un grado de libertad. Una estimación adecuada de $Var(\hat{\beta}_j)$ se puede obtener a través del inverso de la información de Fisher del parámetro.

En la base de datos “datos” definimos como pobres a aquellos hogares que tienen un ingreso “per cápita” inferior al 60% de la mediana.

```
datos$ingpc=datos$impexac/datos$nmiemb
datos$pobre=ifelse(datos$ingpc<0.6*median(datos$ingpc),1,0)
str(datos)
'data.frame': 22146 obs. of 18 variables:
 $ id      : int 55 56 57 58 59 60 61 62 63 1 ...
 $ nmiemb  : int 4 2 1 3 3 1 4 1 2 3 ...
 $ nmiem10 : int 0 2 0 0 0 0 0 0 0 1 ...
 $ nmiem11 : int 2 0 0 2 2 1 2 0 0 2 ...
 $ nmiem12 : int 0 0 1 0 0 0 0 1 0 0 ...
 $ nmiem13 : int 0 0 0 0 0 0 0 0 2 0 ...
 $ numacti : int 2 2 0 2 2 0 2 0 0 1 ...
 $ numinacti : int 1 0 1 0 0 1 0 1 2 2 ...
 $ numocu  : int 2 1 0 2 1 0 1 0 0 1 ...
 $ numnocu : int 1 1 1 0 1 1 1 1 2 2 ...
 $ numestu : int 1 0 0 0 0 0 0 0 0 0 ...
 $ numnoestu: int 2 2 1 2 2 1 2 1 2 3 ...
 $ tiphogar1: Factor w/ 12 levels "Otros hogares",...: 3 8 11 6 6 10 4 11 7 3 ...
 $ situocuhog: Factor w/ 8 levels "El sustentador principal o el cónyuge ocupado, otro de los miembros ocupado",...: 4 3 8 4 3 8 3 8 8 NA ...
 $ situacthog: Factor w/ 8 levels "El sustentador principal o el cónyuge activo, al menos otros dos miembros activos",...: 5 5 8 5 5 8 5 8 8 NA ...
 $ impexac  : num 2142 1550 300 2100 1305 ...
 $ ingpc    : num 536 775 300 700 435 ...
 $ pobre    : num 0 0 1 0 0 0 1 0 0 0 ...
 
# Frecuencias variable respuesta
table(datos$pobre)

 0     1
17167 4979

# Frecuencias relativas variable respuesta
prop.table(table(datos$pobre))

 0     1
0.7751738 0.2248262
```

Estimamos a continuación un modelo logit con glm y realizamos un conteo para ver los resultados obtenidos.

```
# Regresion logistica
est3 <- glm(datos$pobre ~ nmiemb + nmiem11 + numinacti + numocu +
```

```
tiphogar1 + situocuhog, data=datos,family=binomial)
est3

Call: glm(formula = datos$pobre ~ nmiemb + nmiem11 + numinacti + numocu +
tiphogar1 + situocuhog, family = binomial, data = datos)

Coefficients:

(Intercept)                         -2.7081
nmiemb                                1.3216
nmiem11                               -0.1508
numinacti                             -0.8813
numocu                                 -1.5816
tiphogar1Padre o madre solo, con al menos un hijo de 16 o más años
0.5893
tiphogar1Pareja con al menos un hijo de 16 o más años
0.5113
tiphogar1Pareja con dos hijos menores de 16 años
-0.3840
tiphogar1Pareja con tres o más hijos menores de 16 años
-0.6002
tiphogar1Pareja con un hijo menor de 16 años
0.2479
tiphogar1Pareja sin hijos teniendo al menos uno de los miembros 65 años o más
-0.4082
tiphogar1Pareja sin hijos teniendo los dos miembros menos de 65 años
0.2108
tiphogar1Un adulto con niños menores de 16 años
0.7952
tiphogar1Una persona de 30 a 64 años
0.6961
tiphogar1Una persona de 65 o más años
-1.0251
tiphogar1Una persona de menos de 30 años
1.5050
situocuhogEl sustentador principal o el cónyugeocupado, al menos otros dos miembros
ocupados
-0.3353
situocuhogEl sustentador principal o el cónyugeocupado, ninguno de los otros miembros
ocupado (si es que los hay)
0.1355
```

situocuhogEl sustentador principal y el cónyuge ocupados, ninguno de los otros miembros ocupados (si es que los hay)
-0.9875

situocuhogEl sustentador principal y el cónyuge ocupados, al menos otro de los miembros también ocupado
-0.3769

situocuhogNi el sustentador principal ni su cónyuge ocupado, otro miembro ocupado
-0.6623

situocuhogNi el sustentador principal ni su cónyuge ocupados, al menos otros dos miembros ocupados
-0.8303

situocuhogNingún ocupado en el hogar
0.2852

Degrees of Freedom: 18881 Total (i.e. Null); 18859 Residual
(3264 observations deleted due to missingness)

Null Deviance: 20670

Residual Deviance: 14770 AIC: 14820

Tabla de clasificación Prest3edichos * Observados

est3.probs=predict(est3,type="response")

est3.pred=ifelse(est3.probs>0.5,1,0)

clasif3 = table(est3.pred,est3\$y)

clasif3

est3.pred	0	1
0	13480	2422
1	930	2050

Porcentajes fila

prop.table(clasif3,1)

est3.pred	0	1
0	0.8476921	0.1523079
1	0.3120805	0.6879195

Porcentaje global de aciertos

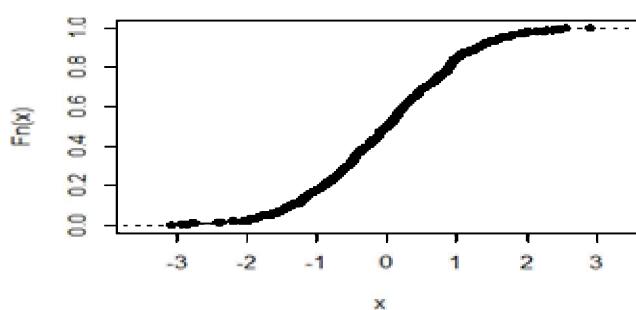
mean(est3.pred==est3\$y)

[1] 0.8224764

2.2.3. Modelo Probit

Mientras que el modelo Logit utiliza la función de distribución logística para acotar la distribución de probabilidad en el modelo de probabilidad lineal, el modelo Probit utiliza la función de distribución Normal.

FIGURA 15. FUNCIÓN DE DISTRIBUCIÓN NORMAL



Fuente: Elaboración propia

Las funciones de distribución normal y logística son muy semejantes; la diferencia principal es que la función de distribución normal se acerca más rápidamente a los ejes que la logística.

Para entender la filosofía del modelo Probit, vamos a suponer que existe una variable desconocida s que cumple lo siguiente:

- Si $I_i = \beta_0 + \sum_j \beta_j X_{ji} \geq s$ entonces $Y_i = 1$
- Si $I_i = \beta_0 + \sum_j \beta_j X_{ji} < s$ entonces $Y_i = 0$

Dado el supuesto de normalidad en un suceso, la probabilidad de que este sea menor o igual al valor (s), se calcula a partir de la función de distribución acumulada de una distribución Normal estandarizada, esto es, con esperanza cero y desviación típica uno.

$$pr(Y_i = 1) = pr(I_i^* \geq I_i) = pr(\beta_0 + \sum_{j=1}^k \beta_j X_{ji} \geq s) = F(\beta_0 + \sum_{j=1}^k \beta_j X_{ji}) \quad [152]$$

F es la FDA normal estandar:

$$F(I_i) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\beta_0 + \sum_j \beta_j X_{ji}} e^{-\frac{z^2}{2}} d(z) \quad [153]$$

donde Z es la variable normal estandarizada $Z \sim N(0, \sigma^2)$

Lo anterior equivale a que la relación entre la endógena y las explicativas venga dada por la siguiente expresión:

$$y_i = F(I_i = \beta_0 + \sum_{j=1}^k \beta_j X_{ji}) + u_i \Rightarrow y_i = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\beta_0 + \sum_j \beta_j X_{ji}} e^{-\frac{z^2}{2}} d(z) + u_i \quad [154]$$

Ahora para obtener información sobre I_i , se toma la inversa de $F(I_i)$:

$$I_i = F^{-1}(I_i) = F^{-1}(P) = \beta_0 + \sum_{j=1}^k \beta_j X_{ji} \quad [155]$$

Donde I_i es negativa siempre que $p_i < 0.5$; en la práctica se agrega el número 5 a I_i y a su resultado se le denomina Probit. Es decir, $Probit = 5 + I_i$.

El término de la perturbación es no obstante heteroscedástico. Gujarati (1999) sugiere que se realice la transformación comentada en el apartado 1.3.2.1 (análisis de la varianza de un factor), para que el modelo transformado sea homocedástico.

Al igual que ocurre en el logit, si tenemos la posibilidad de agrupar los datos individuales, entonces podría estimarse el modelo por MCO.

La estimación en R del modelo probit estimado en el ejemplo anterior se programa:

```
# Regresion probit
est4 <- glm(datos$pobre ~ nmiemb + nmiem11 + numinacti + numocu +
tiphogar1 + situocuhog, data=datos,family=binomial(link=probit))
est4

Call: glm(formula = datos$pobre ~ nmiemb + nmiem11 + numinacti + numocu +
tiphogar1 + situocuhog, family = binomial(link = probit),
data = datos)

Coefficients:
```

(Intercept)
-1.52054

nmiemb
0.74281

nmiem11
-0.08590

numinacti
-0.52252

numocu
-0.89245

tiphogar1Padre o madre solo, con al menos un hijo de 16 o más años
0.28253

tiphogar1Pareja con al menos un hijo de 16 o más años
0.28536

tiphogar1Pareja con dos hijos menores de 16 años
-0.25887

tiphogar1Pareja con tres o más hijos menores de 16 años
-0.35874

tiphogar1Pareja con un hijo menor de 16 años
0.10735

tiphogar1Pareja sin hijos teniendo al menos uno de los miembros 65 años o más
-0.28384

tiphogar1Pareja sin hijos teniendo los dos miembros menos de 65 años
0.06929

tiphogar1Un adulto con niños menores de 16 años
0.42496

tiphogar1Una persona de 30 a 64 años
0.29198

tiphogar1Una persona de 65 o más años
-0.59159

tiphogar1Una persona de menos de 30 años
0.80715

situocuhogEl sustentador principal o el cónyugeocupado, al menos otros dos miembros ocupados
-0.20866

situocuhogEl sustentador principal o el cónyugeocupado, ninguno de los otros miembros ocupado (si es que los hay)
0.11245

situocuhogEl sustentador principal y el cónyuge ocupados, ninguno de los otros miembros ocupados (si es que los hay)
-0.51384

situocuhogEl sustentador principal y el cónyugeocupados, al menos otro de los miembros también ocupado
-0.13813

situocuhogNi el sustentador principal ni su cónyuge ocupado, otro miembro ocupado
-0.36547

situocuhogNi el sustentador principal ni su cónyuge ocupados, al menos otros dos miembros ocupados
-0.43155

situocuhogNingún ocupado en el hogar
0.21552

```
Degrees of Freedom: 18881 Total (i.e. Null); 18859 Residual
(3264 observations deleted due to missingness)
Null Deviance: 20670
Residual Deviance: 14800 AIC: 14840
```

```
# Tabla de clasificación Predichos * Observados
est4.probs=predict(est4,type="response")
est4.pred=ifelse(est4.probs>0.5,1,0)
clasif4=table(est4.pred,est4$y)
clasif4
```

est4.pred	0	1
0	13495	2441
1	915	2031

```
# Porcentajes fila
prop.table(clasif4,1)
```

est4.pred	0	1
0	0.8468248	0.1531752
1	0.3105906	0.6894094

```
# Porcentaje global de aciertos
mean(est4.pred==est4$y)
[1] 0.8222646
```

2.2.4. Modelo Logit vs Modelo Probit

Desde un punto de vista teórico, si se plantean las covariables como directamente relacionadas con la probabilidad de éxito, entonces normalmente se elegiría la Regresión Logística, porque es el enlace canónico.

Si el resultado binario depende de una variable Gaussiana oculta parece razonable optar a priori, por razones teóricas, por el modelo Probit. Consideremos el siguiente ejemplo: se plantea la realización de un modelo de presión arterial alta como una función de algunas covariables. Parece razonable la asunción de que la presión arterial en sí misma se distribuya normalmente en la población, pero sin embargo, los médicos la dicotomizaron durante el estudio (es decir, solo se registró como "alta" o "normal").

Otra consideración es que tanto logit como probit son simétricos. Si se piensa que la probabilidad de éxito sube lentamente de cero, pero luego disminuye más rápidamente cuando se acerca a uno, sería preferible optar por una función vínculo tipo CLogLog, la cual se define como:

$$y = CLogLog(x) = \ln(-\ln(1 - x)) \Rightarrow x = CLogLog^{-1}(y) = 1 - e^{-ey} \quad [156]$$

Por último, cabe señalar que es improbable que el ajuste empírico del modelo a los datos sea de ayuda en la selección de una función vínculo, a menos que las formas de las funciones de enlace en cuestión difieran sustancialmente (cosa que no sucede en Logit y Probit). Por ejemplo, considere la siguiente simulación:

```
library(ggplot2)
set.seed(5000)
probMenor = vector(length=100)
```

```

devprobit = vector(length=100)
devlogit = vector(length=100)

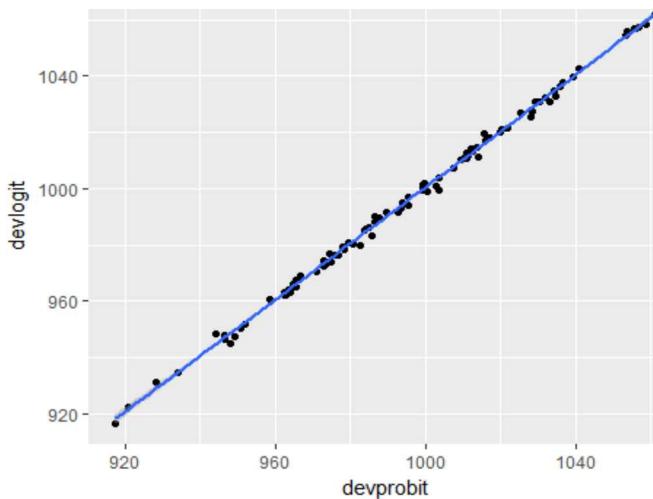
# Genera un modelo probit en cada paso (100 modelos) con 1.000 datos
for(i in 1:100{
  x = rnorm(1000)
  y = rbinom(n=1000, size=1, prob=pnorm(x))
  logitModel = glm(y~x, family=binomial(link="logit"))
  probitModel = glm(y~x, family=binomial(link="probit"))

  probMenor[i] = deviance(probitModel)<deviance(logitModel)
  devprobit[i] = deviance(probitModel)
  devlogit[i] = deviance(logitModel)
}

# Porcentaje de casos en que deviance Probit es menor a Deviance Logit (Probit mejor modelo)
sum(probMenor)/100
[1] 0.65

# Gráfico de ambas devianzas
comparaPL <- cbind(devprobit, devlogit, probMenor)
comparaPLdf <- as.data.frame(comparaPL)
ggplot(comparaPLdf, aes(devprobit, devlogit)) + geom_point() + geom_smooth()
`geom_smooth()` using method = 'loess'

```



Incluso cuando sabemos que los datos han sido generados por un modelo probit, y tenemos 1000 puntos de datos, el modelo probit solo da un mejor ajuste el 65% del tiempo, e incluso entonces, a menudo por solo una cantidad trivial. Consideremos, por ejemplo, las 5 primeras iteraciones:

```

comparaPL[1:5,]
  devprobit devlogit probMenor
[1,] 1013.6699 1014.5126     1
[2,] 1061.7100 1062.4248     1
[3,] 965.5274  967.6012     1
[4,] 1007.5088 1007.2238     0
[5,] 917.3673  916.2442     0

```

La razón de esto es simplemente que las funciones de enlace logit y probit producen salidas muy similares, al ser ambas prácticamente idénticas.

2.3. MODELO TOBIT

Este modelo está diseñado para el caso en que los valores de la variable respuesta Y están acotados inferior y/o superiormente. Luego la expresión matemática de este modelo sería la siguiente:

$$y_i^* = X_i' \beta + u_i \quad [157]$$

$$y_i = \begin{cases} a & \text{si } y_i^* \leq a \\ y_i^* & \text{si } a < y_i^* < b \\ b & \text{si } y_i^* \geq b \end{cases} \quad [158]$$

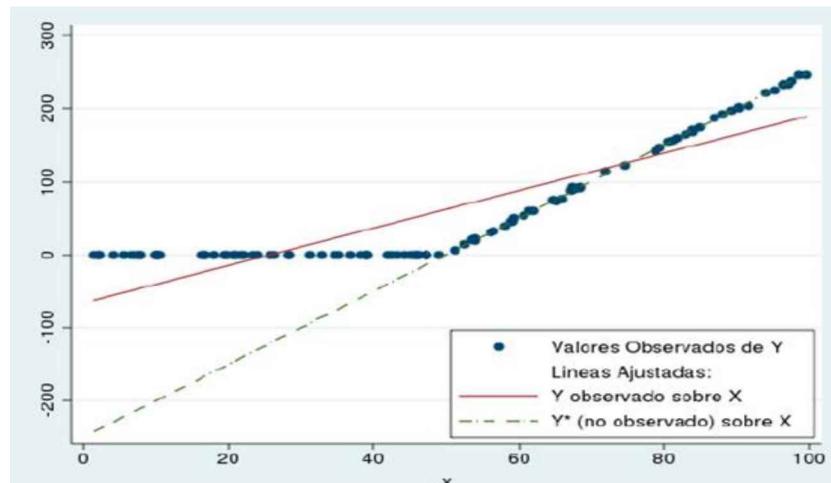
Un modelo de regresión lineal ignorando que hay observaciones censuradas ofrece una estimación sesgada.

Como ejemplo, pongamos el modelo propuesto por Tobin (1958), conocido como modelo estándar Tobit:

$$y_i = \begin{cases} 0 & \text{si } y_i^* < 0 \\ y_i^* & \text{si } y_i^* \geq 0 \end{cases} \quad [159]$$

Véase la Figura 16, donde se representan los datos, y las rectas de regresión de la muestra completa y la de los datos con valores mayores que cero en la variable respuesta.

FIGURA 16. REGRESIÓN CON VARIABLE DEPENDIENTE CENSURADA



Fuente: Elaboración propia

Como puede deducirse de su formulación, este modelo es un mixto entre un modelo Logit y uno de regresión lineal. Con la aplicación del esquema Logit, se deberá determinar las unidades iguales a 0 y las distintas de cero. En estas últimas es donde se aplicará la regresión lineal.

Implementamos a continuación un ejemplo en R. El fichero de datos de partida está en formato SPSS. La información que contiene son las puntuaciones de una prueba de lectura, otra de matemáticas, las relativas a la prueba de aptitud global y el tipo de programa de estudios cursado.

Queremos establecer la relación entre las pruebas de matemáticas y de lectura y el tipo de programa con la prueba de aptitud, donde la puntuación máxima es igual a 800.

```

library(AER)
library(censReg)

# Fijar directorio
setwd("C:/Trabajo/Libro Metodos/Capitulo 4")

# Lectura de fichero SPSS
Notas <- read.spss("Notas_Tobit.sav", use.value.labels=TRUE)

# Modelo de Regresión Censurada
modeloRegCen <- censReg(formula = apt ~ read + math + prog2, right = 800, data =
Notas)
summary(modeloRegCen)

Call:
censReg(formula = apt ~ read + math + prog2, right = 800, data = Notas)

Observations:
    Total Left-censored      Uncensored Right-censored
        200             0            183            17

Coefficients:
            Estimate Std. error t value Pr(> |t|)
(Intercept) 209.56597  32.77196  6.395 1.61e-10 ***
read         2.69794   0.61881  4.360 1.30e-05 ***
math         5.91448   0.70982  8.332 < 2e-16 ***
prog2general -12.71476  12.40646 -1.025 0.305434
prog2vocational -46.14390 13.72419 -3.362 0.000773 ***
logSigma     4.18474   0.05301 78.946 < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Newton-Raphson maximisation, 9 iterations
Return code 1: gradient close to zero
Log-likelihood: -1041.063 on 6 Df

```

Como ejercicio se propone contrastar los resultados de este modelo con los del modelo de regresión lineal.

3. EVALUACIÓN DE MODELOS

Para medir la bondad del ajuste de un modelo MLG se disponen de varias opciones:

- Cuando los datos están en forma binaria, una manera de detectar la falta de ajuste es realizando contrastes condicionales de razón de verosimilitudes entre modelos anidados.
- Cuando los datos se pueden agrupar, se pueden comparar las frecuencias esperadas y observadas en cada combinación de variables explicativas mediante el contraste usual con el estadístico χ^2 de Pearson o con el G^2 basado en la devianza. Estos estadísticos se distribuyen asintóticamente según una chi-cuadrado con grados de libertad igual al número de perfiles distintos menos el número de parámetros en el modelo, si el número de combinaciones o perfiles posibles de las variables explicativas no es elevado y se tienen suficiente número de frecuencias observadas y esperadas en cada combinación.
- Otra opción es utilizar el estadístico de Hosmer-Lemeshow, que realiza una partición de los datos en base a las probabilidades predichas, y analiza posteriormente la tabla de contingencia resultante a través de su estadístico χ^2 asociado.

- Otras aproximaciones calculan medidas tipo R^2 , o analizan el poder predictivo del modelo mediante la tasa de clasificaciones correctas o el análisis de curvas ROC.

Se describen a continuación los contrastes y métodos más utilizados y su aplicación en R. Vamos a comparar la bondad del ajuste de los modelos Logit y Probit sobre la pobreza desarrollada en los apartados 2.2.2 y 2.2.3.

3.1. DEVIANZA. ESTADÍSTICO G² DE WILKS DE RAZÓN DE VEROSIMILITUDES

La función de verosimilitud en un proceso binomial (modelos Logit y Probit) con y_i el número de éxitos en m_i $i = 1 \dots n$ (datos agrupados) y un conjunto de predictores X_i

$$y_i/X \sim Bin(m_i, \theta(X)_i) \quad [160]$$

es de la forma:

$$L = \prod_{i=1}^n P(Y_i = y_i/X) = \prod_{i=1}^n \binom{m_i}{y_i} \theta(X)_i^{y_i} (1 - \theta(X)_i)^{m_i - y_i} \quad [161]$$

Donde θ_X para el modelo logístico será:

$$\theta(X)_i = \frac{1}{1 + e^{-(\beta_0 + \sum_j \beta_j X_{ji})}} \quad [162]$$

mientras que para el modelo Probit:

$$\theta(X)_i = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\beta_0 + \sum_j \beta_j X_{ji}} e^{-\frac{z^2}{2}} d(z) \quad [163]$$

estimando y_i a través de la expresión:

$$\hat{y}_i = m_i \cdot \hat{\theta}(X)_i \quad [164]$$

La log-verosimilitud, operando, será por tanto:

$$LogL = \sum_{i=1}^n [y_i \cdot \log(\frac{\theta(X)_i}{1 - \theta(X)_i}) + m_i \cdot \log(1 - \theta(X)_i) + \log(\binom{m_i}{y_i})] \quad [165]$$

El concepto de suma de cuadrados se sustituye por el de *devianza*. La devianza asociada a un modelo (M) se basa en comparar la log-verosimilitud bajo dicho modelo con la log-verosimilitud bajo el modelo saturado (S). En el modelo saturado, θ_X se estima a través de la proporción observada de éxitos, es decir, $\frac{y_i}{m_i}$.

La devianza se define entonces como dos veces la diferencia entre estas dos log-verosimilitudes. Operando se obtiene:

$$G^2(M) = 2[\log(L_S) - \log(L_M)] = 2 \sum_{i=1}^n [y_i \cdot \log(\frac{y_i}{\hat{y}_i}) + (m_i - y_i) \cdot \log(\frac{m_i - y_i}{m_i - \hat{y}_i})] \quad [166]$$

Bajo la hipótesis nula de que el modelo (M) es adecuado y los m_i son suficientemente grandes, $G^2(M)$ se distribuye aproximadamente como una χ^2_{n-k-1} , donde n es el número de datos o muestras binomiales (o número de combinaciones de valores de las k variables explicativas) y $k+1$, por tanto, será el número de parámetros estimados (número de variables más término independiente). La diferencia entre la devianza de dos modelos anidados también sigue una distribución χ^2 y es la base de los procedimientos de selección de modelos.

En el caso de datos no agrupados ($m_i = 1 \forall i$), la devianza no es una medida de la bondad del ajuste del modelo M , y tampoco sigue una distribución χ^2 . No obstante, la diferencia entre las devianzas de dos modelos ajustados si se distribuye como una χ^2 y se puede utilizar para seleccionar el modelo con mejor ajuste.

Los residuos de la devianza, definidos como:

$$e_i^D = signo(y_i - \hat{y}_i) \left(2[y_i \cdot log(\frac{y_i}{\hat{y}_i}) + (m_i - y_i) \cdot log(\frac{m_i - y_i}{m_i - \hat{y}_i})] \right)^{1/2} \quad [167]$$

bajo la hipótesis nula de que son igual a 0, tienen distribución asintótica de media 0 y varianza menor que 1.

El R también ofrece la posibilidad de calcular estos residuos estandarizados y studentizados, a través de las funciones *rstandard* y *rstudent*.

Partiendo de los modelos Logit y Probit anteriores (se ha incluido también el código de ejecución del modelo), la Devianza la podemos obtener a partir de la función *deviance* o del campo del modelo \$deviance. Para obtener el p-valor de $G^2(M)$ se ha de utilizar la función *pchisq*, obteniendo los grados de libertad a partir del campo \$df.residual de cada modelo.

A partir de la función *residuals*, podemos obtener los residuos de la devianza, del siguiente modo:

```
resid.dev <- residuals(modelo, type = "deviance")
```

donde modelo sería est3 o est4 en nuestro caso.

```
est3 <- glm(datos$pobre ~ nmiemb + nmiem11 + numinacti + numocu + tiphogar1 +
  situocuhog, data=datos,family=binomial)
est4 <- glm(datos$pobre ~ nmiemb + nmiem11 + numinacti + numocu + tiphogar1 +
  situocuhog, data=datos,family=binomial(link=probit))

# Devianza
paste("G2 Logit: ", est3$deviance)
[1] "G2 Logit: 14771.4107366346"
paste("G2 Probit: ", deviance(est4))
[1] "G2 Probit: 14795.7599513679"

# Cálculo de p-valor para la Devianza (no existe función específica)
pD.logit <- 1 - pchisq(deviance(est3), est3$df.residual)
pD.probit <- 1 - pchisq(deviance(est4), est4$df.residual)
paste("p-valor Logit: ", pD.logit)
[1] "p-valor Logit: 1"
paste("p-valor Probit: ", pD.probit)
[1] "p-valor Probit: 1"
```

3.2. ESTADÍSTICO χ^2 DE PEARSON

La formulación del estadístico es la siguiente:

$$\chi^2(M) = \sum_{i=1}^n \frac{(y_i - m_i \hat{\theta}(X)_i)^2}{m_i \hat{\theta}(X)_i (1 - \hat{\theta}(X)_i)} \quad [168]$$

dónde $\hat{\theta}(X)_i$ son las probabilidades estimadas en cada combinación de las variables predictoras y m_i el número de casos totales en cada combinación.

Este estadístico tiene, bajo la hipótesis nula de que el modelo es adecuado, la misma distribución asintótica que el estadístico de Wilks.

El estadístico χ^2 de Pearson se puede calcular fácilmente partiendo de los residuos de pearson (que también pueden ser estandarizados y studentizados), tal como veremos en el ejemplo siguiente.

$$e_i^P = \text{signo}(y_i - m_i \hat{\theta}(X)_i) \sqrt{\frac{(y_i - m_i \hat{\theta}(X)_i)^2}{m_i \hat{\theta}(X)_i (1 - \hat{\theta}(X)_i)}} \quad [169]$$

```
# Estadístico chi-cuadrado
chi2.logit <- sum(residuals(est3, type = "pearson")^2)
chi2.probit <- sum(residuals(est4, type = "pearson")^2)

# p-valores
pchi.logit <- 1 - pchisq(chi2.logit, est3$df.residual)
pchi.probit <- 1 - pchisq(chi2.probit, est4$df.residual)

paste("Logit. Chi-cuadrado, p-valor: ", chi2.logit, " ", pchi.logit)
[1] "Logit. Chi-cuadrado, p-valor: 24428.2140300399    0"
paste("Probit. Chi-cuadrado, p-valor: ", chi2.probit, " ", pchi.probit)
[1] "Probit. Chi-cuadrado, p-valor: 684689.526492789   0"
```

3.3. CRITERIO DE INFORMACIÓN DE AKAIKE (AIC) Y CRITERIO DE INFORMACIÓN BAYESIANO (BIC)

El estadístico AIC (Akaike Information Criterion), formulado por Akaike (1974):

$$AIC = p(k + 1) - 2\log(L_M) \quad \text{con } p = 2 \Rightarrow AIC = 2(k + 1) - 2\log(L_M) \quad [170]$$

donde $\log(L_M)$ es el valor en el óptimo del logaritmo de la función de verosimilitud del modelo M con $k + 1$ parámetros estimados. Siguiendo estos criterios, se seleccionará aquel modelo para el que se obtenga un AIC más bajo.

AIC no proporciona una prueba de un modelo en el sentido de probar una hipótesis nula, es decir AIC no puede decir nada acerca de la calidad del modelo en un sentido absoluto. Si todos los modelos candidatos encajan mal, AIC no dará ningún aviso de ello.

Este criterio penaliza los modelos con muchos parámetros frente a los modelos más parsimoniosos. Con $p = \log(n)$ se obtiene un AIC modificado al que se le denomina BIC (Criterio de Información Bayesiano).

$$BIC = \log(n) \cdot (k + 1) - 2\log(L_M) \quad [171]$$

Estos estadísticos se usan en los procedimientos de selección de variables. La función R que implementa estos procedimientos se denomina *stepAIC*. Esta función permite variar el criterio de penalización eligiendo el valor de p .

Se muestra a continuación cómo calcular en R estos estadísticos y cómo ejecutar el procedimiento de selección de variables.

```
# Modelo Logit
paste ("AIC: ",AIC(est3))
[1] "AIC: 14817.4107366346"
paste ("BIC: ",BIC(est3))
[1] "BIC: 14997.8679170593"

# Modelo Probit
paste ("AIC: ",AIC(est4))
[1] "AIC: 14841.7599513679"
```

```
paste ("BIC: ",BIC(est4))
[1] "BIC: 15022.2171317926"
```

Aplicamos ahora el procedimiento de selección de variables al modelo Logit utilizando la función stepAIC.

En el argumento scope indicamos el modelo más complejo (en nuestro caso *est3*) y el más simple (donde se ha especificado el modelo nulo). Para especificar las interacciones en las fórmulas se utiliza el símbolo ":". Así, la expresión relativa a un modelo saturado con 3 factores y sus interacciones, sería la siguiente:

```
glm(y~x_1+x_2+x_3+x_1:x_2+x_1:x_3+x_2:x_3+x_1:x_2:x_3, data=dataframe,
family=binomial)
```

Otra opción equivalente, pero más cómoda, sería utilizar el operador "*", el cual incorpora todos los efectos involucrados junto con sus interacciones. Así, $x_1 * x_2 \equiv x_1 + x_2 + x_1:x_2$ o la expresión anterior, relativa a un modelo saturado con 3 factores, sería equivalente a:

```
glm(y~x_1*x_2*x_3, data=dataframe, family=binomial)
```

Con el argumento direction="both" indicamos que el procedimiento sea stepwise. La función stepAIC comprueba en cada paso como varía el AIC con la inclusión o supresión de variables. El procedimiento termina cuando la inclusión/supresión de variables no aumenta/disminuye el valor del AIC. También de dispone de los parámetros "forward" y "backward". Por último, por defecto se dispone del parámetro "k" con un valor por defecto de 2 (parámetro de penalización *p* en nuestra notación), es decir, estadístico AIC.

Dado que todas las variables con las que definimos el modelo eran significativas, el procedimiento incluye a todas ellas, obteniendo como resultado el propio modelo inicial.

```
modelo.stp <- stepAIC(est3, scope = list(upper = est3$formula, lower = ~1), direction
= "both")
Start: AIC=14817.41
datos$pobre ~ nmiemb + nmiem11 + numinacti + numocu + tiphogar1 +
situocuhog

      Df Deviance   AIC
<none>      14771 14817
- nmiem11    1    14788 14832
- numocu     1    14789 14833
- situocuhog 7    14927 14959
- tiphogar1  11   15206 15230
- numinacti  1    15466 15510
- nmiemb     1    15971 16015

modelo.stp

Call: glm(formula = datos$pobre ~ nmiemb + nmiem11 + numinacti + numocu +
tiphogar1 + situocuhog, family = binomial, data = datos)

Coefficients:
(Intercept)
-2.7081

nmiemb
1.3216

nmiem11
-0.1508
```

numinacti
-0.8813

numocu
-1.5816

tiphogar1Padre o madre solo, con al menos un hijo de 16 o más años
0.5893

tiphogar1Pareja con al menos un hijo de 16 o más años
0.5113

tiphogar1Pareja con dos hijos menores de 16 años
-0.3840

tiphogar1Pareja con tres o más hijos menores de 16 años
-0.6002

tiphogar1Pareja con un hijo menor de 16 años
0.2479

tiphogar1Pareja sin hijos teniendo al menos uno de los miembros 65 años o más
-0.4082

tiphogar1Pareja sin hijos teniendo los dos miembros menos de 65 años
0.2108

tiphogar1Un adulto con niños menores de 16 años
0.7952

tiphogar1Una persona de 30 a 64 años
0.6961

tiphogar1Una persona de 65 o más años
-1.0251

tiphogar1Una persona de menos de 30 años
1.5050

situocuhogEl sustentador principal o el cónyugeocupado, al menos otros dos miembros ocupados
-0.3353

situocuhogEl sustentador principal o el cónyugeocupado, ninguno de los otros miembros ocupado (si es que los hay)
0.1355

situocuhogEl sustentador principal y el cónyuge ocupados, ninguno de los otros miembros ocupados (si es que los hay)
-0.9875

situocuhogEl sustentador principal y el cónyugeocupados, al menos otro de los miembros también ocupado
-0.3769

situocuhogNi el sustentador principal ni su cónyuge ocupado, otro miembro ocupado
-0.6623

situocuhogNi el sustentador principal ni su cónyuge ocupados, al menos otros dos miembros ocupados
-0.8303

situocuhogNingún ocupado en el hogar
0.2852

```
Degrees of Freedom: 18881 Total (i.e. Null); 18859 Residual
(3264 observations deleted due to missingness)
Null Deviance: 20670
Residual Deviance: 14770 AIC: 14820
```

3.4. PRUEBA DE HOSMER-LEMESHAW

El uso de los contrastes basados en los estadísticos χ^2 o G^2 no resulta adecuado en el caso de datos no agrupados.

Un contraste alternativo es el propuesto por Hosmer y Lemeshow. Los importantes trabajos de Hosmer y Lemeshow (1980) concluyeron con la aportación de una serie de tests estadísticos para medir la bondad del ajuste basándose en la agrupación de las probabilidades estimadas por el modelo. Fueron fundamentalmente dos propuestas de estadísticos que llamaron C_g y H_g .

Hosmer *et al.* (1997) recomendaron el empleo de estos estadísticos después de utilizar otros.

La construcción del estadístico se basa en la agrupación de probabilidades estimadas bajo el modelo de regresión formando, en principio, diez grupos que se denominan deciles de riesgo y se calculan, para estos grupos, las frecuencias esperadas. Finalmente, se comparan las frecuencias de éxito observadas con las estimadas, mediante el estadístico usual χ^2 de Pearson.

Para la creación de los grupos hay que elegir los puntos de corte de las probabilidades estimadas. Existen dos formas:

- Dividir las probabilidades estimadas en intervalos de igual amplitud.
- Dividir los datos en base a los cuantiles de la distribución, obteniendo grupos más homogéneos.

En R se realiza a través de la función *hoslem.test* implementada en la librería *ResourceSelection*. Los tests para nuestros modelos Logit Y Probit sugieren la revisión de los mismos, obteniendo no obstante un resultado ligeramente mejor en el modelo Logit.

```
library(ResourceSelection)

# Hosmer-Lemeshaw con g=10 grupos
# Modelo Logit
h1.est3 <- hoslem.test(est3$y, fitted(est3), g=10)
h1.est3

  Hosmer and Lemeshow goodness of fit (GOF) test

  data: est3$y, fitted(est3)
  X-squared = 56.216, df = 8, p-value = 2.561e-09

# Modelo Probit
h1.est4 <- hoslem.test(est4$y, fitted(est4), g=10)
h1.est4

  Hosmer and Lemeshow goodness of fit (GOF) test

  data: est4$y, fitted(est4)
  X-squared = 113.42, df = 8, p-value < 2.2e-16
```

3.5. MEDIDAS TIPO R^2

En la regresión lineal por mínimos cuadrados tenemos una medida R^2 , que nos ofrece una medida de la bondad del ajuste comparando sumas de cuadrados la variabilidad explicada por el modelo y la total de la variable respuesta.

$$R^2 = 1 - \frac{\sum_i(y_i - \hat{y}_i)^2}{\sum_i(y_i - \bar{y}_i)^2} \quad [172]$$

Podríamos considerar esta medida en el caso de la regresión logística o el modelo Probit sustituyendo \hat{y}_i por $m_i \hat{p}_i$, pero esta medida presenta varios inconvenientes:

- Los estimadores máximo-verosímiles no son los estimadores que maximizan esta medida.
- No tiene en cuenta la dependencia de la varianza de Y respecto de p .

Para solventar estos problemas se han propuesto medidas alternativas, denominadas pseudo R^2 . Las principales son los pseudo R^2 de McFadden, Cox-Snell y Nagelkerke.

Estas medidas están implementadas en la función *nagelkerke* de la librería *rcompanion*.

3.5.1. Pseudo R^2 de McFadden (McFadden, 1974)

Esta medida compara la log-verosimilud del modelo ajustado con la log-verosimilud del modelo que solo tiene el término constante

$$R_{MF}^2 = 1 - \frac{\log(L_M)}{\log(L_{Null})} \quad [173]$$

Asemejándose a la expresión del R^2 en la estimación por mínimos cuadrados, donde $\log(L_{Null})$ sería la suma de cuadrados total y $\log(L_M)$ haría el papel de la suma de cuadrados de los errores. Este R^2 informa de en cuánto se reduce la devianza de los datos al ajustar el modelo.

Su rango teórico es el intervalo $[0, 1]$, si bien muy raramente su valor se aproxima a 1. Suele considerarse una buena calidad del ajuste valores comprendidos en $[0,2, 0,4]$ y excelente para valores superiores.

A partir del modelo, podemos calcularlo del siguiente modo.

```
estnull <- glm(datos$pobre~1, data = datos, binomial)
R2MFLogit <- 1 - logLik(est3)/logLik(estnull)
R2MFProbit <- 1 - logLik(est4)/logLik(estnull)
R2MFLogit
'log Lik.' 0.374235 (df=23)
R2MFProbit
'log Lik.' 0.3732035 (df=23)
```

3.5.2. Pseudo R^2 de Cox-Snell (Cox & Snell, 1989)

Este coeficiente se define como:

$$R_{CS}^2 = 1 - \frac{(\sqrt[n]{L_{Null}})^2}{(\sqrt[n]{L_M})^2} = 1 - \exp\left(\frac{\log(L_M) - \log(L_{Null})}{n}\right) \quad [174]$$

siendo $L_{Null} = \exp(-\log(L_{Null})/2)$ y $L_M = \exp(-\log(L_M)/2)$.

El rango de valores de este coeficiente es $[0, (\sqrt[n]{L_{NULL}})^2]$ lo que le hace poco interpretable al depender de L_{NULL} .

3.5.3. Pseudo R^2 de Nagelkerke (Nagelkerke, 1991)

El pseudo R^2 de Nagelkerke corrige el problema del límite superior del coeficiente estandarizando el anterior, es decir:

$$R_N^2 = \frac{R_{CS}^2}{1 - (\sqrt[n]{L_{NULL}})^2} \quad [174]$$

tomando valores comprendidos en el intervalo $[0, 1]$, por lo que puede interpretarse de igual modo que el coeficiente de determinación de la regresión lineal clásica, aunque es más difícil que alcance valores próximos a 1.

Se muestra a continuación el cálculo de los tres coeficientes en el programa R a través de la función *nagelkerke*.

```
library(rcompanion)
nagelkerke(est3, restrictNobs = TRUE)
$Models

  Model: "glm, datos$pobre ~ nmiemb + nmiem11 + numinacti + numocu + tiphogar1 +
situocuhog, binomial, datos"
  Null: "glm, datos$pobre ~ 1, binomial, fit$model"

$Pseudo.R.squared.for.model.vs.null
                               Pseudo.R.squared
McFadden                      0.374235
Cox and Snell (ML)            0.373653
Nagelkerke (Cragg and Uhler)  0.523662

$Likelihood.ratio.test
  Df.diff LogLik.diff Chisq p.value
    -22      -4417   8834      0

$Number.of.observations

  Model: 18882
  Null: 22146

$Messages
[1] "Note: For models fit with REML, these statistics are based on refitting with
ML"

$Warnings
[1] "WARNING: Fitted and null models have different numbers of observations"
nagelkerke(est4, restrictNobs = TRUE)
$Models

  Model: "glm, datos$pobre ~ nmiemb + nmiem11 + numinacti + numocu + tiphogar1 +
situocuhog, binomial(link = probit), datos"
  Null: "glm, datos$pobre ~ 1, binomial(link = probit), fit$model"

$Pseudo.R.squared.for.model.vs.null
                               Pseudo.R.squared
McFadden                      0.373204
Cox and Snell (ML)            0.372845
Nagelkerke (Cragg and Uhler)  0.522529

$Likelihood.ratio.test
  Df.diff LogLik.diff Chisq p.value
    -22      -4404.8 8809.6      0
```

```
$Number.of.observations
Model: 18882
Null: 22146

$Messages
[1] "Note: For models fit with REML, these statistics are based on refitting with
ML"

$Warnings
[1] "WARNING: Fitted and null models have different numbers of observations"
```

3.6. MÉTODOS ESPECÍFICOS PARA MODELOS DE CLASIFICACIÓN

A continuación, se exponen los métodos de evaluación más habituales utilizados en la clasificación y que se pueden agrupar en tres tipos:

- Métodos basados en métricas.
- Métodos basados en curvas ROC.
- Métodos que incorporan una matriz de costes.

3.6.1. *Métodos basados en métricas*

La evaluación de modelos de clasificación basados en métricas se realiza con los resultados de la matriz de confusión definiendo diferentes métricas en función de la disparidad entre los resultados obtenidos y los datos reales.

Una matriz de confusión estándar tiene la estructura de la tabla 17. Una primera métrica es el porcentaje de acierto definido como el número de casos acertados entre el número de casos totales. Esta es la métrica más sencilla y habitual, pero resulta muy engañosa la mayor parte de las veces, pues no informa nada acerca de la distribución del error entre clases. Además, siempre que apliquemos esta métrica debemos tener en cuenta el balanceo entre clases, ya que, si tenemos una población en la que el 90% de los individuos son de clase A y el 10% restante de clase B, cualquier algoritmo que obtenga un porcentaje de acierto inferior al 90% en la clase A se entiende que no aporta ningún tipo de conocimiento en la clasificación.

Tabla 17. Matriz de confusión.

		Clase clasificada como:		
		A+ (SI)	A- (NO)	Total
Estado real	A+ (SI)	Verdaderos positivos (VP) $FVP = \frac{VP}{TCP}$	Falsos negativos (FN) $FFN = \frac{FN}{TCP}$	1
	A- (SI)	Falsos positivos (FP) $FFP = \frac{FP}{TCA}$	Verdaderos negativos (TN) $FVN = \frac{VN}{TCA}$	

La precisión o exactitud de un clasificador es el cociente entre el número de ejemplos que están bien clasificados, que se corresponde en la matriz de confusión con la suma de los elementos de la diagonal, entre el total de instancias.

La precisión también se puede definir para cada clase como el número de los casos de esa clase clasificados correctamente dividido por el número total de casos clasificados de esa misma clase. Según el ejemplo de la tabla 17, para la clase SI la precisión correspondería con el cociente de los ejemplos clasificados correctamente (verdaderos positivos) entre todos los elementos clasificados de esa clase (verdaderos positivos + falsos positivos).

La cobertura o recall es otra medida muy usada y se define como el número de casos de esa clase predichos correctamente entre el número de casos de esa clase que existen en la base de datos. Por ejemplo, para calcular la cobertura de la clase SI se divide el número de casos correctamente clasificados de SI (verdaderos positivos) por la suma de los verdaderos positivos más los falsos negativos (que realmente corresponden con los positivos de la base de datos que han sido clasificados como negativos).

La precisión habla de la “bondad” de la predicción de una clase, mientras que la cobertura indica la capacidad del modelo para identificar los casos correctos de dicha clase. Como ambas medidas son importantes, existe una métrica que combina mediante una media armónica ambas y que se denomina F-Measure. En algunas disciplinas científicas, son mucho más utilizados los conceptos de “sensibilidad” y “especificidad”. La sensibilidad (S) es la probabilidad de clasificar correctamente a una instancia cuyo estado real sea la presencia de la condición de interés. Este valor también es conocido como la fracción de verdaderos positivos (FVP), recall o exactitud positiva.

La especificidad (E) se define como la probabilidad de clasificar correctamente a un individuo cuyo estado real sea la ausencia de la condición. Es la proporción de respuestas negativas que son correctamente clasificadas.

Existen otras medidas de exactitud como el índice de Youden que se calcula a través de las diferencias entre las proporciones de respuestas positivas correctas e incorrectas. La tasa o razón de verosimilitud (LR) es otra medida muy utilizada como forma de discriminar los clasificadores. Esta tasa de verosimilitud representa el grado de evidencia de una respuesta del clasificador a favor de la presencia de la condición con respecto a la ausencia de la condición.

Otro índice muy utilizado es el estadístico Kappa. Es un coeficiente estadístico que determina la precisión del modelo a la hora de predecir la clase verdadera. Este estadístico está ampliamente difundido.

3.6.2. Métodos basados en la curva ROC

La evaluación de modelos de clasificación basados en curvas ROC (Receiver Operating Characteristic) representan de forma gráfica el rendimiento de un clasificador que muestra la distribución de las fracciones de verdaderos positivos y la fracción de falsos negativos. La curva ROC nos proporciona una herramienta visual para examinar la capacidad que dispone un clasificador para detectar correctamente a los individuos con presencia de la condición de interés en el análisis y su incapacidad para identificar los individuos del grupo de ausencia.

Swets y Pickett (1982) señalan tres importantes propiedades de las curvas ROC:

- Representan un índice de exactitud intrínseco: es un índice de la capacidad del clasificador para discriminar estados y seleccionar el estado correcto, independientemente del criterio de selección. Son curvas que reflejan las probabilidades subjetivas junto con las utilidades que usualmente determina este criterio, por lo que podemos decir que es un índice del criterio de decisión.
- Utilizan las probabilidades a priori de los posibles estados de forma más precisa, y también los costes de las decisiones correctas y equivocadas para determinar el criterio óptimo de decisión de un clasificador.
- En tercer lugar, las curvas ROC contienen las estimaciones de las probabilidades de los distintos tipos de resultados para todos y cada uno de los criterios de decisión.

La curva ROC es el patrón de oro en muchas áreas de análisis de modelos, ya que representan de forma compacta muchísima información del rendimiento de un clasificador. Muchos autores consideran a las curvas ROC como herramientas fundamentales en la fase de evaluación de un modelo, entre ellos Zweig y Campbell (1993).

El área bajo la curva ROC (AUC) es equivalente al valor del estadístico suma de rangos de Wilcoxon, tal y como señalan Bamber (1988) y Hanley y McNeil (1982), lo que permite trasladar las propiedades del estadístico de Wilcoxon a las medidas de exactitud global del AUC. También se interpreta como un promedio de la sensibilidad para todos los valores de especificidad y, de forma análoga, como un promedio de la especificidad para todos los posibles valores de sensibilidad. Breiman *et al.* (1984) relacionan el AUC con el coeficiente de Gini. Para que un modelo aporte información significativa, el valor de su AUC ha de ser estadísticamente superior a su valor mínimo 0,5, lo cual puede contrastarse mediante el correspondiente test estadístico. También es posible contrastar el valor de este estadístico frente a su valor máximo 1, o incluso las diferencias de AUC de dos modelos distintos, de cara a determinar cuál de ellos es el que mejor predice.

3.6.3. Métodos basados en una matriz de costes

La evaluación de modelos de clasificación basados en costes es otra alternativa disponible para comparar modelos. Estos métodos se basan en establecer una matriz de costes asociados a la clasificación.

La mayoría de los métodos estadísticos y algoritmos de aprendizaje, por su propia naturaleza, buscan minimizar el número de errores del clasificador generado. Cuando utilizamos el porcentaje de acierto o de error para evaluar el desempeño de nuestros modelos de clasificación, estamos suponiendo que ambos tipos de errores son equivalentes.

Sin embargo, son múltiples los problemas de Aprendizaje Automático en los que los errores cometidos por el clasificador no tienen la misma importancia, Provost y Fawcett (2001).

Así, los factores de riesgo de los modelos de credit scoring, es decir, los factores que están detrás de los errores tipo I (admitir como sana una operación insolvente) y tipo II (rechazar como insolvente una operación sana) no son los mismos. No es igual, en términos de coste económico, clasificar a un cliente como bueno, concederle el crédito y que luego no nos lo devuelva que no conceder el crédito a una persona que es cliente. En el primer caso estamos expuestos a un caso de riesgo de crédito y, en el otro caso, incurrimos en un coste de oportunidad por la pérdida potencial de buenos clientes.

Una función de coste esperado es aquella que pondera el porcentaje de los que devuelven el crédito y los que no por sus respectivos costes. Si llamamos C_e al coste esperado, la función es la siguiente:

$$C_e = C_{12}\pi_2 \frac{n_2}{N_2} + C_{21}\pi_1 \frac{n_1}{N_1}$$
[175]

Donde $\frac{n_2}{N_2}$ y $\frac{n_1}{N_1}$ es la proporción de buenos y malos pagadores, C_{12} y C_{21} son los costes asociados a los errores de tipo I y II y π_1 y π_2 son las probabilidades a priori de buenos y malos riesgos.

En un problema como el de credit scoring, la complejidad existente en el cálculo de los costes asociados a los dos tipos de errores es considerable, dado que los factores que los afectan son difíciles de cuantificar.

A los costes asociados a la pérdida del monto del crédito otorgado (C_{12}) hay que restarle los ingresos recibidos antes de pasar a la situación de moroso u otros ingresos recibidos por valores de la propiedad asegurada en el momento de la liquidación, y sumar aquellos gastos que se deriven de costes legales, costes administrativos, etcétera.

Los costes asociados al error de tipo II (C_{21}) están asociados a la pérdida de los intereses que se generarían si se hubiera concedido el préstamo al buen pagador más la pérdida o beneficio de destinar este crédito no concedido a otro cliente. Estos costes se pueden llamar coste de oportunidad. Por otra parte, si el solicitante del crédito es un cliente del banco y no se le concede, muy probablemente deje de ser cliente. Si el peticionario del crédito no es cliente y no se le concede el crédito, casi con toda seguridad ese demandante no llegue a ser cliente de la entidad financiera a quien ha dirigido su solicitud de dinero y, desde un punto de vista más práctico, para cuantificar estos costes deberíamos contar con información de todos los productos financieros que dejaría de consumir a lo largo del ciclo de vida del cliente. Estos costos, es muy probable que cambien con el tiempo, por lo que se puede concluir que, aunque se pueda establecer unos rangos entre los que probablemente estén, es prácticamente improbable el cálculo exacto.

Los métodos de aprendizaje sensibles al coste suelen ser adaptaciones de algoritmos existentes, como los árboles de decisión (Ting, 1998). Sin embargo, existen estrategias que son independientes del algoritmo de aprendizaje utilizado. Estas estrategias, corrientemente denominadas meta-esquemas de aprendizaje, toman como entrada un algoritmo de aprendizaje, una colección de datos de entrenamiento y una distribución de costes, y generan un clasificador.

Para los modelos con costes asimétricos, podemos encontrar diferentes estrategias para abordar una correcta clasificación:

- *Basadas en un umbral.* Witten y Frank (1999) proponen un modelo aplicable a todo algoritmo cuya salida sea un clasificador que emite valores numéricos (como probabilidades, similitudes, etc.).
- *Ponderando las instancias modificando los pesos asignados a cada clase,* de manera que se les da más peso a los ejemplos asociados a cometer errores más costosos. Ting, (1998) sugiere dar pesos a cada cliente (*Instance Weighting*) pero un peso mayor a los individuos de una clase (por ejemplo, a la clase de los que no devuelven

el crédito), con el objetivo de que el algoritmo se fije especialmente en clasificar correctamente estos ejemplares, minimizando el error sobre ellos. También se encuentra dentro de este ámbito el algoritmo MetaCost de Domingos (1999) que tiene como objetivo reetiquetar cada muestra de entrenamiento por la estimación del riesgo de Bayes. Finalmente, el clasificador se entrena con un método no basado en costes con el conjunto que ya ha sido reetiquetado. Este método es aplicable a cualquier algoritmo de aprendizaje.

Algunas aportaciones interesantes en los métodos de clasificación a través de los costes las podemos encontrar en Ling *et al.* (2004). Estos autores emplean árboles de decisión con costes mínimos. La idea que subyace es introducir un factor de costes mientras se va construyendo el árbol de acuerdo con los criterios de división que minimizan el coste total, en lugar de minimizar la entropía. En este sentido, los árboles de decisión con costes mínimos y MetaCost son similares, aunque hay una gran diferencia: en los árboles de decisión con un coste mínimo la parte más sensible a los costes se construye directamente en el clasificador, mientras que el algoritmo MetaCost puede utilizar no solo los árboles de clasificación, sino cualquier método de clasificación: redes neuronales, máquinas de vectores soporte, redes bayesianas, etcétera.

Otro enfoque importante es el de López *et al.* (2010) que utilizan reglas difusas para problemas de bases de datos no balanceados. Desde esta perspectiva, el aprendizaje sensible al coste en las bases de datos que analizan alcanza un buen equilibrio entre las clases, mejorando la clase positiva (sensibilidad) y no perjudicando la precisión de la clase considerada negativa (especificidad).

TEMA 4: MÉTODOS ESTADÍSTICOS DE REDUCCIÓN DE DIMENSIONES

ÍNDICE

1. Análisis Factorial y Componentes Principales

- 1.1. Introducción
- 1.2. Análisis Factorial vs Componentes Principales
- 1.3. Análisis de Componentes Principales
- 1.4. Análisis Factorial

2. Análisis de Correspondencias

- 2.1. Introducción
- 2.2. Objetivo
- 2.3. Análisis de Correspondencias Simple
- 2.4. Análisis de Correspondencias Múltiple
- 2.5. Ejemplo de Análisis de Correspondencias Simple con el software R
- 2.6. Ejemplo de Análisis de Correspondencias Múltiple en R

OBJETIVOS	PALABRAS CLAVE
<ul style="list-style-type: none">· Reducir la dimensionalidad de un conjunto de variables cuantitativas a partir del Análisis de Componentes Principales.· Estudio de los factores comunes y específicos de un conjunto de variables cuantitativas a partir del Análisis Factorial.· Reducir la dimensionalidad y estudiar las asociaciones entre los atributos de un conjunto de variables cualitativas a través del Análisis de Correspondencias.	<ul style="list-style-type: none">· Factor, componente principal.· Inercia, comunalidad.· Rotación de los factores· Dimensiones, puntos fila, puntos columna.

1. ANÁLISIS FACTORIAL Y COMPONENTES PRINCIPALES

1.1. INTRODUCCIÓN

En muchas ocasiones, nos encontramos con un número muy grande de variables para medir una determinada realidad. El Análisis Factorial es una técnica de reducción de datos, es decir, pretende pasar de ese número tan elevado de variables, a un número más pequeño, de tal manera que podamos explicar de una manera más sencilla dicha realidad. A esas nuevas variables se les denomina *factores o componentes*.

Es evidente que estas nuevas variables tendrán que obtenerse e interpretarse a partir de las variables iniciales, y también es cierto que el modelo perderá poder explicativo cuanto menor sea el número de factores construidos. El objetivo será, por tanto, obtener la mayor facilidad posible en la interpretación de la realidad que estemos estudiando al menor coste posible en términos de pérdida de información.

Existen dos tipos de análisis factorial: el *exploratorio* y el *confirmatorio*. El análisis exploratorio se caracteriza porque no se conoce a priori el número de factores, y es en la aplicación empírica donde se determina este número. Por el contrario, en el análisis de tipo confirmatorio, los factores están fijados a priori y, por tanto, el análisis se encamina a corroborar su existencia.

Los antecedentes del Análisis Factorial se encuentran en las técnicas de regresión lineal, iniciadas por Galton. Un continuador suyo fue K. Pearson (1901), que presentó la primera propuesta del "*método de componentes principales*", primer paso para el cálculo del Análisis Factorial.

El origen del Análisis Factorial suele atribuirse a Spearman (1904), en su clásico trabajo sobre inteligencia, donde distingue un factor general (factor G) y cierto número de factores específicos.

Hotelling (1933), desarrolló un método de extracción de factores sobre la técnica de "componentes principales".

Thurstone (1947), expresó la relación entre las correlaciones y las saturaciones de las variables en los factores. Introdujo el concepto de *estructura simple*. También desarrolló la teoría y método de las rotaciones factoriales para obtener la estructura factorial más sencilla. En un principio las rotaciones eran gráficas.

Kaiser (1958) desarrolló el método *Varimax* para realizar rotaciones ortogonales mediante procedimientos matemáticos.

A lo largo del desarrollo histórico del Análisis Factorial se han planteado algunos problemas de fondo que han dado lugar a distintas propuestas de solución. Los aspectos más polémicos han sido:

- a) La estimación de las comunidades.
- b) Los métodos de extracción de factores.
- c) El número de factores a extraer.
- d) Los métodos de rotación de factores.

Por ejemplo, se han propuesto múltiples métodos para la extracción de factores, comprobándose que había distintas soluciones a un mismo problema, según el método que

se adoptase. Con esto se plantea el dilema de qué método elegir. Las respuestas han sido distintas según las diversas tendencias. El método de Componentes Principales suele ser el más utilizado. De todas formas, hay autores que consideran que el Análisis de Componentes Principales es distinto del Análisis Factorial.

1.2. ANÁLISIS FACTORIAL VS COMPONENTES PRINCIPALES

El Análisis Factorial y el Análisis de Componentes Principales están muy relacionados. Algunos autores consideran el segundo como una etapa del primero y otros los consideran como técnicas diferentes.

El Análisis de Componentes Principales trata de hallar componentes (factores) que sucesivamente expliquen la mayor parte de la varianza total. Este tipo de análisis persigue la reducción de la dimensión de una tabla de datos mediante el cálculo de variables que son “perfectamente calculables” y que son combinaciones lineales de las originales. Estas nuevas variables (factores o componentes) contienen la mayor parte de información de los datos originales.

Por su parte el Análisis Factorial busca factores que expliquen la mayor parte de la varianza común. En este caso se calculan nuevas “variables ficticias” que, aunque no observables, son combinación lineal de las reales y recogen la mayor parte de información correspondiente a las primeras.

En el Análisis Factorial se distingue entre varianza común y varianza única.

- ✓ La *varianza común* es la parte de la variación de la variable que es compartida con las otras variables.
- ✓ La *varianza única* es la parte de la variación de la variable que es propia de esa variable.

El Análisis de Componentes Principales no hace esa distinción entre los dos tipos de varianza, se centra en la varianza total.

Mientras que el Análisis de Componentes Principales busca hallar combinaciones lineales de las variables originales que expliquen la mayor parte de la variación total, el Análisis Factorial pretende hallar un nuevo conjunto de variables, menor en número que las variables originales, que exprese lo que es común a esas variables. El Análisis Factorial supone que existe un factor común subyacente a todas las variables, el Análisis de Componentes Principales no hace tal asunción.

En el Análisis de Componentes Principales, el primer factor o componente sería aquel que explica una mayor parte de la varianza total, el segundo factor sería aquel que explica la mayor parte de la varianza restante, es decir, de la que no explicaba el primero, y así sucesivamente. De este modo, sería posible obtener tantos componentes como variables originales, aunque esto en la práctica no tiene sentido.

En resumen, tenemos dos grandes tendencias:

- a) Análisis de Componentes Principales.
- b) Análisis factorial, dentro del cual existen diferentes métodos de extracción de los factores, como el método de los componentes principales, el método de máxima verosimilitud, ejes principales, etc.

1.3. ANÁLISIS DE COMPONENTES PRINCIPALES

El Análisis de Componentes Principales (ACP) es una técnica estadística de síntesis de la información, o reducción de la dimensión (número de variables).

Dadas n observaciones de p variables, se buscan $m < p$ variables que sean combinaciones lineales de las p originales y que estén incorreladas, recogiendo la mayor parte de la información o variabilidad de los datos. Hay que tener presente que, si las variables originales están incorreladas de partida, entonces no tiene sentido realizar un análisis de componentes principales.

Se considera una serie de variables (x_1, x_2, \dots, x_p) sobre un grupo de objetos o individuos y se trata de calcular, a partir de ellas, un nuevo conjunto de variables (y_1, y_2, \dots, y_p) , incorreladas entre sí, cuyas varianzas vayan decreciendo progresivamente. Cada y_j (donde $j = 1, \dots, p$) es una combinación lineal de las x_1, x_2, \dots, x_p originales, es decir:

$$y_j = a_{j1}x_1 + a_{j2}x_2 + \dots + a_{jp}x_p = a'_j x \quad [1]$$

donde $a'_j = (a_{j1}, a_{j2}, \dots, a_{jp})$ y

$$x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix} \quad [2]$$

El objeto del análisis es maximizar la varianza del componente obtenido, pero manteniendo la ortogonalidad de la transformación, esto es:

$$a'_j a_j = \sum_{k=1}^p a_{kj}^2 = 1 \quad [3]$$

La varianza del primer componente, por tanto, será:

$$Var(y_1) = Var(a'_1 x) = a'_1 \sigma a_1 \quad [4]$$

El problema consiste en maximizar la función $a'_1 \sigma a_1$ donde σ es la matriz de covarianzas de las p variables, sujeta a la restricción $a'_1 a_1 = 1$. Para resolver dicho problema, se Utilizan los multiplicadores de Lagrange, siendo por tanto la función objetivo:

$$L(a_1) = a'_1 \sigma a_1 - \lambda(a'_1 a_1 - 1) \quad [5]$$

A continuación, se busca el maximo derivando según la incognita a_1 :

$$\frac{\partial L(a_1)}{\partial a_1} = 0 \rightarrow (\sigma - \lambda I)a_1 = 0 \quad [6]$$

La solución del sistema de ecuaciones requiere que el determinante $|\sigma - \lambda I|$ sea no nulo, de manera que λ es un autovalor de σ , que al ser definida positiva, tendrá p autovalores distintos.

Entonces, para maximizar la varianza de y_1 se tiene que tomar el mayor autovalor, λ_1 y el correspondiente autovector será a_1 .

Dado que $\sigma a_1 = \lambda a_1$, si multiplicamos a ambos lados por a'_1 :

$$a'_1 \sigma a_1 = \lambda a'_1 a_1 = \lambda \quad [7]$$

Es decir, λ será igual a la varianza de y_1 .

El segundo componente principal, $y_2 = a'_2 x$ ha de estar incorrelacionado con y_1 , es decir:

$$\text{cov}(y_1, y_2) = \text{cov}(a'_1 x, a'_2 x) = a'_1 \sigma a_2 = 0 \quad [8]$$

Para maximizar la $\text{Var}(y_2) = a'_2 \sigma a_2$ se imponen ahora dos restricciones:

$$a'_2 a_2 = 1 \quad [9]$$

$$a'_2 a_1 = 0 \quad [10]$$

El objetivo que se plantea es maximizar la suma de las varianzas de $y_1 = a'_1 x$ y $y_2 = a'_2 x$, donde a_1 y a_2 son los vectores que definen el plano. La función objetivo será por tanto:

$$L(a_1, a_2) = a'_1 \sigma a_1 + a'_2 \sigma a_2 - \lambda_1(a'_1 a_1 - 1) - \lambda_2(a'_2 a_2 - 1) \quad [11]$$

Derivando respecto a las incógnitas, a_1 y a_2 , e igualando a cero:

$$\frac{\partial L(a_1, a_2)}{\partial a_1} = 0 \rightarrow (\sigma - \lambda_1 I)a_1 = 0 \quad [12]$$

$$\frac{\partial L(a_1, a_2)}{\partial a_2} = 0 \rightarrow (\sigma - \lambda_2 I)a_2 = 0 \quad [13]$$

que indica que a_1 y a_2 deben ser vectores propios de σ .

Es claro que λ_1 y λ_2 deben ser los dos autovalores mayores de la matriz σ y a_1 y a_2 sus correspondientes autovectores. Observemos que la covarianza entre y_1 y y_2 , dada por $a'_1 \sigma a_2$ es cero, ya que $a'_1 a_2 = 0$, y las variables y_1 y y_2 estarán incorreladas.

Puede demostrarse análogamente que el espacio de dimensión m que mejor representa a los puntos viene definido por los vectores propios asociados a los m mayores valores propios de σ . Estas direcciones se denominan *direcciones principales* de los datos y a las nuevas variables por ellas definidas *componentes principales*.

Entonces todos los p componentes de y se pueden expresar como el producto de una matriz formada por los autovectores de σ , multiplicada por el vector x que contiene las variables originales:

$$y = Ax \quad [14]$$

donde $A = \begin{bmatrix} a'_1 \\ a'_2 \\ \vdots \\ a'_p \end{bmatrix}$ y la $\text{Var}(y_j) = \lambda_j$ y la $\text{Cov}(y_j, y_s) = 0$.

Calcular los componentes principales equivale, por tanto, a aplicar una transformación ortogonal A a las variables x (ejes originales) para obtener unas nuevas variables y incorreladas entre sí. Esta operación puede interpretarse como elegir unos nuevos ejes coordinados, que coincidan con los “ejes naturales” de los datos.

Si sumamos todos los autovalores, tendremos entonces la varianza total de los componentes, que por las propiedades del operador traza, acaba siendo igual que la varianza de las variables originales, es decir:

$$\sum_{j=1}^p \text{Var}(y_j) = \sum_{j=1}^p (\lambda_j) = \sum_{j=1}^p \text{Var}(x_j) \quad [15]$$

El porcentaje de varianza total que recoge cada componente principal, $\frac{\lambda_j}{\sum_{j=1}^p(\lambda_j)}$ es el criterio utilizado para reducir las p variables de nuestro conjunto de datos, por m combinaciones lineales de dichas variables.

El ACP se emplea sobre todo en análisis exploratorio de datos (Analisis factorial) y para construir modelos predictivos cuando las variables explicativas tienen problemas de colinealidad.

El ACP comporta el cálculo de la descomposición en autovalores de la matriz de covarianza. Normalmente, las variables originales se tipifican primero, lo cual implica el trabajar con la matriz de correlación, y dado que $Var(x_j) = 1 \forall j$:

$$\sum_{j=1}^p Var(y_j) = \sum_{j=1}^p (\lambda_j) = \sum_{j=1}^p Var(x_j) = p \quad [17]$$

Las covarianzas entre cada componente principal y las variables x vienen dadas por el producto de las coordenadas del vector propio que define el componente por su valor propio:

$$Cov(y_i, x) = Cov(a'_i x, x_1, \dots, x_p) = a'_i \lambda_i = (a_{i1} \lambda_i, a_{i2} \lambda_i, \dots, a_{ip} \lambda_i) \quad [18]$$

y, por tanto, las correlaciones:

$$Corr(y_i, x_j) = \frac{Cov(y_i, x_j)}{\sqrt{Var(y_i)Var(x_j)}} = a_{ij} \sqrt{\frac{\lambda_i}{Var(x_j)}} \quad [19]$$

o, en términos matriciales:

$$Corr(y, x) = \Lambda^{1/2} AD \quad [20]$$

donde $\Lambda^{1/2}$ es la matriz diagonal con términos $\sqrt{\lambda_i}$ y D es también diagonal, registrando las inversas de las desviaciones típicas de cada variable.

Las m componentes principales ($m < p$) proporcionan la predicción lineal óptima con m variables del conjunto de variables x .

Si estandarizamos los componentes principales, dividiendo cada uno por su desviación típica, se obtiene la estandarización multivariante de los datos originales. Si trabajamos con las variables originales tipificadas, entonces $Corr(y_i, x_j) = a_{ij}$.

Por tanto, la transformación mediante componentes principales conduce a variables incorreladas, pero con distinta varianza. Puede interpretarse como rotar los ejes de la elipse que definen los puntos para que coincidan con sus ejes naturales. La estandarización multivariante produce variables incorreladas con varianza unidad, lo que supone buscar los ejes naturales y luego estandarizarlos. En consecuencia, si estandarizamos los componentes se obtienen las variables estandarizadas de forma multivariante.

Cuando las escalas de medida de las variables son muy distintas, la maximización de las funciones objetivo dependerá decisivamente de estas escalas de medida y las variables con valores más grandes tendrán más peso en el análisis. Si queremos evitar este problema, conviene estandarizar las variables antes de calcular los componentes, de manera que las magnitudes de los valores numéricos de las variables x sean similares. Por otra parte, si las variabilidades de las x son muy distintas, las variables con mayor varianza van a influir más

en la determinación de la primera componente, siendo éste otro problema que evita la estandarización.

Por tanto, cuando las variables x originales están en distintas unidades conviene aplicar el análisis de la matriz de correlaciones o análisis normado. Cuando las variables tienen las mismas unidades, ambas alternativas son posibles. Si las diferencias entre las varianzas de las variables son informativas y queremos tenerlas en cuenta en el análisis, no debemos estandarizar las variables: por ejemplo, supongamos que tenemos dos índices con la misma base pero uno fluctúa mucho y el otro es casi constante. Este hecho es informativo, y para tenerlo en cuenta no se deben estandarizar las variables, de manera que el índice de mayor variabilidad tenga más peso. Por el contrario, si las diferencias de variabilidad no son relevantes se eliminan con el análisis normado. En caso de duda, conviene realizar ambos análisis, y seleccionar aquel que conduzca a conclusiones más informativas.

El análisis de componentes principales (PCA) puede realizarse con una función de la librería básica de «R»: *prcomp* o *princomp*.

Para nuestro ejemplo, partimos del dataframe *USAArrests* que contiene información sobre el número de arrestos por cada 100.000 residentes, por asalto, asesinato y violación en cada uno de los 50 estados de los Estados Unidos en 1973. También se da el porcentaje de la población que vive en áreas urbanas. Las variables, en concreto, serían:

- ✓ Murder. Arrestos por asesinato (por 100.000 hab.)
- ✓ Assault. Arrestos por asalto (por 100.000 hab.)
- ✓ UrbanPop. Porcentaje de población urbana
- ✓ Rape. Arrestos por violación (por 100.000 hab.)

El análisis realizado sería el siguiente:

```
require(graphics)
str(USAArrests)
'data.frame':   50 obs. of  4 variables:
 $ Murder : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
 $ Assault : int  236 263 294 190 276 204 110 238 335 211 ...
 $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
 $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...

# scale = TRUE para análisis normado
prcomp(USAArrests, scale = TRUE)
  Standard deviations (1, ..., p=4):
 [1] 1.5748783 0.9948694 0.5971291 0.4164494

  Rotation (n x k) = (4 x 4):
            PC1        PC2        PC3        PC4
Murder -0.5358995 0.4181809 -0.3412327 0.64922780
Assault -0.5831836 0.1879856 -0.2681484 -0.74340748
UrbanPop -0.2781909 -0.8728062 -0.3780158 0.13387773
Rape    -0.5434321 -0.1673186 0.8177779 0.08902432

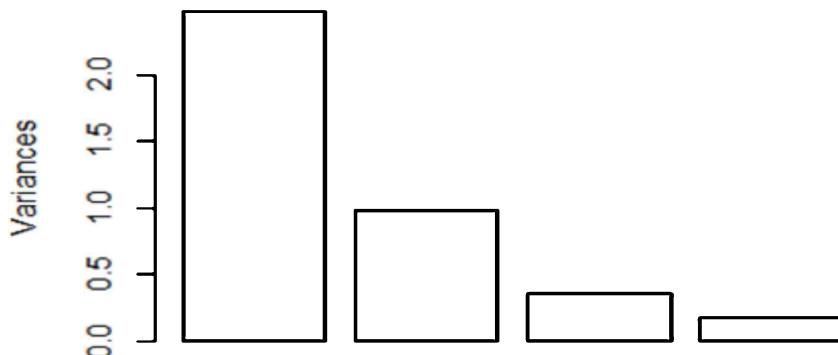
# Sin porcentaje de población urbana. La fórmula se establece sin
variable dependiente
prcomp(~ Murder + Assault + Rape, data = USAArrests, scale = TRUE)
  Standard deviations (1, ..., p=3):
 [1] 1.5357670 0.6767949 0.4282154

  Rotation (n x k) = (3 x 3):
```

	PC1	PC2	PC3
Murder	-0.5826006	0.5339532	-0.6127565
Assault	-0.6079818	0.2140236	0.7645600
Rape	-0.5393836	-0.8179779	-0.1999436

```
plot(prcomp(USArrests, scale = TRUE))
```

prcomp(USArrests, scale = TRUE)

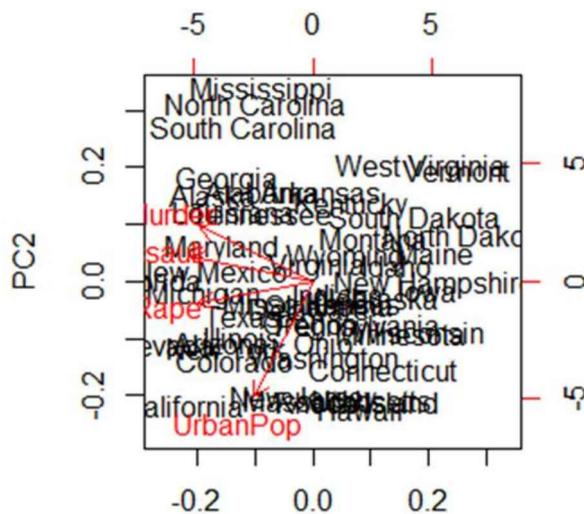


```
summary(prcomp(USArrests, scale = TRUE))
```

Importance of components:

	PC1	PC2	PC3	PC4
Standard deviation	1.5749	0.9949	0.59713	0.41645
Proportion of Variance	0.6201	0.2474	0.08914	0.04336
Cumulative Proportion	0.6201	0.8675	0.95664	1.00000

```
biplot(prcomp(USArrests, scale = TRUE))
```



Si no tipificamos las variables, la variable con mayor varianza (arrestos por asalto) será la que recoja la primera componente, acumulando el mayor peso en el análisis, frente a los arrestos por asesinato.

```
prcomp(USArrests)
```

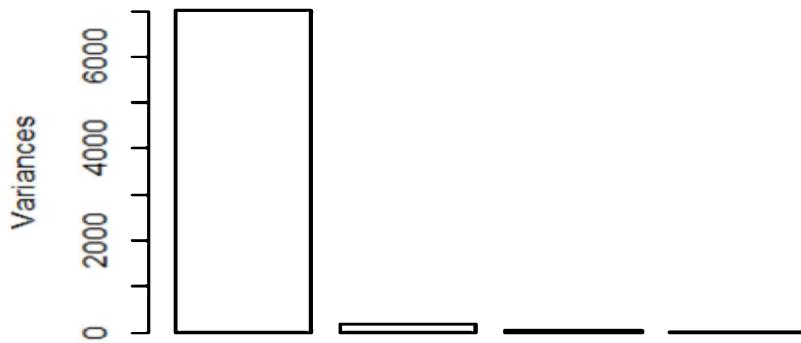
Standard deviations (1, ..., p=4):
[1] 83.732400 14.212402 6.489426 2.482790

Rotation (n x k) = (4 x 4):

	PC1	PC2	PC3	PC4
Murder	0.04170432	-0.04482166	0.07989066	-0.99492173
Assault	0.99522128	-0.05876003	-0.06756974	0.03893830

```
UrbanPop 0.04633575 0.97685748 -0.20054629 -0.05816914
Rape      0.07515550 0.20071807  0.97408059  0.07232502
```

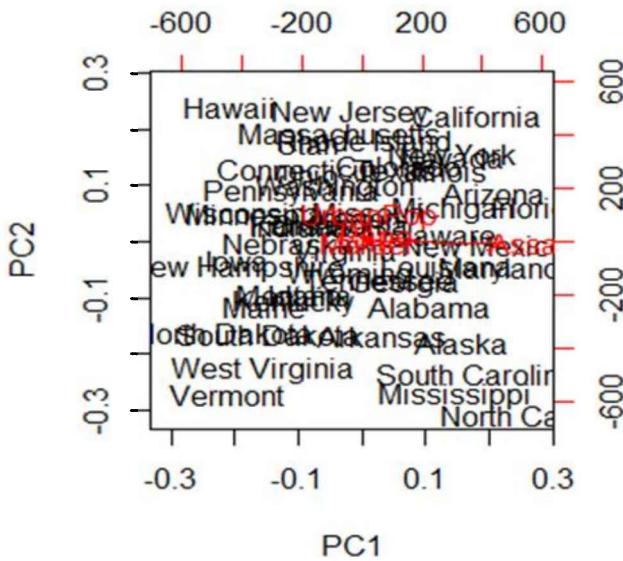
scale = FALSE es el valor por defecto
`plot(prcomp(USArrests))`

prcomp(USArrests)

```
summary(prcomp(USArrests))
Importance of components:
```

	PC1	PC2	PC3	PC4
Standard deviation	83.7324	14.21240	6.4894	2.48279
Proportion of Variance	0.9655	0.02782	0.0058	0.00085
Cumulative Proportion	0.9655	0.99335	0.9991	1.00000

`biplot(prcomp(USArrests))`



1.4. ANÁLISIS FACTORIAL

El análisis factorial (AF) es también una técnica estadística de reducción de datos usada para explicar las correlaciones entre las variables observadas en términos de un número menor de variables no observadas llamadas factores.

El análisis factorial exploratorio, se utiliza para tratar de descubrir la estructura interna de un número relativamente grande de variables. La hipótesis a priori del investigador es que pueden existir una serie de factores asociados a grupos de variables. Las cargas de los distintos factores se utilizan para intuir la relación de éstos con las distintas variables. Es el tipo de análisis factorial más común.

El análisis factorial confirmatorio, AFC, trata de determinar si el número de factores obtenidos y sus cargas se corresponden con los que cabría esperar a la luz de una teoría previa acerca de los datos. La hipótesis a priori es que existen unos determinados factores preestablecidos y que cada uno de ellos está asociado con un determinado subconjunto de las variables. El análisis factorial confirmatorio entonces arroja un nivel de confianza para poder aceptar o rechazar dicha hipótesis.

El análisis de componentes principales es el método apropiado de extracción de factores, cuando el interés primordial se centra en la predicción o el número mínimo de factores necesarios para justificar la porción máxima de varianza representada en la serie de variables original, y cuando el conocimiento previo sugiere que la varianza específica y de error representan una porción relativamente pequeña de la varianza total.

Por el contrario, cuando se pretende identificar las dimensiones latentes o las construcciones representadas en las variables originales y se tiene poco conocimiento de la varianza específica y el error, lo más apropiado es utilizar el método factorial común, si bien las complicaciones del análisis factorial común han contribuido al análisis generalizado de la técnica de componentes principales.

Para interpretar bien los factores se utiliza una rotación de ejes, ya que las soluciones factoriales no rotadas extraen factores según su orden de importancia. El primer factor tiende a ser un factor general por el que casi toda variable se ve afectada significativamente, dando cuenta del mayor porcentaje de varianza. Los métodos de rotación ortogonales más utilizados son VARIMAX, QUARTIMAX y EQUIMAX.

En la práctica no se utiliza un criterio único de los factores a extraer, si bien se utiliza como primera aproximación el gráfico de autovalor para el criterio de contraste de caída. Después de interpretar los factores extraídos hay que valorar su carácter práctico.

1.4.1. Planteamiento

Consideramos las variables observadas $X_1, X_2, X_3, \dots, X_p$ como variables tipificadas (con media cero y varianza unidad) y vamos a formalizar la relación entre las variables definiendo el siguiente modelo factorial:

$$\begin{bmatrix} X_1 = l_{11}F_1 + l_{12}F_2 + \dots + l_{1K}F_K + u_1 \\ X_2 = l_{21}F_1 + l_{22}F_2 + \dots + l_{2K}F_K + u_2 \\ \vdots \\ X_p = l_{p1}F_1 + l_{p2}F_2 + \dots + l_{pK}F_K + u_p \end{bmatrix} \quad [21]$$

En este modelo F_1, F_2, \dots, F_k son los factores comunes, u_1, u_2, \dots, u_p son los factores únicos o unicidades y l_{ij} es el peso del factor j en la variable i , denominado también *carga factorial* o *saturación de una variable en un factor*. Según la formulación del modelo, cada una de las p variables observables es una combinación lineal de k factores comunes a todas las variables ($k < p$) y de un factor único para cada variable. Así, todas las variables originales están influenciadas por todos los factores comunes, mientras que para cada variable existe un único factor que es específico para esa variable. Tanto los factores comunes como los específicos son variables no observables. De forma matricial el modelo queda:

$$X = LF + U \quad [22]$$

donde X es la matriz formada por las variables originales, L es la matriz de cargas factoriales, F es la matriz formada por los factores comunes y U es la matriz formada por los factores únicos o específicos.

1.4.2. Hipótesis en el Modelo Factorial

Para poder aplicar la teoría de la inferencia estadística en el modelo factorial es necesario formular hipótesis sobre los factores comunes y sobre los específicos. Estas hipótesis son:

- ✓ $E(F) = 0$. la esperanza del vector de factores communes es cero.
- ✓ $Cov(F) = I$. La matriz de covarianzas de los factores communes es la matriz identidad.
- ✓ $E(U) = 0$. La esperanza del vector de factores únicos se supone que es el vector cero.
- ✓ $Cov(U) = \Psi = diag(\psi_1, \dots, \psi_p)$. Las varianzas de los factores únicos pueden ser distintas y dichos factores están incorrelacionados entre si.
- ✓ $Cov(F, U) = 0$. La matriz de covarianzas entre los factores communes y los factores únicos es la matriz cero.

1.4.3. Comunalidad y especificidad (unicidad)

Dado que la variable X_j está tipificada (media cero y varianza unidad), se puede descomponer su variabilidad como:

$$Var(X_j) = 1 = (l_{j1}^2 + \dots + l_{jk}^2) + \psi_j^2 = h_j^2 + \psi_j^2 \quad [23]$$

Donde h_j^2 es la parte de varianza de la variable X_j debida a los factores comunes, y se denomina **comunalidad**. Por otro lado, ψ_j^2 es la parte de la varianza de la variable X_j debida a los factores únicos o específicos y se denomina *especificidad o unicidad*.

1.4.4. Diseño del análisis

Los pasos a seguir son los siguientes:

- ✓ Selección de variables: Seleccionamos variables métricas. En caso contrario, necesitamos realizar una transformación de no métricas a métricas.
- ✓ Tamaño muestral: La muestra no debe ser inferior a 50 observaciones. Lo aconsejable es que sea ≥ 100 . Como regla general, deben tenerse por lo menos cinco veces más observaciones que variables, tomando como ratio óptimo un ratio de diez a uno.
- ✓ Cálculo de la matriz de correlaciones. El primer paso del análisis es el cálculo de la matriz de correlaciones de todas las variables que entran en el análisis. Una vez que se dispone de esta matriz concierne examinarla para comprobar si sus características son adecuadas para realizar un Análisis Factorial. Uno de los requisitos que deben cumplirse para que el Análisis Factorial tenga sentido es que las variables estén altamente correlacionadas. Pueden utilizarse diferentes métodos para comprobar el grado de asociación entre las variables:
 - El *determinante de la matriz de correlaciones*: un determinante muy bajo indicará altas correlaciones entre las variables, pero no debe ser cero (matriz no singular), pues esto indicaría que algunas de las variables son linealmente dependientes y no se podrían realizar ciertos cálculos necesarios en el Análisis Factorial.
 - *Test de Esfericidad de Bartlett*: Comprueba que la matriz de correlaciones se ajuste a la matriz identidad, es decir, ausencia de correlación significativa entre

las variables. Esto significa que la nube de puntos se ajustará a una esfera perfecta, expresando así la hipótesis nula por:

$$H_0: R = I \Rightarrow H_0: |R| = 1 \quad [24]$$

es decir, que el determinante de la matriz de correlaciones es igual a 1. Bartlett introdujo un estadístico para este contraste basado en la matriz de correlación R , que bajo la hipótesis nula H_0 tiene una distribución chi-cuadrado con $p(p - 1)/2$ grados de libertad. La expresión del estadístico es la siguiente:

$$\chi^2 = -[n - 1 - \frac{2p+5}{6}] \ln|R| \quad [25]$$

cuando $p > 0,05$, es decir, que no tenemos evidencias para rechazar la hipótesis nula, significa que las variables no están correlacionadas y por tanto no tiene mucho sentido llevar a cabo un Análisis Factorial. Este contraste es útil cuando el tamaño muestral es pequeño.

- *Indice KMO de Kaiser-Meyer-Olkin.* En un modelo con varias variables, el coeficiente de correlación parcial mide la correlación existente entre ellas una vez se han descontado los efectos lineales del resto de variables. En el modelo factorial, se pueden considerar tanto los efectos de las otras variables como los correspondientes a los factores comunes. Por lo tanto, el coeficiente de correlación parcial entre dos variables sería equivalente al coeficiente de correlación parcial entre los factores únicos de esas dos variables. Pero de acuerdo con el modelo factorial los coeficientes de correlación teóricos calculados entre cada par de factores únicos son nulos por hipótesis, y como los coeficientes de correlación parcial constituyen una aproximación a dichos coeficientes teóricos, deben estar próximos a cero. El índice KMO de adecuación global del modelo está basado en los coeficientes de correlación observados en cada par de variables y en sus coeficientes de correlación parcial de la siguiente forma:

$$KMO = \frac{\sum_j \sum_{h \neq j} r_{jh}^2}{\sum_j \sum_{h \neq j} r_{jh}^2 + \sum_j \sum_{h \neq j} a_{jh}^2} \quad [26]$$

donde r_{jh}^2 representa al coeficiente de correlación simple al cuadrado y a_{jh}^2 al coeficiente de correlación parcial al cuadrado. En el caso de que exista adecuación de los datos a un modelo de Análisis Factorial, el valor de los coeficientes de correlación parcial será pequeño y por tanto el valor de KMO será próximo a la unidad. Por tanto, valores bajos del índice KMO desaconsejan la utilización del Análisis Factorial. Como baremo para interpretar el índice KMO podría tomarse según Kaiser:

<u>Criterio</u>	<u>Valoración</u>
$1 \geq KMO \geq 0.9$	muy bueno
$0.9 \geq KMO \geq 0.8$	meritorio
$0.8 \geq KMO \geq 0.7$	mediano
$0.7 \geq KMO \geq 0.6$	mediocre
$0.6 \geq KMO > 0.5$	bajo
$KMO \leq 0.5$	inaceptable

- *Correlación Anti-imagen.* El negativo del coeficiente de correlación parcial. Deberá haber pocos coeficientes altos para que sea razonable aplicar el Análisis Factorial.
- *Medida de Adecuación de la Muestra* (MSA, measure of sampling adequacy). Equivalente a KMO, pero evaluando cada variable X_1, \dots, X_p . Por tanto:

$$MSA_j = \frac{\sum_{h \neq j} r_{jh}^2}{\sum_{h \neq j} r_{jh}^2 + \sum_{h \neq j} a_{jh}^2} \quad [27]$$

- *Coeficiente de Correlación Múltiple.* Deberá ser alto. Esta técnica, por defecto, toma los valores de la correlación múltiple al cuadrado como los valores iniciales de comunalidad.

1.4.5. Extracción de los factores

Existen diversos procedimientos de extracción de los factores. Veamos algunos de ellos:

- a) *Componentes principales.* Como hemos visto, se forman combinaciones lineales independientes de las variables observadas. La primera componente tiene la varianza máxima. Las componentes sucesivas explican progresivamente proporciones menores de la varianza y no están correlacionadas las unas con las otras. Por tanto, se trata de considerar como factores los k primeros componentes principales. El análisis de componentes principales se utiliza para obtener la solución factorial inicial. Puede utilizarse cuando una matriz de correlaciones es singular.

Siguiendo el apartado anterior:

$$y = Ax \Rightarrow A'y = x \quad [28]$$

El modelo factorial quedaría definido como:

$$L = A_{(k)}' = [a_1, a_2, \dots, a_k] \quad F = y_{(k)} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_k \end{bmatrix} \quad U = X - LF \quad [29]$$

- b) *Mínimos cuadrados no ponderados.* Los factores se obtienen minimizando la suma de los cuadrados de las diferencias entre las matrices de correlaciones observada y reproducida, ignorando las diagonales.
- c) *Mínimos cuadrados generalizados.* Se minimiza la suma de los cuadrados de las diferencias entre las matrices de correlación observada y reproducida. Las correlaciones se ponderan por el inverso de su unicidad, de manera que las variables que tengan un valor alto de unicidad reciban un peso menor que aquellas que tengan un valor bajo de unicidad.
- d) *Método de máxima verosimilitud.* Proporciona las estimaciones de los parámetros que con mayor probabilidad han producido la matriz de correlaciones observada, bajo el supuesto de que la muestra procede de una distribución normal multivariada. Las correlaciones se ponderan por el inverso de la unicidad de las variables, y se emplea un algoritmo iterativo.
- e) *Factor o Eje principal.* Parte de la matriz de correlaciones original con los cuadrados de los coeficientes de correlación múltiple insertados en la diagonal principal como estimaciones iniciales de las comunalidades. Las saturaciones factoriales resultantes se utilizan para estimar de nuevo las comunalidades y reemplazan a las estimaciones previas en la diagonal de la matriz. Las iteraciones continúan hasta que el cambio en

las comunidades, de una iteración a la siguiente, satisface el criterio de convergencia para la extracción.

- f) *Análisis Alfa*. Considera a las variables incluidas en el análisis como una muestra del universo de las variables posibles. Este método maximiza el Alfa de Cronbach para los factores.
- g) *Análisis Imagen*. Fue desarrollado por Guttman y está basado en la teoría de las imágenes. La parte común de una variable, llamada la imagen parcial, se define como su regresión lineal sobre las restantes variables, en lugar de ser una función de los factores hipotéticos.

Sin entrar en demasiados detalles, el primero es adecuado cuando se trata de resumir la mayor parte posible de la información inicial (varianza) en el menor número de factores posibles que, por otra parte, es la utilización más frecuente del Análisis Factorial. El método de ejes principales es más adecuado cuando se trata de identificar factores o dimensiones que reflejan lo que las variables comparten en común. Dado que, básicamente, todos los métodos ofrecen resultados muy similares, se suele utilizar siempre el primero de ellos.

1.4.6. La matriz factorial o de componentes

A partir de una matriz de correlaciones, el Análisis Factorial extrae otra matriz que reproduce la primera de forma más sencilla. Esta nueva matriz se denomina *matriz factorial* y adopta la siguiente forma:

	1	2
1	p_{11}	p_{21}
2	p_{12}	p_{22}
3	p_{13}	p_{23}
4	p_{14}	p_{24}
5	p_{15}	p_{25}
6	p_{16}	p_{26}

Cada columna es un factor y hay tantas filas como variables originales. Los elementos p_{ij} pueden interpretarse como índices de correlación entre el factor i y la variable j, aunque estrictamente solo son correlaciones cuando los factores no están correlacionados entre sí, es decir, son ortogonales. Estos coeficientes reciben el nombre de *pesos, cargas, ponderaciones o saturaciones* factoriales. Los pesos factoriales indican el peso de cada variable en cada factor. Lo ideal es que cada variable cargue alto en un factor y bajo en los demás. Cada variable se podría escribir en función de los factores de la siguiente forma:

$$\begin{aligned} V_1 &= p_{11}F_1 + p_{21}F_2 \\ V_2 &= p_{12}F_1 + p_{22}F_2 \\ &\dots \\ V_6 &= p_{16}F_1 + p_{26}F_2 \end{aligned} \quad [30]$$

La suma de los cuadrados de los pesos por filas indica la cantidad total de varianza de cada variable que explica el conjunto de factores considerados, es decir, las comunidades.

1.4.7. Autovalores o valores propios

El cuadrado de una carga factorial indica la proporción de la varianza explicada por un factor en una variable particular.

La suma de los cuadrados de los pesos de cualquier columna de la matriz factorial es lo que denominamos autovalores e indican la cantidad total de varianza que explica ese factor para las variables consideradas como grupo.

Las cargas factoriales pueden tener como valor máximo 1, por tanto, el valor máximo que puede alcanzar el valor propio es igual al número de variables.

Si dividimos el valor propio entre el número de variables, nos indica la proporción (tanto por ciento si multiplicamos por 100) de la varianza de las variables que explica el factor.

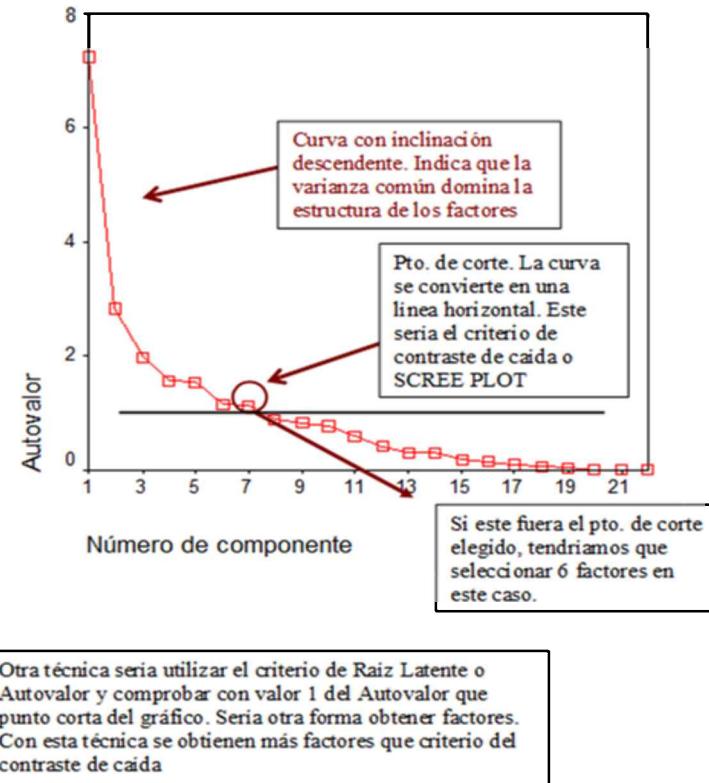
1.4.8. Número de factores a conservar

La matriz factorial puede presentar un número de factores superior al necesario para explicar la estructura de los datos originales. Generalmente hay un conjunto reducido de factores, los primeros, que son los que explican la mayor parte de la variabilidad total. Los otros factores suelen contribuir relativamente poco.

Uno de los problemas que se plantean, por tanto, consiste en determinar el número de factores que debemos conservar, de manera que se cumpla el principio de parsimonia. No debe olvidarse que el número de factores es una variable relacionada directamente con la pérdida de información e inversamente con la interpretabilidad de los resultados. Los procedimientos más habituales son los siguientes:

- a) *Criterio de los autovalores.* Solamente aquellos factores que tengan un autovalor superior a la unidad se retendrán para el análisis. La lógica de este criterio es que un factor, para ser útil, debe servir para explicar la varianza de, al menos, una variable. Este criterio es adecuado cuando el número de variables está entre 20 y 50. Cuando es inferior a 20, hay una tendencia a que este procedimiento extraiga pocos factores y demasiados si hay más de 50 variables. Este criterio también se llama "Criterio de la varianza media". Se eligen aquellas componentes (factores) cuyo autovalor sea mayor que la varianza media medida como: $\sum(\text{autovalores})/p$, donde p es el número de variables. Cuando trabajamos con la matriz de correlaciones (R) o datos estandarizados, la varianza media es igual a uno ya que $\sum(\text{autovalores})/p = p/p=1$. Hay que tener en cuenta que este criterio es solo aplicable cuando trabajamos con la matriz de correlaciones.
- b) *Criterio de porcentaje de varianza explicada.* El fin de este criterio es asegurar la significatividad de los resultados del análisis factorial. Aunque no se ha establecido unos niveles objetivos, se considera que en las ciencias naturales hay que extraer factores hasta que se explique por lo menos el 95% de la varianza, mientras que en las ciencias sociales el 60% (incluso menos), es un nivel considerado como razonable.
- c) *Criterio del gráfico de sedimentación o del contraste de caída.* El gráfico de sedimentación se obtiene al representar en ordenadas las raíces características y en abcisas el número de factores. Uniendo todos los puntos se obtiene una figura que, en general, se parece al perfil de una montaña con una pendiente fuerte hasta llegar a la base, formada por una meseta con una ligera inclinación. De acuerdo con el criterio gráfico, se retienen todos aquellos factores previos a la zona de sedimentación.

FIGURA 1. GRÁFICO DE SEDIMENTACIÓN



Fuente: Elaboración Propia

- d) *Criterio a priori.* Ya se sabe de antemano cuántos factores hay que extraer sobre la base de un estudio previo.

Por último, los factores pueden ser calculados en base a la matriz de covarianzas o de correlaciones. Se considera más adecuado, en general, el criterio de la matriz de correlaciones, puesto que de esta manera el análisis no estará influenciado por las unidades de medida.

1.4.9. Rotación de los factores

La matriz factorial indica, como sabemos, la relación entre los factores y las variables. Sin embargo, a partir de la matriz factorial muchas veces resulta difícil la interpretación de los factores. Es frecuente que varias variables presenten coeficientes factoriales elevados en más de un factor, cuando lo que interesa es que la mayor parte de su variabilidad quede explicada por un solo factor.

Esto lleva al desarrollo del principio de *estructura simple*, según el cual las variables han de saturar básicamente en un factor, es decir, que sus coeficientes factoriales han de ser elevados en un solo factor y bajos en el resto. Este principio es exigido en la mayoría de los análisis e incluye los siguientes supuestos:

- ✓ Cada factor o componente ha de tener unos pocos coeficientes factoriales elevados y el resto próximos a cero.
- ✓ Cada variable original ha de presentar un coeficiente factorial elevado en un solo factor (saturar solo a un factor).
- ✓ Los factores o componentes no han de tener la misma distribución. Deben mostrar unos modelos diferentes de coeficientes factoriales elevados y bajos.

Si buscamos simplificar la estructura factorial tenemos que proceder a la rotación. La rotación consiste en girar los ejes factoriales para que éstos se aproximen a las variables originales. La finalidad es facilitar la interpretación de la matriz factorial, forzando a las variables a definirse más en una dimensión latente, con preferencia a otras. De esta forma se obtiene una mayor diferenciación entre los factores que logran un perfil más definido.

Tras la rotación el número de factores se mantiene al igual que el porcentaje de varianza total explicada por el modelo original y la communalidad de las variables. Lo que varía es la composición de los factores al cambiar los coeficientes factoriales de cada variable en cada factor. Esto también altera la proporción de variabilidad explicada por cada factor. En la rotación se redistribuye la varianza entre todos los factores.

Existen diferentes procedimientos de rotación que se pueden englobar en dos grandes grupos:

- ✓ *Rotación ortogonal.* Parte del supuesto de incorrelación de las dimensiones latentes. Todo factor o componente es independiente de los demás por lo que los ejes factoriales forman entre sí un ángulo de 90º. Dentro de este tipo de rotaciones destacan se encuentran:
 - Varimax: Es el más empleado y el que se aplica por defecto en la mayoría de los paquetes estadísticos. Fue propuesto por Kaiser y su finalidad es simplificar la estructura factorial maximizando la varianza de los coeficientes factoriales al cuadrado para cada factor. En la aplicación de la rotación varimax pueden emplearse coeficientes factoriales brutos o normalizados. Lo habitual es utilizar normalizados (o estandarizados) para equilibrar la influencia de variables con distintas communalidades en la solución factorial. La aplicación de los coeficientes normalizados proporciona a cada variable original igual peso en la rotación. De esta forma se evita que variables con communalidades más elevadas influyan más en la solución factorial. Esta rotación es muy adecuada cuando las variables presentan communalidades altas en unas variables y bajas en otras.
 - Quartimax: Este procedimiento también busca la maximización de la varianza, pero en este caso la de los coeficientes factoriales al cuadrado para cada variable. De esta manera cada variable deberá tener una correlación elevada en unos pocos factores y baja en otros. Analíticamente quartimax es más sencillo que varimax, pero este último ofrece una separación más clara de los factores.
 - Equamax: Este procedimiento se presenta como una síntesis de los dos anteriores. Busca tanto la simplificación de variables como de factores. Es el menos utilizado.
- ✓ *Rotación oblicua.* Muestra mayor adecuación a situaciones habituales de interrelación entre dimensiones latentes. Al permitirse la correlación, los ejes factoriales no son perpendiculares entre sí. La consideración de dimensiones interrelacionadas comporta mayor complejidad en la interpretación de la solución factorial, aunque ésta es teórica y empíricamente más realista. Dentro de este tipo de rotaciones destacan:
 - Oblimin. Es uno de los procedimientos de rotación oblicua más utilizado. Suele aplicarse con coeficientes factoriales normalizados, obtenidos de la división del coeficiente factorial al cuadrado por la raíz cuadrada de la communalidad de la variable respectiva. Este procedimiento de rotación resulta de la combinación de dos criterios:

- Quartimin. Se asemeja al quartimax en la rotación ortogonal. Minimiza la suma de los productos de los coeficientes factoriales y produce factores de mayor oblicuidad.
- Biquartimin o covarimin. Minimiza la covarianza de los coeficientes factoriales cuadrados. Tiende a generar factores menos oblicuos y, también suele utilizar coeficientes factoriales normalizados.
- Oblimax. Criterio alternativo al quartimin. Persigue la simplificación de la estructura factorial aumentando el número de coeficientes factoriales elevados y bajos. Disminuye los de rango medio.
- Promax. Es uno de los procedimientos más novedosos. Se caracteriza por conseguir la solución oblicua aplicando algunas funciones de la solución ortogonal, como la matriz "target": aquella que el investigador tiene en mente o cree que existe. Este criterio se utiliza cuando se quiere comprobar la congruencia de una estructura factorial con otra que se conoce.

1.4.10. Puntuaciones factoriales

En ocasiones el análisis factorial es un paso previo a otros análisis. Por eso, una vez que se tienen los factores puede interesar conocer qué puntuación obtendrían los sujetos en estos factores. Para contestar a esto hay que calcular lo que se conoce como puntuaciones factoriales de cada individuo. Estos coeficientes de puntuaciones factoriales se extraen de la matriz de coeficientes de puntuaciones factoriales.

Sin embargo, salvo que se haya aplicado el análisis en componentes principales para la extracción de los factores, no se obtienen unas puntuaciones exactas para los factores. En su lugar, es preciso realizar estimaciones para obtenerlas. Algunos métodos para realizar estas estimaciones son los de mínimos cuadrados, regresión, Anderson-Rubin y Bartlett.

1.4.11. Interpretación de los factores

En la fase de interpretación juega un papel preponderante la teoría y el conocimiento sustantivo. A efectos prácticos se sugieren dos pasos en el proceso de interpretación:

1. Estudiar la composición de las saturaciones factoriales significativas de cada factor.
2. Intentar dar nombre a los factores. Nombre que se debe dar de acuerdo con la estructura de sus saturaciones, es decir, conociendo su contenido.

Dos cuestiones que pueden ayudar a la interpretación son:

- ✓ Ordenar la matriz rotada de forma que las variables con saturaciones altas en un factor aparezcan juntas.
- ✓ La eliminación de las cargas factoriales bajas (generalmente aquellas que van por debajo de 0,25).

Llamaremos *variable compleja* a aquella que satura altamente en más de un factor y que no debe ser utilizada para dar nombre a los factores.

Factores bipolares, son aquellos factores en los que unas variables cargan positivamente y otras tienen carga negativa.

1.4.12. Casos de Heywood y otras anomalías sobre estimaciones de comunalidad

Como las comunalidades son correlaciones al cuadrado, es de esperar que siempre estén entre 0 y 1. Sin embargo, una peculiaridad matemática del modelo de factor común es que las estimaciones finales de comunalidad podrían exceder 1.

Si una comunalidad es igual a 1, estamos ante un caso Heywood, y si excede de 1, ante un caso ultra-Heywood. Un caso ultra-Heywood implica que algún factor único tiene varianza negativa, una clara evidencia de que algo está mal. Las causas posibles incluyen las siguientes:

- ✓ Malas estimaciones de comunalidad previa.
- ✓ Demasiados factores comunes.
- ✓ Muy pocos factores comunes.
- ✓ No hay suficientes datos para proporcionar estimaciones estables.
- ✓ El modelo de factor común no es un modelo apropiado para los datos.

Un caso ultra-Heywood hace que una solución factorial sea inválida. Los analistas no están de acuerdo sobre si una solución factorial con un caso Heywood puede considerarse legítima o no.

Teóricamente, la comunalidad de una variable no debe exceder su confiabilidad. La violación de esta condición se conoce como un caso cuasi-Heywood y debe considerarse con la misma sospecha que un caso ultra-Heywood.

Los elementos de la estructura factorial y las matrices de estructura de referencia pueden exceder 1 solo en presencia de un caso ultra-Heywood. Por otro lado, un elemento del patrón factorial podría superar 1 en una rotación oblicua.

El método de Máxima Verosimilitud es especialmente susceptible a casos cuasi- o ultra-Heywood. Durante el proceso de iteración, una variable con alta comunalidad recibe un alto peso; esto tiende a aumentar su comunalidad, lo que aumenta su peso, y así sucesivamente.

A menudo se afirma que la correlación múltiple al cuadrado de una variable con las otras variables es un límite inferior a su comunalidad. Esto es cierto si el modelo de factores comunes se ajusta perfectamente a los datos, pero generalmente no es el caso con datos reales. Una estimación de comunalidad final que es menor que la correlación múltiple al cuadrado puede, por lo tanto, indicar ajuste pobre, posiblemente debido a factores no suficientes. De ninguna manera es un problema tan serio como un caso ultra-Heywood.

Los métodos de factor que usan el método de Newton-Raphson en realidad pueden producir comunalidades menores que 0, un resultado aún más desastroso que un caso ultra-Heywood.

La correlación múltiple al cuadrado de un factor con las variables puede superar 1, incluso en ausencia de casos ultra-Heywood. Esta situación también es motivo de alarma. El Análisis Alfa parece ser especialmente propenso a este problema, pero no ocurre con la Máxima Verosimilitud. Si una correlación múltiple al cuadrado es negativa, hay demasiados factores retenidos.

Con datos que no se ajustan perfectamente al modelo de factores comunes, puede esperarse que algunos de los valores propios sean negativos. Si un método de factor iterativo converge adecuadamente, la suma de los valores propios correspondientes a los factores rechazados debe ser 0; por lo tanto, algunos valores propios son positivos y algunos negativos. Si un

Análisis de Factor o Eje Principal no produce ningún valor propio negativo, las estimaciones de comunalidad previa son probablemente demasiado grandes. Los autovalores negativos causan que la proporción acumulada de varianza explicada exceda de 1 para un número suficientemente grande de factores. La proporción acumulada de varianza explicada por los factores retenidos debe ser de aproximadamente 1 para el Análisis de Ejes Principales y debe converger a 1 para los métodos iterativos. Ocasionalmente, un solo factor puede explicar más del 100 por ciento de la varianza común en un Análisis de Ejes Principales, lo que indica que las estimaciones de comunalidad previa son demasiado bajas.

Si una correlación canónica al cuadrado o un Coeficiente Alfa es negativo, hay demasiados factores retenidos.

El Análisis de Componentes Principales, a diferencia del análisis de factor común, no presenta ninguno de estos problemas si la matriz de covarianza o correlación se calcula correctamente a partir de un conjunto de datos sin valores faltantes. Varios métodos de valores perdidos para la correlación o el redondeo severo de las correlaciones pueden producir valores propios negativos en los Componentes Principales.

1.4.13. Ejemplos con R

Se considera la base de datos Seatbelts, que contiene datos mensuales de conductores de Gran Bretaña, muertos o heridos entre enero de 1969 y diciembre de 1983.

```
library(rela)
Belts <- Seatbelts[,1:7]
summary(Belts)

  DriversKilled      drivers      front      rear
  Min. : 60.0  Min. :1057  Min. : 426.0  Min. :224.0
  1st Qu.:104.8  1st Qu.:1462  1st Qu.: 715.5  1st Qu.:344.8
  Median :118.5  Median :1631  Median : 828.5  Median :401.5
  Mean   :122.8  Mean   :1670  Mean   : 837.2  Mean   :401.2
  3rd Qu.:138.0  3rd Qu.:1851  3rd Qu.: 950.8  3rd Qu.:456.2
  Max.   :198.0  Max.   :2654  Max.   :1299.0  Max.   :646.0

  kms      PetrolPrice      VanKilled
  Min. : 7685  Min. :0.08118  Min. : 2.000
  1st Qu.:12685 1st Qu.:0.09258  1st Qu.: 6.000
  Median :14987  Median :0.10448  Median : 8.000
  Mean   :14994  Mean   :0.10362  Mean   : 9.057
  3rd Qu.:17203 3rd Qu.:0.11406  3rd Qu.:12.000
  Max.   :21626  Max.   :0.13303  Max.   :17.000
```

Cálculo de la matriz de correlaciones:

```
correl=cor(Belts,use="pairwise.complete.obs")
correl

  DriversKilled      drivers      front      rear      kms
  DriversKilled  1.0000000  0.8888264  0.7067596  0.3533510 -0.3211016
  drivers        0.8888264  1.0000000  0.8084114  0.3436685 -0.4447631
  front          0.7067596  0.8084114  1.0000000  0.6202248 -0.3573823
  rear           0.3533510  0.3436685  0.6202248  1.0000000  0.3330069
  kms            -0.3211016 -0.4447631  -0.3573823  0.3330069  1.0000000
  PetrolPrice    -0.3866061 -0.4576675  -0.5392394 -0.1326272  0.3839004
  VanKilled      0.4070412  0.4853995  0.4724207  0.1217581 -0.4980356

  PetrolPrice      VanKilled
  DriversKilled -0.3866061  0.4070412
  drivers        -0.4576675  0.4853995
  front          -0.5392394  0.4724207
  rear           -0.1326272  0.1217581
  kms            0.3839004 -0.4980356
```

```
PetrolPrice    1.0000000 -0.2885584
VanKilled     -0.2885584  1.0000000
```

La prueba de esfericidad de Bartlett, como vimos, contrasta la hipótesis nula de que la matriz de correlaciones es una matriz identidad, en cuyo caso no existirían correlaciones significativas entre las variables y el modelo factorial no sería pertinente.

```
library(psych)
cortest.bartlett(Belts)
  R was not square, finding R from data
  $chisq
  [1] 968.7579

  $p.value
  [1] 1.260101e-191

  $df
  [1] 21
```

Para calcular los componentes principales basados en la matriz de correlaciones:

```
Belts.pc<-princomp(Belts, cor=TRUE)
summary(Belts.pc,loadings=TRUE)
  Importance of components:
                Comp.1   Comp.2   Comp.3   Comp.4   Comp.5
Standard deviation     1.9203515 1.2141729 0.8489513 0.78140438 0.5754445
Proportion of Variance 0.5268214 0.2106022 0.1029598 0.08722754 0.0473052
Cumulative Proportion  0.5268214 0.7374236 0.8403834 0.92761093 0.9749161
                                         Comp.6   Comp.7
Standard deviation     0.3298839 0.258386585
Proportion of Variance 0.0155462 0.009537661
Cumulative Proportion  0.9904623 1.000000000

  Loadings:
                Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6  Comp.7
DriversKilled      -0.444       0.136  0.521 -0.360  0.519 -0.323
drivers           -0.480       0.100  0.377          -0.505  0.596
front             -0.475 -0.198          -0.103  0.417 -0.414 -0.612
rear              -0.226 -0.686          -0.322  0.277  0.406  0.361
kms               0.276 -0.627          -0.606 -0.365 -0.168
PetrolPrice        0.327 -0.141  0.829  0.277  0.311
VanKilled         -0.334  0.259  0.526 -0.627 -0.387
```

A continuación se realiza un Análisis factorial con la librería **psych**. Para especificar el método de extracción de los factores se dispone del parámetro "fm", el cual puede tomar los siguientes valores:

- ✓ "minres" o "uls". Solución de Mínimos Cuadrados no ponderados.
- ✓ "ols". Difiere muy levemente de "minres" en que minimiza toda la matriz residual usando un procedimiento de MCO pero usa la primera derivada empírica. Esto será más lento.
- ✓ "wls". Solución de Mínimos Cuadrados Ponderados, MCP (WLS en inglés).
- ✓ "gls". Solución de Mínimos Cuadrados Generalizados ponderados, MCG (GLS en inglés).
- ✓ "pa". Solución de Factor o Eje Principal.
- ✓ "ml". Solución de Máxima Verosimilitud.
- ✓ "minchi". Se minimiza el chi-cuadrado ponderado por el tamaño de la muestra, al tratar correlaciones por pares con diferente número de sujetos por par.
- ✓ "minrank". Solución de Rango Mínimo (método en prueba).

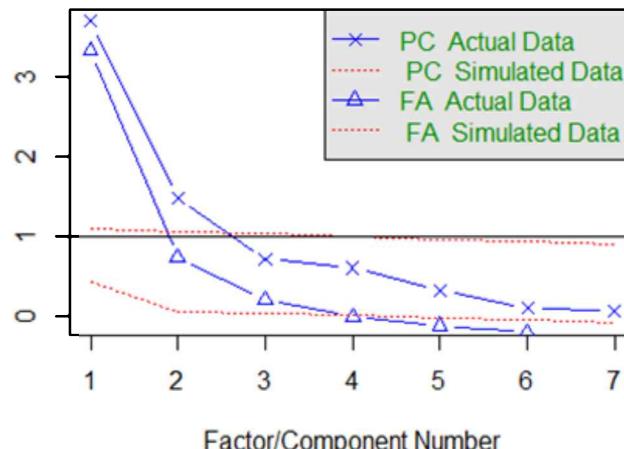
- ✓ "old.min". Solución de Mínimos Cuadrados no ponderados de la forma en que se hizo antes de abril de 2017.

Como método de extracción utilizamos los Mínimos Cuadrados Generalizados (`fm = "gls"`).

Representando el gráfico de autovalor para el criterio de contraste de caida, decidimos extraer 3 factores. Como vemos a continuación, se ha detectado un caso ultra-Heywwod.

```
library(psych)
fa.parallel(correl, n.obs = length(Belts), fm = "gls")
```

Parallel Analysis Scree Plots



```
Parallel analysis suggests that the number of factors = 3 and the number of
components = 2
Belts.ps <- fa(correl, n.obs = length(Belts), fm = "gls", nfactors = 3, rotate
="varimax")
Warning in fac(r = r, nfactors = nfactors, n.obs = n.obs, rotate =
rotate, : An ultra-Heywood case was detected. Examine the results carefully
print(Belts.ps) # Factor loadings as $loadings
Factor Analysis using method = gls
Call: fa(r = correl, nfactors = 3, n.obs = length(Belts), rotate = "varimax",
fm = "gls")
Standardized loadings (pattern matrix) based upon correlation matrix
      GLS1   GLS3   GLS2    h2     u2 com
DriversKilled  0.82  0.28  0.24  0.81  0.1855 1.4
drivers        0.87  0.41  0.24  0.99  0.0149 1.6
front          0.47  0.56  0.68  1.00 -0.0034 2.7
rear           0.21 -0.10  0.85  0.78  0.2249 1.2
kms            -0.19 -0.89  0.34  0.94  0.0603 1.4
PetrolPrice    -0.25 -0.47 -0.19  0.32  0.6818 1.9
VanKilled      0.29  0.50  0.10  0.34  0.6574 1.7

      GLS1   GLS3   GLS2
SS loadings  1.88  1.83  1.46
Proportion Var 0.27  0.26  0.21
Cumulative Var 0.27  0.53  0.74
Proportion Explained 0.36  0.35  0.28
Cumulative Proportion 0.36  0.72  1.00

Mean item complexity = 1.7
Test of the hypothesis that 3 factors are sufficient.

The degrees of freedom for the null model are 21 and the objective function was
5.16 with Chi Square of 6910.24
The degrees of freedom for the model are 3 and the objective function was 0.07
```

The root mean square of the residuals (RMSR) is 0.02
The df corrected root mean square of the residuals is 0.04

The harmonic number of observations is 1344 with the empirical chi square 15.05
with prob < 0.0018

The total number of observations was 1344 with Likelihood Chi Square = 92.1
with prob < 7.8e-20

Tucker Lewis Index of factoring reliability = 0.909
RMSEA index = 0.149 and the 90 % confidence intervals are 0.123 0.176
BIC = 70.49
Fit based upon off diagonal values = 1
Measures of factor score adequacy

	GLS1	GLS3	GLS2
Correlation of (regression) scores with factors	0.99	0.98	0.99
Multiple R square of scores with factors	0.97	0.96	0.99
Minimum correlation of possible factor scores	0.95	0.92	0.98

La librería factanal realiza el Análisis Factorial de Máxima Verosimilitud.

```
Belts.fak<-factanal(Belts, factors=3, rotation="varimax", scores = "Bartlett")
Belts.fa
```

Call:
factanal(x = Belts, factors = 3, scores = "Bartlett", rotation = "varimax")

Uniquenesses:

DriversKilled	drivers	front	rear	kms
0.197	0.005	0.005	0.227	0.051
PetrolPrice	VanKilled			
0.660	0.645			

Loadings:

	Factor1	Factor2	Factor3
DriversKilled	0.820	0.264	0.244
drivers	0.883	0.391	0.249
front	0.485	0.531	0.691
rear	0.204	-0.121	0.846
kms	-0.191	-0.903	0.311
PetrolPrice	-0.241	-0.469	-0.249
VanKilled	0.301	0.507	

	Factor1	Factor2	Factor3
SS loadings	1.915	1.813	1.482
Proportion Var	0.274	0.259	0.212
Cumulative Var	0.274	0.533	0.744

Test of the hypothesis that 3 factors are sufficient.
The chi square statistic is 11.54 on 3 degrees of freedom.
The p-value is 0.00914

La librería rela realiza el Análisis Factorial de Factor o Eje Principal.

```
library(rela)
res <- paf(as.matrix(Belts))
summary(res) # Calcula automáticamente KMO con MSA, determina el número de factores,
$KMO
[1] 0.66899

$MSA
      MSA
DriversKilled 0.75338
drivers        0.72295
front          0.65795
rear           0.40214
kms            0.53517
```

```

PetrolPrice  0.83413
VanKilled    0.88197

$Bartlett
[1] 968.76

$Communalities
      Initial Communalities Final Extraction
DriversKilled          0.79824      0.66181
drivers                 0.87486      0.85141
front                  0.87225      0.91049
rear                   0.76216      0.69016
kms                     0.67184      0.89761
PetrolPrice             0.36120      0.29405
VanKilled               0.36294      0.34590

$Factor.Loadings
      [,1]      [,2]
DriversKilled -0.80934 -0.082307
drivers        -0.92263 -0.012744
front          -0.92638 -0.228709
rear           -0.41852 -0.717642
kms            0.53335 -0.783039
PetrolPrice    0.53096 -0.110142
VanKilled      -0.55008  0.208105

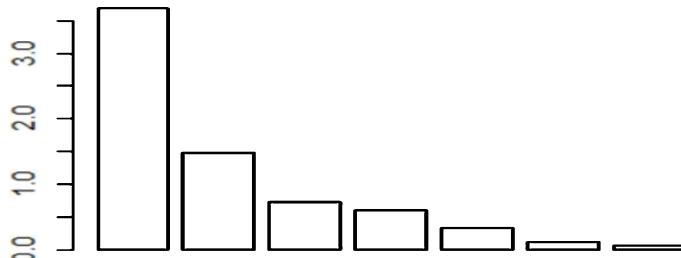
$RMS
[1] 0.050492

```

```

# calcula chi-cuadrado de la prueba de esfericidad de Bartlett, comunidades y
# cargas factoriales. Las comunidades son 1 menos la unicidad.
barplot(res$Eigenvalues[,1]) # Primera columna de valores propios.

```



```

resv <- varimax(res$Factor.Loadings) # La rotación Varimax es posible más tarde.
print(resv)
\$loadings

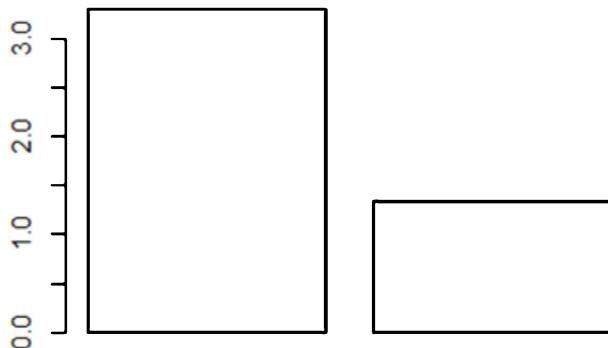
Loadings:
      [,1]      [,2]
DriversKilled -0.773 -0.254
drivers        -0.898 -0.211
front          -0.856 -0.422
rear           -0.255 -0.791
kms            0.689 -0.650
PetrolPrice    0.542
VanKilled      -0.582

      [,1]      [,2]
SS loadings   3.309 1.343
Proportion Var 0.473 0.192
Cumulative Var 0.473 0.664

$rotmat
      [,1]      [,2]
[1,]  0.97667  0.21474
[2,] -0.21474  0.97667

```

```
barplot(sort(colSums(loadings(resv))^2),decreasing=TRUE)) # Gráfico de sedimentación
usando las cargas rotadas.
```



```
scores <- as.matrix(Belts) %*% as.matrix(resv$loadings) # Obtiene las puntuaciones de
una manera sencilla.
```

2. ANÁLISIS DE CORRESPONDENCIAS

2.1. INTRODUCCIÓN

El nombre de análisis de correspondencia es una traducción del francés "Analyse des correspondances", el cual fue propuesto en los años 60 por el físico-matemático francés Benzécri, con el fin de definir, describir e interpretar el análisis a través de un gráfico geométrico.

Esta técnica analiza los datos tal como fue diseñada por algunos precursores de la estadística, entre los cuales destacan Pearson, Guttman o Fisher, los cuales, sin embargo, no pudieron llevar a cabo los cálculos por la carencia de instrumentos que permitiesen cálculos matemáticos tan complejos como los que pueden hoy en día realizar las computadoras y los software existentes en el mercado (SPSS, R, SAS, Minitab, Xlstat, etc.).

El análisis de correspondencias permite analizar las variables nominales y ordinales que dan lugar a una tabla de contingencia. Este análisis permite así, tanto descomponer y observar las relaciones de los niveles de las variables, como realizar un estudio de las mismas características que el análisis factorial, pero referido a variables medidas en una escala ordinal o nominal.

El Análisis de Correspondencias es una técnica estadística que se aplica al análisis de tablas de contingencia y construye un diagrama cartesiano basado en la asociación entre las variables analizadas. En dicho gráfico se representan conjuntamente las distintas modalidades de la tabla de contingencia, de forma que la proximidad entre los puntos representados está relacionada con el nivel de asociación entre dichas modalidades. Constituye el equivalente de componentes principales y coordenadas principales para variables cualitativas. Por tanto, estamos ante una técnica descriptiva diseñada para representar tablas de contingencia.

Cuando el análisis es de dos variables hablamos de análisis simple de correspondencias. Si es de tres o más, lo denominamos análisis de correspondencias múltiple.

En las tablas de contingencia se estudia si dos variables medidas en una escala nominal u ordinal tenían relación entre sí. En el análisis de correspondencias vamos a profundizar en la relación que existe entre ellas y los niveles de las mismas.

Entre la utilización del Análisis de Correspondencias Simple y Múltiple, citar: -² Preferencias de consumo en Investigación de Mercados. -² Posicionamiento de empresas a partir de las preferencias de consumidores. -² Búsqueda de tipologías de individuos respecto a variables cualitativas (patrones de enfermedades en medicina, perfiles psicológicos, comportamiento de especies en biología, etc.).

2.2. OBJETIVO

El objetivo principal de este análisis es determinar la posición que tienen los distintos niveles de las variables y relacionarlos entre sí en unas coordenadas a las que denominamos dimensiones. Éstas tienen mucha similitud con lo que en el análisis factorial se denominaban factores. Además, todo esto queda representado mediante gráficos o mapas de correspondencias que permiten visualizar mejor las relaciones obtenidas.

Como técnica factorial, las dimensiones que definen el espacio en que se representan las categorías se obtienen como factores cuantitativos, por lo que este análisis acaba siendo un método de extracción de variables ficticias cuantitativas a partir de variables cualitativas. Este hecho, puede permitir la aplicación de otros tipos de análisis de variables cuantitativas.

2.3. ANÁLISIS DE CORRESPONDENCIAS SIMPLE

2.3.1. Planteamiento

Si m y n son el número de categorías de la primera y la segunda variable respectivamente, la tabla de contingencia correspondiente tendrá m filas y n columnas, donde la intersección entre una fila y una columna da lugar a una celda, o casilla, cuya frecuencia observada es n_{ij} .

TABLA 1. TABLA DE CORRESPONDENCIAS

	Y_1	Y_2	...	Y_j	...	Y_n	
X_1	n_{11}	n_{12}	...	n_{1j}	...	n_{1n}	$n_{1\cdot}$
X_2	n_{21}	n_{22}	...	n_{2j}	...	n_{2n}	$n_{2\cdot}$
:	:	:	:	:	:	:	:
X_i	n_{i1}	n_{i2}	...	n_{ij}	...	n_{in}	$n_{i\cdot}$
:	:	:	:	:	:	:	:
X_m	n_{m1}	n_{m2}	...	n_{mj}	...	n_{mn}	$n_{m\cdot}$
$n_{\cdot 1}$	$n_{\cdot 2}$...	$n_{\cdot j}$...	$n_{\cdot n}$	$n_{\cdot \cdot}$	

Fuente: Elaboración propia

Siendo las frecuencias marginales:

$$n_{i\cdot} = \sum_{j=1}^n n_{ij} \quad n_{\cdot j} = \sum_{i=1}^m n_{ij} \quad n_{\cdot \cdot} = \sum_{i=1}^m n_{i\cdot} = \sum_{j=1}^n n_{\cdot j} = N \quad [31]$$

Cada fila puede ser considerada como un punto, dotado de **masa**, en un espacio de n dimensiones. Las coordenadas de cada punto se obtendrán a partir de las frecuencias en las n celdas de la fila correspondiente.

De la misma forma, cada columna puede ser considerada como un punto en un espacio de m dimensiones. En este caso, las coordenadas de cada punto se obtendrán a partir de las frecuencias en las m celdas de la columna correspondiente.

A partir de la representación de los m puntos-fila o, equivalentemente, de los n puntos-columna, se tratará de extraer un nuevo espacio, de pequeña dimensión, tal que, al proyectar la nube de puntos en dicho espacio, la deformación de las distancias originales entre los puntos sea pequeña.

2.3.2. Definición de perfiles

Perfiles marginales

Los perfiles marginales describen la distribución marginal de las variables X e Y, respectivamente, a través de las frecuencias relativas marginales:

TABLA 2. PERFILES MARGINALES

	X_1	X_2	...	X_i	...	X_m	Total
Perfil fila	$\frac{n_{1 \cdot}}{n_{..}}$	$\frac{n_{2 \cdot}}{n_{..}}$...	$\frac{n_{i \cdot}}{n_{..}}$...	$\frac{n_{m \cdot}}{n_{..}}$	1
	$n_{..}$	$n_{..}$		$n_{..}$		$n_{..}$	
	Y_1	Y_2	...	Y_j	...	Y_n	Total
Perfil columna	$\frac{n_{\cdot 1}}{n_{..}}$	$\frac{n_{\cdot 2}}{n_{..}}$...	$\frac{n_{\cdot j}}{n_{..}}$...	$\frac{n_{\cdot n}}{n_{..}}$	1
	$n_{..}$	$n_{..}$		$n_{..}$		$n_{..}$	

Fuente: Elaboración propia

Perfiles condicionales

Los perfiles fila describen las distribuciones condicionadas de la variable Y por las distintas modalidades de la variable X. Se obtienen a partir de la Tabla de Correspondencias y el perfil marginal de X de la siguiente manera.

TABLA 3. PERFILES FILA

$f(Y/X = x_i)$	Y_1	Y_2	...	Y_j	...	Y_n	Total
X_1	$\frac{n_{11}}{n_{1 \cdot}}$	$\frac{n_{12}}{n_{1 \cdot}}$...	$\frac{n_{1j}}{n_{1 \cdot}}$...	$\frac{n_{1n}}{n_{1 \cdot}}$	1
X_2	$\frac{n_{21}}{n_{2 \cdot}}$	$\frac{n_{22}}{n_{2 \cdot}}$...	$\frac{n_{2j}}{n_{2 \cdot}}$...	$\frac{n_{2n}}{n_{2 \cdot}}$	1
:	:	:	:	:	:	:	:
X_i	$\frac{n_{i1}}{n_{i \cdot}}$	$\frac{n_{i2}}{n_{i \cdot}}$...	$\frac{n_{ij}}{n_{i \cdot}}$...	$\frac{n_{in}}{n_{i \cdot}}$	1
:	:	:	:	:	:	:	:
X_m	$\frac{n_{m1}}{n_{m \cdot}}$	$\frac{n_{m2}}{n_{m \cdot}}$...	$\frac{n_{mj}}{n_{m \cdot}}$...	$\frac{n_{mn}}{n_{m \cdot}}$	1

Fuente: Elaboración propia

Los perfiles columna describen las distribuciones condicionadas de la variable X por las distintas modalidades de la variable Y. Se obtienen a partir de la Tabla de Correspondencias y el perfil marginal de Y de la siguiente manera:

TABLA 4. PERFILES COLUMNA

$f(X/Y = y_i)$	Y_1	Y_2	...	Y_j	...	Y_n
X_1	$\frac{n_{11}}{n_{\cdot 1}}$	$\frac{n_{12}}{n_{\cdot 2}}$...	$\frac{n_{1j}}{n_{\cdot j}}$...	$\frac{n_{1n}}{n_{\cdot n}}$
X_2	$\frac{n_{21}}{n_{\cdot 1}}$	$\frac{n_{22}}{n_{\cdot 2}}$...	$\frac{n_{2j}}{n_{\cdot j}}$...	$\frac{n_{2n}}{n_{\cdot n}}$
:	:	:	:	:	:	:
X_i	$\frac{n_{i1}}{n_{\cdot 1}}$	$\frac{n_{i2}}{n_{\cdot 2}}$...	$\frac{n_{ij}}{n_{\cdot j}}$...	$\frac{n_{in}}{n_{\cdot n}}$
:	:	:	:	:	:	:
X_m	$\frac{n_{m1}}{n_{\cdot 1}}$	$\frac{n_{m2}}{n_{\cdot 2}}$...	$\frac{n_{mj}}{n_{\cdot j}}$...	$\frac{n_{mn}}{n_{\cdot n}}$
Total	1	1	1	1	1	1

Fuente: Elaboración propia

2.3.3. Medida de distancia utilizada

Trabajar con perfiles facilita la interpretación, pero también puede producir una visión distorsionada de la relación entre las variables, dado que bajo esta perspectiva todos los puntos tienen la misma importancia (los marginales de los perfiles fila y columna son iguales a 1).

Para evitar este problema el análisis de correspondencias utiliza una distancia que considera las diferencias entre los efectivos de cada fila (o columna). Esta distancia es la *chi-cuadrado*, la cual asigna a cada perfil un peso. Así cada fila (o columna) está afectada de un peso proporcional a su importancia en el conjunto, peso conocido como *masa*. Al considerar cada punto con una masa proporcional a su frecuencia se evita privilegiar las categorías con pocos efectivos. Se trata, por tanto, de una distancia euclídea ponderada por el inverso de la masa de las columnas cuando se mide la distancia entre filas, o por la masa de las filas para la distancia entre las columnas.

En el supuesto de independencia entre ambas variables, la frecuencia esperada será:

$$e_{ij} = \frac{n_{i\cdot}n_{\cdot j}}{n_{\cdot \cdot}} \quad [32]$$

Así, la distancia chi-cuadrado entre perfiles fila será:

$$d_{ij} = \sum_{h=1}^n \frac{1}{n_{\cdot h}} \left[\frac{n_{ih}}{n_{i\cdot}} - \frac{n_{jh}}{n_{\cdot j}} \right]^2 \quad [33]$$

En el caso de los perfiles columna:

$$d_{ij} = \sum_{h=1}^m \frac{1}{n_{\cdot h}} \left[\frac{n_{hi}}{n_{\cdot i}} - \frac{n_{hj}}{n_{\cdot j}} \right]^2 \quad [34]$$

El análisis de correspondencias intenta reproducir en sus representaciones gráficas estas distancias.

Las distancias no se miden entre dos filas o dos columnas sino con relación al perfil medio de fila o columna, es decir, con relación al promedio de las coordenadas de esa fila (o columna) ponderada por su masa (peso proporcional a su importancia en el conjunto). Este perfil medio aparecerá situado en el origen de coordenadas y es conocido como *centro de gravedad*.

La media de las distancias al cuadrado de cada punto fila al centro de gravedad se conoce como *inercia de las filas*. De igual modo, se define la *inercia de las columnas* como la media de las distancias al cuadrado de cada punto columna al centro de gravedad.

La *inercia total* de la nube de puntos (medida análoga a la variación total en el caso de las componentes principales) se calcula considerando todos los elementos de la tabla y, como puede observarse, coincide con el denominado estadístico chi-cuadrado dividido por el número de casos total. Esta inercia total va a ser posteriormente repartida entre las distintas dimensiones. Se puede decir por ello que existen muchos paralelismos con el análisis factorial. Lo que allí denominábamos valores propios aquí se denomina inercia.

$$\text{Inercia total} = \mu_T^2 = \frac{1}{n..} \sum_{i=1}^m \sum_{j=1}^n \frac{(n_{ij} - e_{ij})^2}{e_{ij}} \quad [35]$$

Una inercia baja significa que todos los puntos están situados muy cerca del centro de gravedad y que, en consecuencia, son muy similares, mientras que altos valores de inercia en determinadas categorías implican grandes diferencias con respecto al perfil medio de las filas o las columnas.

La distancia chi-cuadrado cumple el principio de la equivalencia distribucional, que postula que si dos categorías tienen perfiles idénticos pueden ser sustituidas por una sola categoría con peso igual a la suma de sus pesos, sin que con ello se modifique la distancia entre las filas o columnas.

La importancia de esta propiedad estriba en que garantiza la estabilidad en los resultados con independencia de la codificación en las variables. Esto permite agrupar categorías que tienen perfiles coincidentes, tanto por filas como por columnas. De igual modo, los resultados no mejorarán al realizar subdivisiones de categorías homogéneas.

Otra opción, utilizada en menor medida, sería trabajar con la *distancia euclídea*. El programa SPSS tiene por defecto la distancia chi-cuadrado, pero en la pantalla de especificación del modelo permite opcionalmente trabajar con esta distancia.

2.3.4. Extracción de las dimensiones o espacios factoriales

Con los perfiles de filas y columnas descritos se elabora la matriz de coordenadas (distancias) utilizando la distancia chi-cuadrado, que permitirá calibrar la magnitud de las diferencias entre la tabla de datos analizada y una tabla de datos sin relación entre las variables.

La extracción del espacio factorial se realiza mediante la representación de las categorías de las variables como puntos dotados de masa. La *masa* de cada punto será igual a la frecuencia relativa de observaciones en la categoría correspondiente.

Se extraerá un nuevo espacio de dimensión c , donde c es el mínimo entre m y n menos 1, de forma que:

- El primer eje o factor del nuevo espacio será aquel que, de todas las posibles proyecciones de la nube de puntos sobre un único eje, obtenga la mínima deformación.
- El segundo eje o factor será aquel que, de todas las posibles proyecciones de la nube de puntos sobre un espacio de dos dimensiones generado por el primer eje y un segundo eje perpendicular al primero, obtenga la mínima deformación, y así sucesivamente con el resto de los c ejes ($c = \min(m, n) - 1$). Es decir, si estuviésemos estudiando la relación entre cuatro productos (A, B, C, y D) y el aspecto más apreciado

en ellos por una serie de individuos (precio, estética y utilidad), el número máximo de dimensiones sería igual a $3-1 = 2$, ya que el menor número de niveles se da en las columnas (3) al restarle uno obtendremos dos.

Sin embargo, si el número de factores es grande será difícil su interpretación. Por tanto, a partir del espacio de dimensión c , se tratará de encontrar un subespacio de dimensión k que sea pequeño, pero que además pierda poca información respecto al espacio c -dimensional. Teniendo en cuenta que los primeros factores extraídos son los más importantes, el subespacio de dimensión k estará formado por los k primeros factores.

De forma análoga al Análisis de Componentes Principales, el método consiste en encontrar la descomposición en valores singulares de una matriz, en este caso:

$$C = (r_{ij}) \text{ siendo } r_{ij} = \frac{n_{ij} - e_{ij}}{\sqrt{e_{ij}}} \quad [36]$$

es decir, la matriz formada por los residuos de Pearson tipificados (o alternativamente las distancias euclídeas).

El resultado de la asignación de masas, en las direcciones de los ejes del espacio factorial, hará que unas categorías tengan más influencia que otras. Es decir, a mayor masa, mayor será la importancia relativa de la categoría correspondiente.

A continuación, se procede a diagonalizar la matriz C de varianza-covarianza con el fin de obtener los vectores y valores propios que definirán los nuevos ejes sobre los que será proyectada la nube de puntos.

El análisis de correspondencias busca encontrar dos matrices (A , B) de coordenadas cartesianas:

$$A = \begin{bmatrix} a'_1 \\ a'_2 \\ \vdots \\ a'_m \end{bmatrix} \quad B = \begin{bmatrix} b'_1 \\ b'_2 \\ \vdots \\ b'_n \end{bmatrix} \quad [37]$$

Donde la matriz A representa a los puntos fila, con $a_i = (a_{i1}, \dots, a_{ik})'$, y la matriz B a los puntos columna, con $b_j = (b_{j1}, \dots, b_{jk})'$. Generalmente, suele tomarse $k = 2$, si es posible.

Hay diversas formas de calcular las matrices A y B , conocidas como normalizaciones. El procedimiento más utilizado es el conocido como *normalización simétrica o canónica* (ACC), que busca satisfacer que el producto escalar (a_i, b_j) sea proporcional a los residuos tipificados r_{ij} .

La normalización simétrica o canónica descompone la matriz $C = (r_{ij})$ en valores singulares calculando las matrices $U_{m \times c}$, D y $V_{n \times c}$, con $c = \min\{m-1, n-1\}$, tales que $C = UDV'$, siendo $U'U = V'V = I$, con $D = \text{diag}(\lambda_1, \dots, \lambda_c)$, siendo λ_i los autovalores.

Las matrices A y B se calculan a partir de las siguientes expresiones:

$$A = D_m^{-1/2} UD \quad [38]$$

$$B = D_n^{-1/2} VD \quad [39]$$

con $D_m = \text{diag}(n_1, n_2, \dots, n_m)$ y $D_n = \text{diag}(n_1, n_2, \dots, n_n)$.

2.3.5. Método de normalización

Es una de las decisiones que se deben tomar a la hora de realizar el análisis. Existen cinco posibilidades:

- a) *Simétrica*. Para cada dimensión, las puntuaciones de fila son la media ponderada de las puntuaciones de columna divididas por el valor singular coincidente y las puntuaciones de columna son la media ponderada de las puntuaciones de fila divididas por el valor propio coincidente. Se utilizará esta visualización cuando se quiera examinar las diferencias o similaridades entre las categorías de las dos variables. Esta es la opción por defecto en SPSS. En R, este gráfico se realiza poniendo en el argumento "map" de la función "fviz_ca" el valor "symbiplot".
- b) *Principal*. El procedimiento descrito anteriormente. Las distancias entre los puntos de fila y los puntos de columna son aproximaciones de las distancias en la tabla de correspondencias de acuerdo con la medida de distancia seleccionada. Este método está diseñado para examinar las diferencias entre las categorías de una o de ambas variables, en lugar de las diferencias entre las dos variables. Esta es la opción por defecto en R, con valor "symmetric" en el argumento "map" de la función "fviz_ca".
- c) *Principal por fila*. Las distancias entre los puntos de fila son aproximaciones de las distancias en la tabla de correspondencias de acuerdo con la medida de distancia seleccionada. Las puntuaciones de fila son la media ponderada de las puntuaciones de columna. Este método está diseñado para examinar las diferencias o similaridades entre las categorías de la variable de filas. Por tanto, se debe emplear en casos en los que en las filas tengamos individuos, grupos, países o, en general, la categoría de la que queremos encontrar las diferencias.
- d) *Principal por columna* es igual que el caso anterior, pero referido a las columnas.
- e) Otras opciones disponibles, según programa usado, R o SPSS:
 - En R. Mapas asimétricos de Gabriel y Odoroff (1990). Se describe en el siguiente apartado.
 - En SPSS. Se ofrece una opción de *Personalizado*, como una opción intermedia entre la opción Principal y Principal por fila (o columna), donde se debe especificar un valor entre -1 y 1. El valor -1 sería equivalente a la opción correspondiente a Principal por columna. El valor 1 correspondería a Principal por fila. El valor 0 correspondería al caso simétrico. Todos los demás valores intermedios dispersarían la inercia entre las puntuaciones de columna y de fila en diferentes grados.

2.3.6. Interpretación de resultados

Inercia

La inercia total, expresada en función de las matrices A y B , se reparte entre las distintas dimensiones construidas de la siguiente manera:

$$\mu_T^2 = \sum_{h=1}^k \mu_h^2 = \sum_{h=1}^k \sum_{i=1}^m n_i \cdot a_{ih}^2 = \sum_{h=1}^k \sum_{j=1}^n n_j b_{jh}^2 \quad [40]$$

Las proporciones de inercia explicada para cada una de las dimensiones, $\frac{\mu_h^2}{\mu_T^2}$, ayudan a ponderar la importancia de cada una de las dimensiones a la hora de explicar las dependencias observadas

Las proporciones de inercia acumulada explicada por las i-ésimas primeras dimensiones, $\sum_{h=1}^i \frac{\mu_h^2}{\mu_T^2}$, permiten decidir el número mínimo de dimensiones necesario para explicar dichas dependencias.

Contribuciones totales

Cuantifican la importancia de cada una de las modalidades de las variables analizadas en la construcción de los ejes factoriales. Se utilizan para interpretar el significado de los ejes utilizando, para cada uno de ellos, las modalidades con contribuciones más fuertes. Se definen de la siguiente manera:

Contribución de la fila i-ésima a la dimensión h:

$$CT_h(i) = \frac{n_i \cdot a_{ih}^2}{\sum_{i=1}^m n_i \cdot a_{ih}^2} = \frac{n_i \cdot a_{ih}^2}{\mu_h^2} \quad [41]$$

Contribución de la columna j-ésima a la dimensión h:

$$CT_h(j) = \frac{n_j b_{jh}^2}{\sum_{j=1}^n n_j b_{jh}^2} = \frac{n_j b_{jh}^2}{\mu_h^2} \quad [42]$$

donde:

$$\sum_{i=1}^m CT_h(i) = \sum_{j=1}^n CT_h(j) = 1 \quad [43]$$

Contribuciones relativas

Miden la importancia de cada factor para explicar la posición (en el diagrama cartesiano) de cada una de las modalidades de las variables analizadas, representando la parte de la distancia al origen de coordenadas, explicada por dicho factor. El programa R las denomina cosenos al cuadrado.

Contribución relativa i-ésima fila a la dimensión h:

$$CR_h(i) = \frac{a_{ih}^2}{\sum_{h=1}^k a_{ih}^2} \quad [44]$$

Contribución relativa j-ésima columna a la dimensión h:

$$CR_h(j) = \frac{b_{jh}^2}{\sum_{h=1}^k b_{jh}^2} \quad [45]$$

Donde:

$$\sum_{i=1}^m CR_h(i) = \sum_{j=1}^n CR_h(j) = 1 \quad [46]$$

Se utilizan para analizar las proximidades entre los puntos haciendo hincapié en aquellas categorías cuyas contribuciones totales sean más elevadas cuando se desee explicar dichas proximidades.

Elementos complementarios

Son filas o columnas de la tabla de contingencia no utilizadas en el cálculo de los ejes factoriales pero que, una vez calculados éstos, se sitúan en el diagrama cartesiano con el fin

de ayudar en la interpretación de los resultados obtenidos. Sus coordenadas se calculan utilizando las relaciones baricéntricas existentes entre los puntos fila y columna.

2.4. ANÁLISIS DE CORRESPONDENCIAS MÚLTIPLE

2.4.1. Planteamiento

La extensión del análisis de correspondencias simples al caso de varias variables nominales (tablas de contingencia multidimensionales) se denomina Análisis de Correspondencias Múltiples, y utiliza los mismos principios generales que la técnica anterior. En general se orienta a casos en los cuales una variable representa ítems o individuos y el resto son variables cualitativas u ordinales que representan cualidades.

Suponiendo que tenemos n individuos en filas y q variables categóricas en las columnas, con p_k categorías cada una, $k = 1, 2, \dots, q$ mutuamente excluyentes y exhaustivas, entonces la tabla de datos de partida, o *Tabla disyuntiva* será ahora:

$$Z = [Z_1, Z_2, \dots, Z_q] \quad [47]$$

El número de columnas de Z será igual a $p = \sum_{k=1}^q p_k$, siendo Z_k una matriz $(n \times p_k)$ de forma que:

$$z_{ij}^k = \begin{cases} 1 & \text{si el individuo } i \text{ésimo elige la modalidad } j \\ 0 & \text{si el individuo } i \text{ésimo no elige la modalidad } j \end{cases} \quad [48]$$

Si hubiera alguna variable continua, debe transformarse en nominal, ordenándose en intervalos según rango de valores.

Bajo este esquema, los márgenes serán igual a:

$$z_{i\cdot} = q \quad [49]$$

$z_{\cdot j}$ = individuos que tienen modalidad j ésimas

$$z_{\cdot\cdot} = \sum_{i=1}^n \sum_{j=1}^p z_{ij} = nq \quad [50]$$

El Análisis de Correspondencias Múltiple se basa en realizar un Análisis de correspondencias sobre la llamada *Matriz de Burt*:

$$B = Z'Z \quad [51]$$

Ejemplo de cálculo de la Tabla disyuntiva y de la Matriz de Burt

Supongamos que estamos estudiando el colectivo de trabajadores de una determinada empresa. Los datos de partida serían los siguientes:

TABLA 5. DATOS DE PARTIDA

Individuo	Sexo	Edad	Nacionalidad
1 Hombre		De 50 a 64	Resto del mundo
2 Mujer		De 20 a 34	Española
3 Mujer		De 20 a 34	Española
4 Mujer		De 65 y más	Resto de la U.E.
5 Hombre		De 35 a 49	Española
6 Hombre		De 35 a 49	Resto del mundo
7 Mujer		De 65 y más	Resto de la U.E.
8 Mujer		Menos de 20	Española
9 Hombre		De 65 y más	Resto de la U.E.
10 Hombre		De 50 a 64	Resto de la U.E.
11 Hombre		Menos de 20	Resto de la U.E.
12 Hombre		De 20 a 34	Resto del mundo
13 Mujer		De 35 a 49	Española
14 Mujer		De 65 y más	Resto de la U.E.
15 Mujer		De 20 a 34	Resto del mundo

Fuente: Elaboración propia

La Tabla disyuntiva (Matriz Z) tendrá tantas columnas como categorías, 2 para sexo, 5 para edad y 3 para nacionalidad, en total 10.

TABLA 6. TABLA DISYUNTIVA

Sexo		Edad					Nacionalidad		
Hombre	Mujer	Menos de 20	De 20 a 34	De 35 a 49	De 50 a 64	De 65 y más	Española	Resto de la U.E.	Resto del mundo
1	0	0	0	0	1	0	0	0	1
0	1	0	1	0	0	0	1	0	0
0	1	0	1	0	0	0	1	0	0
0	1	0	0	0	0	1	0	1	0
1	0	0	0	1	0	0	1	0	0
1	0	0	0	1	0	0	0	0	1
0	1	0	0	0	0	1	0	1	0
0	1	1	0	0	0	0	1	0	0
1	0	0	0	0	0	1	0	1	0
1	0	0	0	0	1	0	0	1	0
1	0	1	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	1
0	1	0	0	1	0	0	1	0	0
0	1	0	0	0	0	1	0	1	0
0	1	0	1	0	0	0	0	1	0

Fuente: Elaboración propia

Las frecuencias marginales de las filas son iguales al número de preguntas ($q = 3$), y las frecuencias marginales de las columnas corresponden al número de sujetos que han elegido la modalidad (j) de la pregunta correspondiente, $j = 1, \dots, 10$, por lo que para cada subtabla (sexo, edad, nacionalidad), el número total de individuos es $n = 15$.

La tabla de contingencia de Burt (B) es una tabla simétrica de orden ($p \times p$): $B = Z' \cdot Z$.

TABLA 7. TABLA DE BURT

Sexo		Edad					Nacionalidad		
Hombre	Mujer	Menos de 20	De 20 a 34	De 35 a 49	De 50 a 64	De 65 y más	Española	Resto de la U.E.	Resto del mundo
Sexo	Hombre	7	0	1	1	2	2	1	3
	Mujer	0	8	1	3	1	0	3	1
Edad	Menos de 20	1	1	2	0	0	0	1	0
	De 20 a 34	1	3	0	4	0	0	2	2
	De 35 a 49	2	1	0	0	3	0	2	0
	De 50 a 64	2	0	0	0	0	2	0	1
	De 65 y más	1	3	0	0	0	4	0	0
Nacionalidad	Española	1	4	1	2	2	0	5	0
	Resto de la U.E.	3	3	1	0	0	1	4	6
	Resto del mundo	3	1	0	2	1	0	0	4

Fuente: Elaboración propia

Como puede observarse, B es una yuxtaposición de tablas de contingencias formada por $q^2 = 9$ bloques. Cada bloque es una tabla de contingencia de las variables dos a dos, salvo los bloques que están en la diagonal principal, que son las tablas de contingencia de cada variable consigo misma. Estos bloques son matrices con el valor de las frecuencias de cada categoría en la diagonal y cero en el resto de las celdas.

La tabla disyuntiva completa es equivalente a la tabla de Burt y ambas producen los mismos factores. Con la tabla de Burt se podrán obtener las puntuaciones (distancias al centro de gravedad), contribuciones absolutas de cada modalidad y variable a los ejes o factores obtenidos (contribución de cada modalidad o variable a la inercia de los nuevos ejes) y contribuciones relativas o correlaciones de cada modalidad con los nuevos ejes.

Como en la tabla de Burt las filas y las columnas representan las mismas modalidades, el estudio de ambas ofrece iguales resultados, por lo que solo se representan las de las filas.

2.4.2. Nube de puntos, perfiles

Perfiles fila (individuos)

$$\text{Coordenadas: } \frac{z_{ij}}{z_{i\cdot}} = \frac{z_{ij}}{q} \Rightarrow \text{Tabla: } \frac{1}{q} Z \quad [52]$$

$$\text{Peso: } \frac{z_{i\cdot}}{nq} = \frac{q}{nq} = \frac{1}{n} \quad [53]$$

$$\text{Distancia } \chi^2: d^2(i, i') = \frac{n}{q} \sum_{j=1}^p (z_{ij} - z_{i'j})^2 \quad [54]$$

Perfiles columna (modalidades)

$$\text{Coordenadas: } \frac{z_{ij}}{z_{\cdot j}} \quad [55]$$

$$\text{Peso: } \frac{z_{\cdot j}}{nq} \quad [56]$$

$$\text{Distancia } \chi^2: d^2(j, j') = n \sum_{j=1}^p \left(\frac{z_{ij}}{z_{\cdot j}} - \frac{z_{ij'}}{z_{\cdot j'}} \right)^2 \quad [57]$$

2.4.3. Inercia

Centro de gravedad

$$G = \left(\frac{1}{n}, \dots, \frac{1}{n} \right) \quad [58]$$

$$d^2(j, G) = n \sum_{j=1}^p \left(\frac{z_{ij}}{z_{\cdot j}} - \frac{1}{n} \right)^2 = n \sum_{j=1}^p \left(\frac{z_{ij}}{z_{\cdot j}^2} - \frac{2z_{ij}}{nz_{\cdot j}} + \frac{1}{n^2} \right) = \frac{n}{z_{\cdot j}} - 1 \quad [59]$$

Inercia de la modalidad j

$$I(j) = \frac{z_{\cdot j}}{nq} d^2(j, G) = \frac{z_{\cdot j}}{nq} \left(\frac{n}{z_{\cdot j}} - 1 \right) = \frac{1}{q} \left(1 - \frac{z_{\cdot j}}{n} \right) \quad [60]$$

Inercia de la variable k

$$I(k) = \sum_{j=1}^{p_k} I(j) = \sum_{j=1}^{p_k} \frac{1}{q} \left(1 - \frac{z_{\cdot j}}{n} \right) = \frac{1}{q} (p_k - 1) \quad [61]$$

Inercia total

$$I = \sum_{k=1}^q I(k) = \sum_{k=1}^q \frac{1}{q} (p_k - 1) = \frac{1}{q} (p - q) = \frac{p}{q} - 1 \quad [62]$$

(sin significado estadístico)

2.4.4. Solución del Análisis de Correspondencias

Matriz a diagonalizar

$$S = \frac{1}{q} Z' Z D^{-1} = \frac{1}{q} B D^{-1} \quad [63]$$

con $D = \text{diag}(z_{\cdot j})$

En \mathbb{R}^n :

$$\frac{1}{q} Z D^{-1} Z' \psi_a = \lambda_a \psi_a \quad [64]$$

En \mathbb{R}^p :

$$\frac{1}{q} Z' Z D^{-1} u_b = \lambda_b u_b \equiv \frac{1}{q} D^{-1} Z' Z \psi_b = \lambda_b \psi_b \quad [65]$$

siendo: $\psi_b = D^{-1} u_b$

Donde las matrices ψ_a, ψ_b serán las coordenadas en las dimensiones de la solución factorial.

2.4.5. Interpretación de los resultados

- Proximidad entre individuos en términos de parecido: dos individuos se parecen si tienen casi las mismas modalidades. Es decir, dos individuos están próximos si han elegido globalmente las mismas modalidades.
- Proximidad entre modalidades de variables diferentes en términos de asociación: dos modalidades están próximas si han sido elegidas globalmente por el mismo conjunto de individuos.
- Proximidad entre modalidades de una misma variable en términos de parecido: dado como ha sido construido el análisis, serán excluyentes; si son cercanas es porque los individuos que las poseen presentan casi el mismo comportamiento en las otras variables.

2.5. EJEMPLO DE ANÁLISIS DE CORRESPONDENCIAS SIMPLE CON EL SOFTWARE R

Varias funciones de diferentes paquetes están disponibles en el software R para calcular el Análisis de Correspondencias Simple:

- CA () [paquete *FactoMineR*]
- ca () [paquete *ca*]
- dudi.coa () [paquete *ade4*]
- corresp () [paquete *MASS*]
- epCA () [paquete *ExPosition*]

Independientemente de la función que se decida utilizar, se pueden extraer y visualizar fácilmente los resultados del análisis de correspondencia utilizando las funciones R proporcionadas en el paquete *factoextra*.

Aquí, usaremos FactoMineR (para el análisis) y factoextra (para la visualización basada en ggplot2).

2.5.1. Análisis exploratorio

La información que utilizaremos es una tabla de contingencia que contiene 13 tareas y su repartición en la pareja. Las filas son las diferentes tareas y las columnas (variables) el encargado de realizarlas (la esposa, alternativamente, el esposo o conjuntamente). Los valores consignados son las frecuencias de realización de cada una de las tareas.

```
library("FactoMineR")
library("factoextra")
  Loading required package: ggplot2

  Attaching package: 'ggplot2'
  The following objects are masked from 'package:psych':
    %+%, alpha
  Welcome! Related Books: `Practical Guide To Cluster Analysis in R` at
  https://goo.gl/13EFCZ
  data(housetasks)
  print(housetasks)

    Wife Alternating Husband Jointly
  Laundry      156      14       2       4
  Main_meal    124      20       5       4
  Dinner       77       11       7      13
  Breakfast    82       36      15       7
  Tidying      53       11       1      57
  Dishes        32       24       4      53
  Shopping      33       23       9      55
  Official     12       46      23      15
  Driving       10       51      75       3
  Finances      13       13      21      66
  Insurance      8       1       53      77
  Repairs        0       3      160       2
  Holidays       0       1       6      153
```

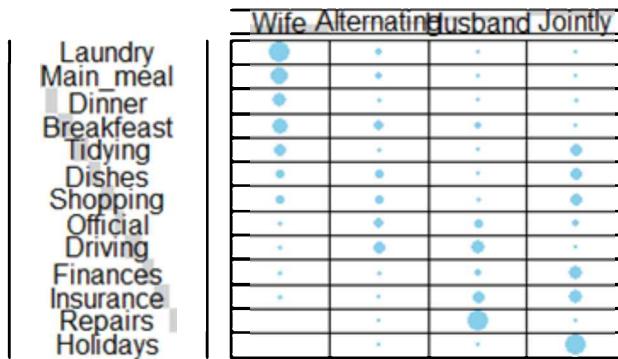
En primer lugar, realizaremos una análisis exploratorio de los datos mediante una visualización de la tabla de contingencia.

```
library("gplots")

  Attaching package: 'gplots'
  The following object is masked from 'package:stats':
    lowess

  # 1. Convierto los datos a formato tabla
  dt <- as.table(as.matrix(housetasks))
  # 2. Gráfico de Contingencia
  balloonplot(t(dt), main ="housetasks", xlab = "", ylab = "",
              label = FALSE, show.margins = FALSE)
```

housetasks



2.5.2. Estimación del modelo

Para realizar el análisis usaremos la función *CA* () [paquete FactoMiner], con parámetros:

CA (*X*, *ncp* = 5, *graph* = TRUE)

- *X*: un marco de datos (tabla de contingencia).
- *ncp*: número de dimensiones guardadas en los resultados finales (por defecto, 5).
- *graph*: un valor lógico. Si es TRUE (por defecto), se muestra un gráfico.

Para interpretar el análisis de correspondencias, el primer paso es evaluar si existe una dependencia significativa entre las filas y las columnas.

Esto se hace a través del estadístico chi-cuadrado para examinar la asociación entre las variables de fila y columna. Esto aparece en la parte superior del informe generado por el resumen de la función (res.ca) o impresión (res.ca). Una estadística alta de chi-cuadrado significa un fuerte vínculo entre las variables de fila y columna.

```
library("FactoMineR")
res.ca <- CA(housetasks, graph = FALSE)
print(res.ca)

**Results of the Correspondence Analysis (CA)**
The row variable has 13 categories; the column variable has 4 categories
The chi square of independence between the two variables is equal to 1944.5 (p-value = 0).
*The results are available in the following objects:
```

name	description
1 "\$eig"	"eigenvalues"
2 "\$col"	"results for the columns"
3 "\$col\$coord"	"coord. for the columns"
4 "\$col\$cos2"	"cos2 for the columns"
5 "\$col\$contrib"	"contributions of the columns"
6 "\$row"	"results for the rows"
7 "\$row\$coord"	"coord. for the rows"
8 "\$row\$cos2"	"cos2 for the rows"
9 "\$row\$contrib"	"contributions of the rows"
10 "\$call"	"summary called parameters"
11 "\$call\$marge.col"	"weights of the columns"
12 "\$call\$marge.row"	"weights of the rows"

Usaremos las siguientes funciones [en factoextra] para ayudar en la interpretación y la visualización del análisis de correspondencia:

- *get_eigenvalue(res.ca)*: Extrae los valores propios / varianzas retenidos por cada dimensión (eje).

- `fviz_eig(res.ca)`: Visualiza los valores propios.
- `get_ca_row(res.ca), get_ca_col(res.ca)`: Extrae los resultados para filas y columnas, respectivamente.
- `fviz_ca_row(res.ca), fviz_ca_col(res.ca)`: Visualiza los resultados para filas y columnas, respectivamente.
- `fviz_ca_biplot(res.ca)`: crea un biplot de filas y columnas.

2.5.3. Valores propios

Como se ha visto, los valores propios corresponden a la cantidad de información retenida por cada eje. Las dimensiones se ordenan de forma decreciente y se enumeran de acuerdo con la cantidad de varianza explicada en la solución. La dimensión 1 explica la mayor varianza en la solución, seguida de la dimensión 2, y así sucesivamente.

Los valores propios se pueden usar para determinar la cantidad de ejes que se deben retener. Como ya vimos en el análisis factorial, no existe una "regla de oro" para elegir la cantidad de dimensiones que se mantendrán para la interpretación de los datos. Depende del objetivo de la investigación y de la necesidad del investigador. Por ejemplo, si se está satisfecho con el 80% de las varianzas totales explicadas, entonces se usarán la cantidad de dimensiones necesarias para lograr eso.

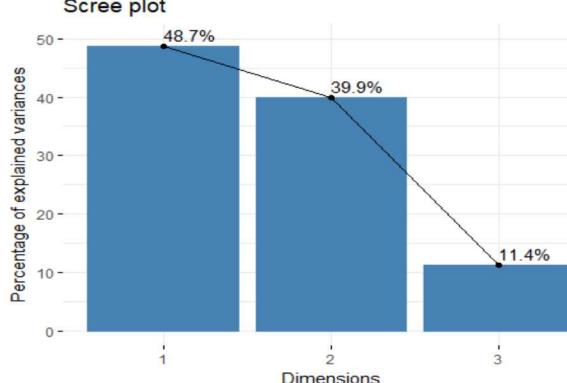
Otra alternativa es usar el gráfico de sedimentación, el cual obtendremos usando la función `fviz_eig()` o `fviz_screeplot()` [paquete factoextra].

También es posible calcular un valor propio promedio por encima del cual el eje debe mantenerse en la solución. Dado que nuestros datos contienen 13 filas y 4 columnas, si los datos fueran aleatorios, el valor esperado del valor propio para cada eje sería $1 / (m - 1) = 1/12 = 8.33\%$ en términos de filas. Del mismo modo, el eje promedio debería representar $1 / (n - 1) = 1/3 = 33.33\%$ en términos de las 4 columnas.

Como vemos en los resultados, alrededor del 88.6% de la variación se explica por las dos primeras dimensiones.

```
library("factoextra")
eig.val <- get_eigenvalue(res.ca)
eig.val
  eigenvalue variance.percent cumulative.variance.percent
Dim.1  0.54289    48.692        48.692
Dim.2  0.44500    39.913        88.605
Dim.3  0.12705    11.395       100.000
```

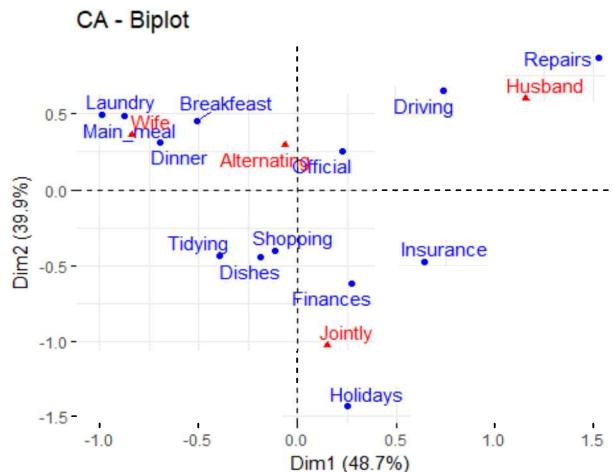
```
fviz_screeplot(res.ca, addlabels = TRUE, ylim = c(0, 50))
```



2.5.4. Biplot simétrico

Usaremos la función `fviz_ca_biplot()` [paquete factoextra] para dibujar el biplot de variables de filas y columnas (método de normalización Principal).

```
# repel= TRUE para evitar la superposición de texto (lento si hay muchos puntos)
fviz_ca_biplot(res.ca, repel = TRUE)
```



Este gráfico anterior el R lo denomina gráfico simétrico y muestra un patrón global dentro de los datos. Las filas están representadas por puntos azules y las columnas por triángulos rojos. La distancia entre cualquier punto fila (o columna) da una medida de su similaridad (o disimilaridad). Veremos después, los asimétricos, diseñados para poder comparar puntos fila y columna.

El siguiente paso será determinar qué perfiles de fila y columna contribuyen más en la definición de las diferentes dimensiones retenidas en el modelo.

2.5.5. Análisis de perfiles fila

Usaremos la función `get_ca_row()` [factoextra] para extraer los resultados de las variables de fila. Esta función devuelve una lista que contiene las coordenadas, el coseno cuadrado (calidad de la representación), la contribución y la inercia de las variables de fila.

Usaremos la función `fviz_ca_row()` [factoextra] para visualizar los resultados.

```
row <- get_ca_row(res.ca)
row
Correspondence Analysis - Results for rows
=====
  Name      Description
1 "$coord" "Coordinates for the rows"
2 "$cos2"   "Cos2 for the rows"
3 "$contrib" "contributions of the rows"
4 "$inertia" "Inertia of the rows"

# Coordenadas
row$coord
      Dim 1    Dim 2    Dim 3
Laundry -0.99184  0.49532 -0.316729
Main_meal -0.87559  0.49011 -0.164065
Dinner   -0.69257  0.30810 -0.207414
Breakfast -0.50860  0.45280  0.220405
Tidying   -0.39381 -0.43434 -0.094214
Dishes    -0.18896 -0.44197  0.266949
```

```

Shopping   -0.11768 -0.40332  0.202615
Official    0.22663  0.25361  0.923364
Driving     0.74177  0.65341  0.544458
Finances    0.27077 -0.61787  0.034797
Insurance   0.64708 -0.47378 -0.289361
Repairs     1.52878  0.86426 -0.472088
Holidays    0.25249 -1.43501 -0.129587

```

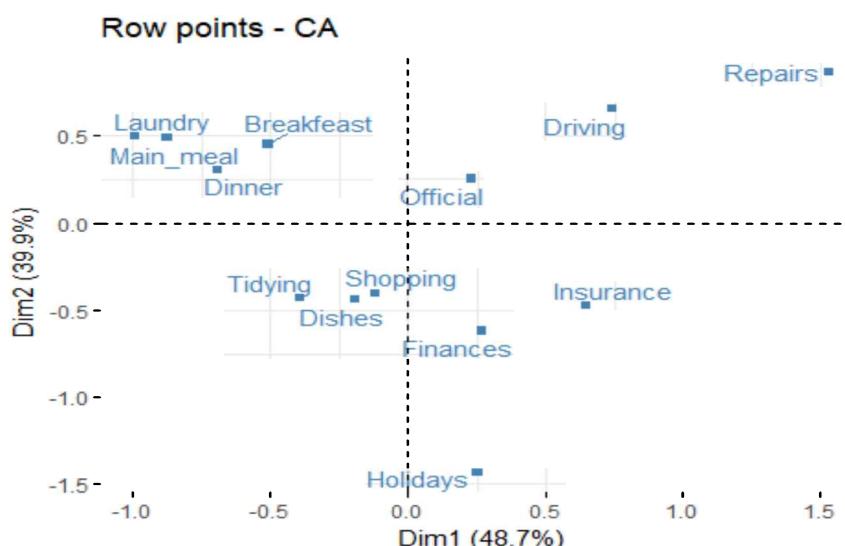
Cosenos al cuadrado: calidad en el mapa de factores. Contribución relativa
row\$cos2

	Dim 1	Dim 2	Dim 3
Laundry	0.739987	0.184552	0.0754605
Main_meal	0.741603	0.232359	0.0260379
Dinner	0.776640	0.153703	0.0696567
Breakfeast	0.504943	0.400230	0.0948267
Tidying	0.439812	0.535015	0.0251725
Dishes	0.118118	0.646153	0.2357297
Shopping	0.063654	0.747655	0.1886912
Official	0.053045	0.066426	0.8805289
Driving	0.432019	0.335229	0.2327523
Finances	0.160677	0.836670	0.0026536
Insurance	0.576012	0.308802	0.1151860
Repairs	0.706736	0.225871	0.0673928
Holidays	0.029792	0.962360	0.0078478

Contribuciones a los componentes principales
row\$contrib

	Dim 1	Dim 2	Dim 3
Laundry	18.28670	5.56389	7.968424
Main_meal	12.38884	4.73552	1.858689
Dinner	5.47140	1.32102	2.096926
Breakfeast	3.82493	3.69861	3.069399
Tidying	1.99835	2.96564	0.488734
Dishes	0.42617	2.84412	3.634294
Shopping	0.17552	2.51516	2.223357
Official	0.52078	0.79562	36.940389
Driving	8.07784	7.64686	18.596386
Finances	0.87501	5.55855	0.061751
Insurance	6.14706	4.02036	5.252639
Repairs	40.73009	15.88065	16.596391
Holidays	1.07730	42.45400	1.212620

Los parámetros col.row y shape.row hacen referencial al color y forma de los puntos
fviz_ca_row(res.ca, col.row="steelblue", shape.row = 15, repel = TRUE)



La gráfica anterior muestra las relaciones entre los puntos fila. Así, las filas con un perfil similar aparecerán juntas, mientras que las filas negativamente correlacionadas se colocan en lados opuestos del origen del gráfico (cuadrantes opuestos).

La distancia entre los puntos fila y el origen mide la calidad de la representación de los puntos fila en el mapa de factores. Los puntos fila que están lejos del origen están bien representados en el mapa de factores.

Calidad en la representación de los puntos fila

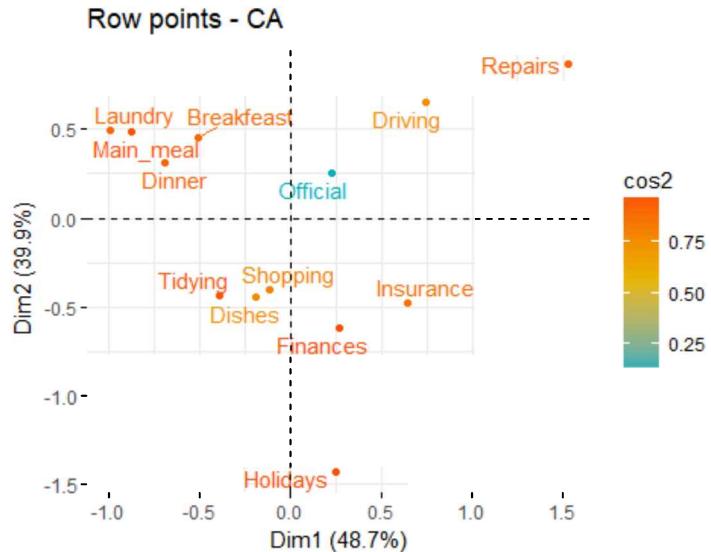
El resultado del análisis muestra que la tabla de contingencia se ha representado con éxito en el espacio de baja dimensión mediante el análisis de correspondencias. Las dos primeras dimensiones son suficientes para retener el 88.6% de la inercia total (variación) contenida en los datos. Sin embargo, no todos los puntos se muestran igualmente bien en las dos dimensiones.

Los cosenos al cuadrado mide el grado de asociación entre filas / columnas y un eje particular. Los valores están comprendidos entre 0 y 1. La suma de cosenos al cuadrado para filas en todas las dimensiones es igual a uno (todos los puntos bien representados). Los resultados para nuestro ejemplo serían:

	Dim 1	Dim 2	Dim 3
Laundry	0.739987	0.184552	0.0754605
Main_meal	0.741603	0.232359	0.0260379
Dinner	0.776640	0.153703	0.0696567
Breakfeast	0.504943	0.400230	0.0948267
Tidying	0.439812	0.535015	0.0251725
Dishes	0.118118	0.646153	0.2357297
Shopping	0.063654	0.747655	0.1886912
Official	0.053045	0.066426	0.8805289
Driving	0.432019	0.335229	0.2327523
Finances	0.160677	0.836670	0.0026536
Insurance	0.576012	0.308802	0.1151860
Repairs	0.706736	0.225871	0.0673928
Holidays	0.029792	0.962360	0.0078478

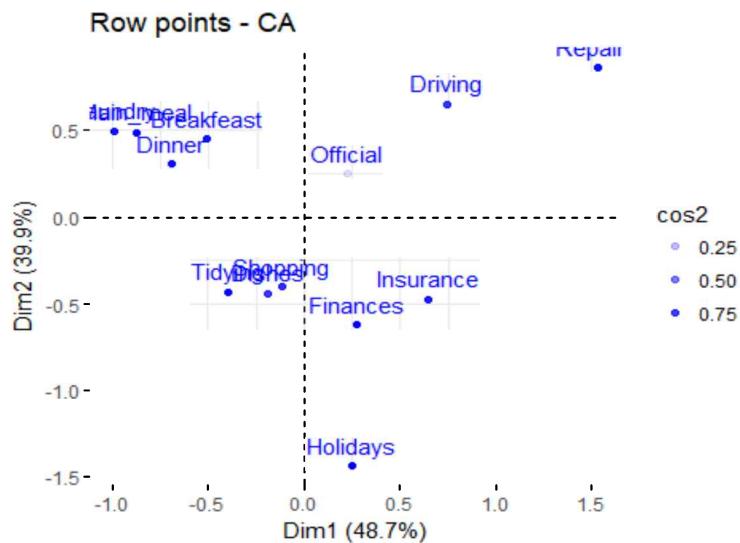
Es posible colorear los puntos fila por sus valores de cosenos al cuadrado usando el argumento col.row = "cos2". Esto produce un degradado de colores, que se puede personalizar con el argumento gradient.cols. Por ejemplo, gradient.cols = c ("white", "blue", "red") significa que los perfiles con valores bajos de cos2 se colorearán en "blanco", aquellos con valores medios en "azul" y los que tienen valores altos en rojo.

```
fviz_ca_row(res.ca, col.row = "cos2",
            gradient.cols = c("#00AFBB", "#E7B800", "#FC4E07"),
            repel = TRUE)
```



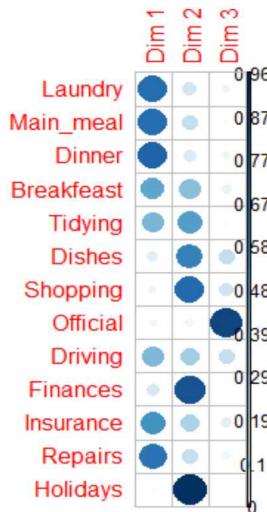
Otra opción sería cambiar la transparencia de los puntos fila:

```
fviz_ca_row(res.ca, alpha.row="cos2")
```

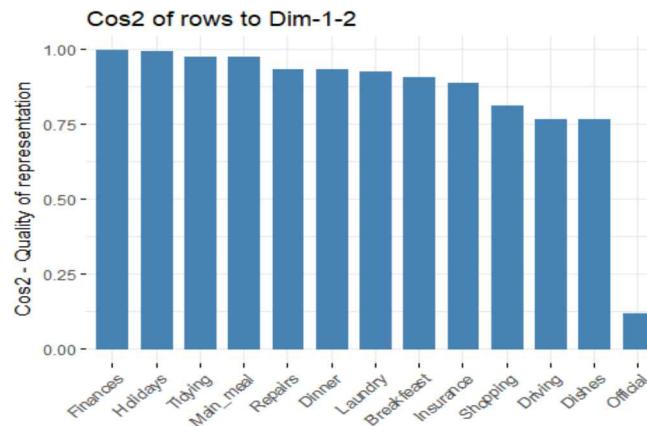


Usando el paquete **corrplot**, para visualizar los cosenos al cuadrado de los puntos fila en todas las dimensiones, y la función **fviz_cos2()**, para dibujar el gráfico de barras de la calidad en dos dimensiones, obtenemos:

```
library("corrplot")
corrplot(row$cos2, is.corr=FALSE)
```



```
fviz_cos2(res.ca, choice = "row", axes = 1:2)
```



Contribuciones de las filas a las dimensiones

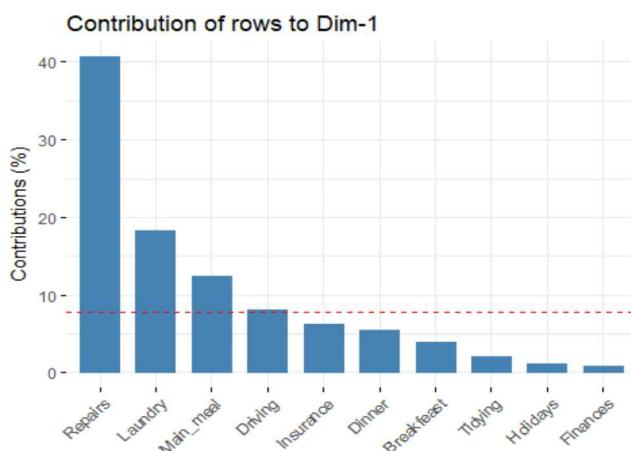
Analizamos a continuación la contribución de las filas (en %) a la definición de las dimensiones. Las filas que contribuyen más a Dim.1 y Dim.2 son las más importantes para explicar la variabilidad en el conjunto de datos. Las filas que no contribuyen mucho a ninguna dimensión o que contribuyen a la última dimensión son menos importantes.

	Dim 1	Dim 2	Dim 3
Laundry	18.28670	5.56389	7.968424
Main_meal	12.38884	4.73552	1.858689
Dinner	5.47140	1.32102	2.096926
Breakfeast	3.82493	3.69861	3.069399
Tidying	1.99835	2.96564	0.488734
Dishes	0.42617	2.84412	3.634294
Shopping	0.17552	2.51516	2.223357
Official	0.52078	0.79562	36.940389
Driving	8.07784	7.64686	18.596386
Finances	0.87501	5.55855	0.061751
Insurance	6.14706	4.02036	5.252639
Repairs	40.73009	15.88065	16.596391
Holidays	1.07730	42.45400	1.212620

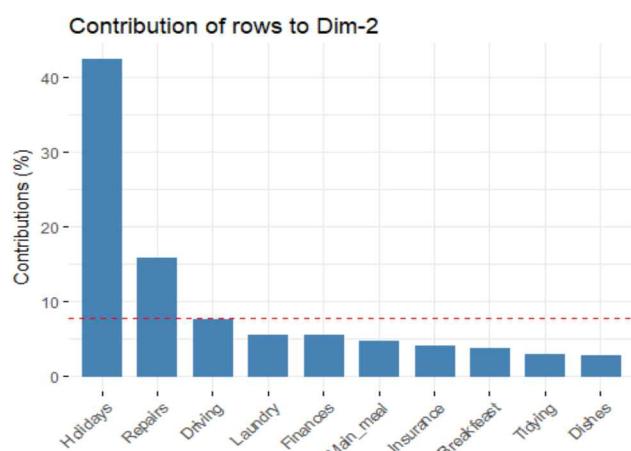
```
corrplot(row$contrib, is.corr=FALSE)
```



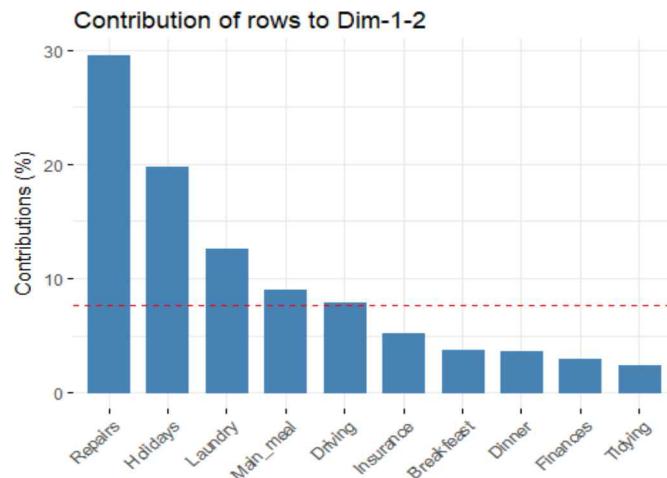
```
# Contribuciones de filas a la dimensión 1
fviz_contrib(res.ca, choice = "row", axes = 1, top = 10)
```



```
# Contribuciones de filas a la dimensión 2
fviz_contrib(res.ca, choice = "row", axes = 2, top = 10)
```

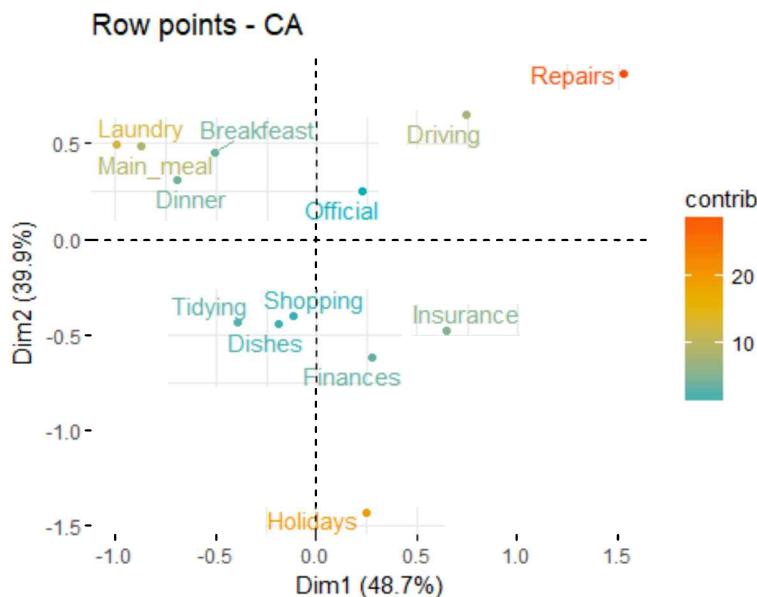


```
# Total contribución a las dimensiones 1 y 2
fviz_contrib(res.ca, choice = "row", axes = 1:2, top = 10)
```

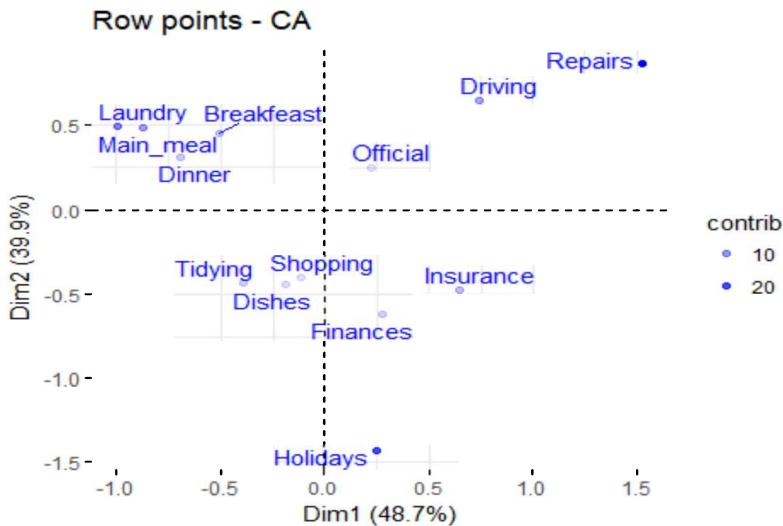


La línea roja discontinua indica el valor promedio esperado, si las contribuciones fueron uniformes. Según puede observarse, las tareas de Reparaciones, Lavandería, Comida principal y Conducción son las más importantes en la definición de la primera dimensión. Las Vacaciones y Reparaciones son las que más contribuyen a la segunda.

```
# Cambio en los colores según los valores de contribución
fviz_ca_row(res.ca, col.row = "contrib", gradient.cols = c("#00AFBB", "#E7B800",
"FC4E07"), repel = TRUE)
```



```
# Cambio en la transparencia según los valores de contribución
fviz_ca_row(res.ca, alpha.row="contrib", repel = TRUE)
```



2.5.6. Análisis de perfiles columna

Se realiza a continuación el análisis de los perfiles columna, utilizando para ello la función `get_ca_col()`.

```
col <- get_ca_col(res.ca)
col
  Correspondence Analysis - Results for columns
  =====
  Name      Description
  1 "$coord" "Coordinates for the columns"
  2 "$cos2"  "Cos2 for the columns"
  3 "$contrib" "contributions of the columns"
  4 "$inertia" "Inertia of the columns"

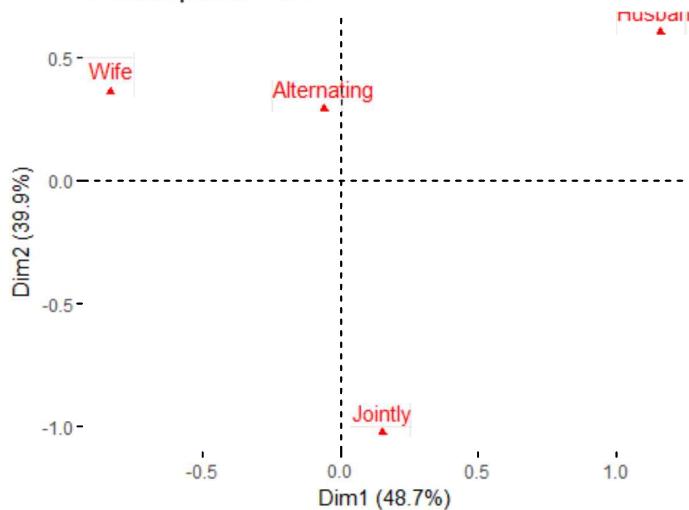
# Coordenadas de puntos columna
head(col$coord)
      Dim 1       Dim 2       Dim 3
Wife    -0.837622   0.36522  -0.199911
Alternating -0.062185   0.29159   0.848589
Husband   1.160918   0.60192  -0.188859
Jointly    0.149426  -1.02658  -0.046443

# Calidad de representación (contribución relativa)
head(col$cos2)
      Dim 1       Dim 2       Dim 3
Wife    0.8018759   0.15245  0.0456758
Alternating 0.0047799   0.10510  0.8901185
Husband   0.7720262   0.20754  0.0204317
Jointly    0.0207059   0.97729  0.0020002

# Contribuciones
head(col$contrib)
      Dim 1       Dim 2       Dim 3
Wife    44.46202  10.3122  10.8221
Alternating 0.10374   2.7828  82.5492
Husband   54.23388  17.7866  6.1332
Jointly    1.20036  69.1184  0.4955

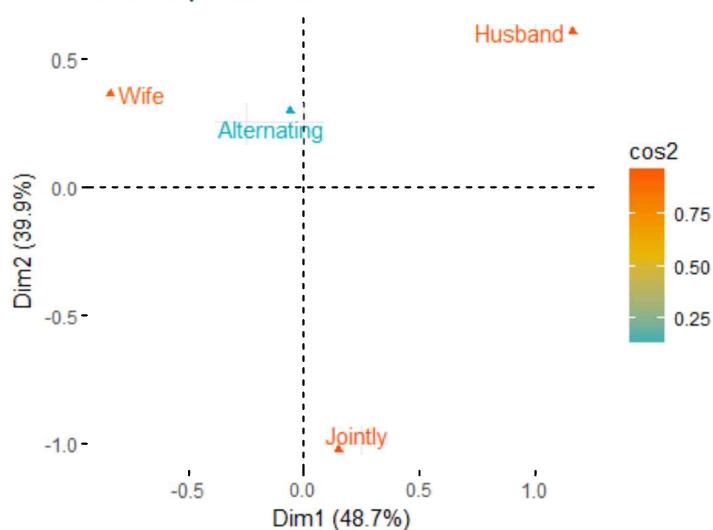
# Gráfico simple
fviz_ca_col(res.ca)
```

Column points - CA

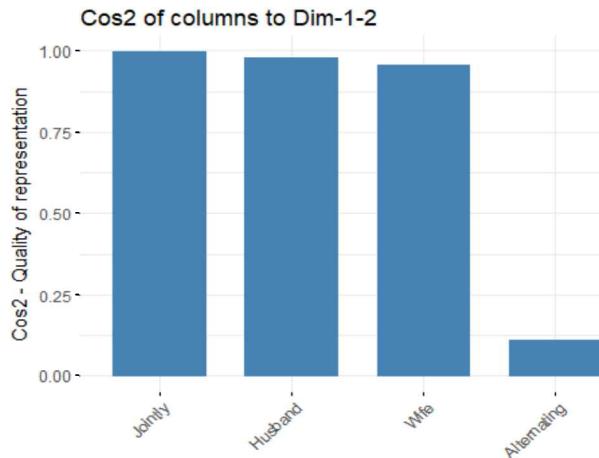


```
# Cambio de color según valores de cos2
fviz_ca_col(res.ca, col.col = "cos2", gradient.cols = c("#00AFBB", "#E7B800",
"FC4E07"),
repel = TRUE)
```

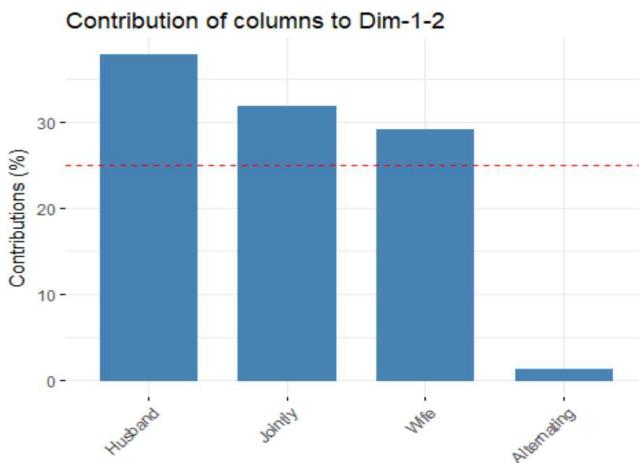
Column points - CA



```
# Gráfico de barras de cos2
fviz_cos2(res.ca, choice = "col", axes = 1:2)
```



```
# Gráfico de barras de contribuciones
fviz_contrib(res.ca, choice = "col", axes = 1:2)
```



La dimensión 1 contrapone a hombres y mujeres, mientras que la dimensión 2 separa las actividades realizadas conjuntamente de las que se hacen de forma individual.

2.5.7. Biplots asimétricos

Para hacer un biplot asimétrico, los puntos fila (o columna) se trazan a partir de las coordenadas estándar (S) y los perfiles de las columnas (o las filas) se grafican desde las coordenadas principales (P) (Bendixen, 2003).

Para un eje dado, las coordenadas estándar y principales se relacionan de la siguiente manera:

$$P = \sqrt{\text{valor propio}} \times S \quad [66]$$

- P : la coordenada principal de una fila (o una columna) en el eje
- valor propio: el valor propio del eje

Dependiendo de la situación, se pueden establecer otros tipos de visualización usando el argumento *map* (Nenadic y Greenacre 2007) de la función *fviz_ca_biplot()* [factoextra].

Las opciones permitidas para el argumento *map* son:

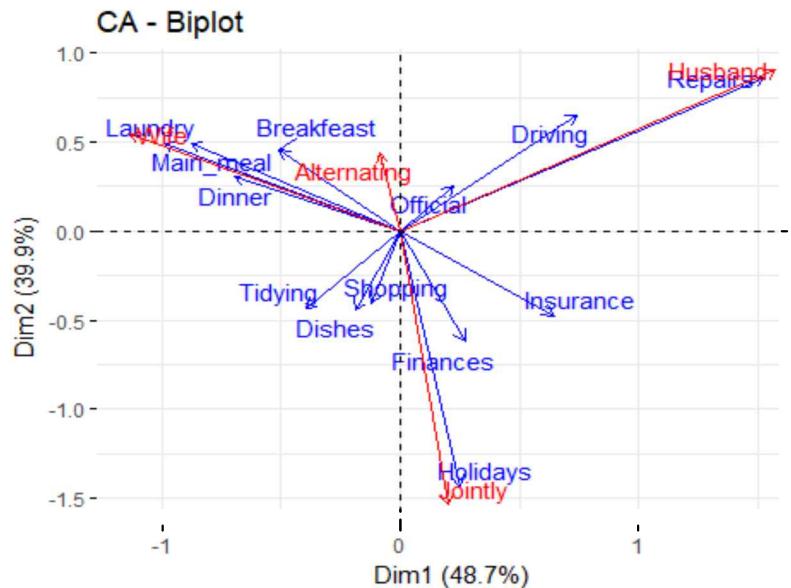
1. "rowprincipal" o "colprincipal". Normalización principal por fila/columna. Estos son los denominados *biplots asimétricos*, con filas en coordenadas principales y

columnas en coordenadas estándar, o viceversa (también conocido como row-metric-preserving o column-metric-preserving, respectivamente).

- "rowprincipal". Las columnas se representan en el espacio fila.
 - "colprincipal". Las filas están representadas en el espacio columna.
2. "symbiplot". Método simétrico. Tanto las filas como las columnas se escalan para tener varianzas iguales a los valores singulares (raíces cuadradas de valores propios), lo que da un *biplot simétrico* pero no conserva las métricas de fila o columna. Para comparar categorías de ambas variables, pero no las categorías de una misma variable.
 3. "rowgab" o "colgab". Mapas asimétricos propuestos por Gabriel y Odoroff (Gabriel y Odoroff, 1990).
 - "rowgab". Filas en coordenadas principales y columnas en coordenadas estándar multiplicadas por la masa.
 - "colgab". Columnas en coordenadas principales y filas en coordenadas estándar multiplicadas por la masa.
 4. "rowgreen" o "colgreen". Los llamados *biplots de contribución*, que muestran visualmente los puntos que más contribuyen (Greenacre, 2006b).
 - "rowgreen". Filas en coordenadas principales y columnas en coordenadas estándar multiplicadas por raíz cuadrada de la masa.
 - "colgreen": Columnas en coordenadas principales y filas en coordenadas estándar multiplicadas por la raíz cuadrada de la masa.

El siguiente código R dibuja un biplot asimétrico estándar:

```
fviz_ca_biplot(res.ca, map = "rowprincipal", arrow = c(TRUE, TRUE), repel = TRUE)
```



El argumento *arrow* es un vector con dos valores lógicos que especifica si la gráfica debe contener puntos (FALSE, predeterminado) o flechas (TRUE). El primer valor establece las filas y el segundo valor establece las columnas.

Si el ángulo entre dos flechas es agudo, entonces hay una fuerte asociación entre la fila y la columna correspondientes.

Para interpretar la distancia entre las filas y una columna, se debe proyectar perpendicularmente los puntos de fila en la flecha de la columna.

2.5.8. Biplot de contribución

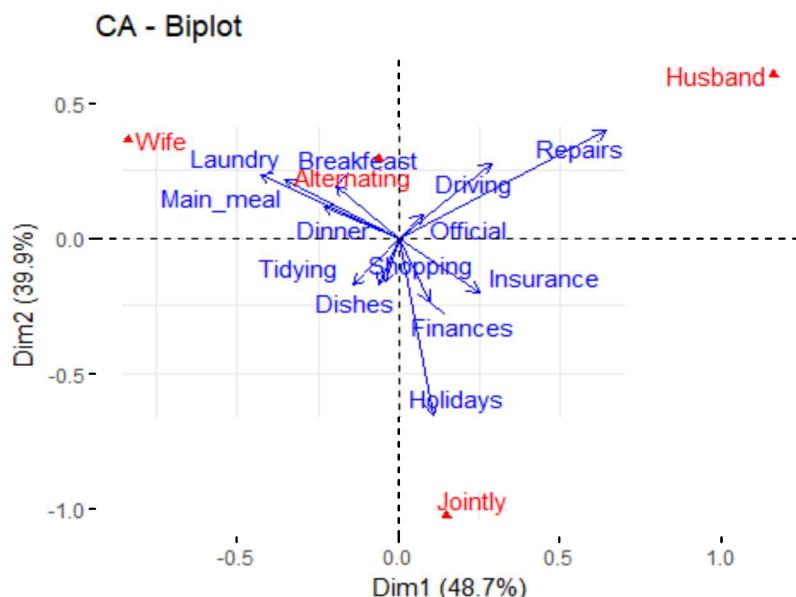
En el biplot simétrico estándar, es difícil conocer los puntos que más contribuyen a la solución del Análisis de Correspondencias.

Michael Greenacre propuso una nueva escala de visualización (el denominado *biplot de contribución*) que incorpora la contribución de los puntos (M. Greenacre, 2013). En esta visualización, los puntos que contribuyen muy poco a la solución, están cerca del centro del biplot y son relativamente poco importantes para la interpretación.

En primer lugar, debemos decidir si analizamos las contribuciones de filas o columnas en la definición de los ejes.

En nuestro ejemplo, interpretaremos la contribución de las filas a los ejes. Para ello, usaremos el argumento `map = "colgreen"`. En este caso, recuerde que las columnas están en coordenadas principales y las filas en coordenadas estándar multiplicadas por la raíz cuadrada de la masa. Para una fila dada, el cuadrado de la nueva coordenada en un eje i es exactamente la contribución de esta fila a la inercia del eje i .

```
fviz_ca_biplot(res.ca, map = "colgreen", arrow = c(TRUE, FALSE), repel = TRUE)
```



En el gráfico anterior, la posición de los puntos del perfil de la columna no se modifica con respecto a la del biplot convencional. Sin embargo, las distancias de los puntos de fila desde el origen del gráfico están relacionadas con sus contribuciones al mapa de factores bidimensional.

Cuanto más cerca esté una flecha (en términos de distancia angular) a un eje, mayor será la contribución de la categoría de fila en ese eje con relación al otro eje. Si la flecha está a medio camino entre los dos, esa categoría de fila contribuirá a los dos ejes en la misma medida.

- Es evidente que la categoría de filas correspondiente a Reparaciones tiene una contribución importante al polo positivo de la primera dimensión, mientras que las categorías Lavandería y Comida principal tienen una contribución importante al polo negativo.
- La dimensión 2 está principalmente definida por la categoría de fila Vacaciones.
- La categoría Conducción contribuye a los dos ejes en la misma medida.

2.5.9. Descripción de la dimensión

Para identificar fácilmente los puntos fila y columna que están más asociados con las dimensiones principales, puede usarse la función `dimdesc()` [FactoMineR]. Las variables de fila / columna se ordenan por sus coordenadas en la salida de `dimdesc()`.

```
# Descripción de la dimensión
res.desc <- dimdesc(res.ca, axes = c(1,2))

# Descripción de la dimensión 1 por puntos fila
res.desc[[1]]$row
      coord
Laundry      -0.99184
Main_meal    -0.87559
Dinner       -0.69257
Breakfeast   -0.50860
Tidying      -0.39381
Dishes        -0.18896
Shopping     -0.11768
Official     0.22663
Holidays      0.25249
Finances      0.27077
Insurance    0.64708
Driving       0.74177
Repairs       1.52878

# Descripción de la dimensión 1 por puntos columna
res.desc[[1]]$col
      coord
Wife          -0.837622
Alternating   -0.062185
Jointly       0.149426
Husband       1.160918

# Descripción de la dimensión 2 por puntos fila
res.desc[[2]]$row
      coord
Holidays     -1.43501
Finances      -0.61787
Insurance    -0.47378
Dishes        -0.44197
Tidying       -0.43434
Shopping     -0.40332
Official     0.25361
Dinner        0.30810
Breakfeast   0.45280
Main_meal    0.49011
Laundry       0.49532
Driving       0.65341
Repairs       0.86426

# Descripción de la dimensión 2 por puntos columna
res.desc[[2]]$col
      coord
Jointly      -1.02658
Alternating   0.29159
Wife          0.36522
Husband       0.60192
```

2.6. EJEMPLO DE ANÁLISIS DE CORRESPONDENCIAS MÚLTIPLE EN R

Las funciones disponibles en el software R para calcular el Análisis de Correspondencias Múltiple son las siguientes:

- MCA () [paquete *FactoMineR*]
- dudi.mca () [paquete *ade4*]
- epMCA () [paquete *ExPosition*]

Al igual que en el caso del Análisis de Correspondencias Simple, independientemente de la función que se decida utilizar, se pueden extraer y visualizar fácilmente los resultados del análisis utilizando las funciones R proporcionadas en el paquete *factoextra*.

De manera análoga, usaremos FactoMineR (para el análisis) y factoextra (para la visualización basada en ggplot2).

2.6.1. Análisis exploratorio

Para nuestro ejemplo, usaremos el data set de ejemplo *poisson* disponible en el paquete FactoMineR. La información que contiene este fichero es la relativa a una encuesta realizada a niños de la escuela primaria que sufrieron intoxicación alimentaria. Se les preguntó sobre sus síntomas y sobre lo que comieron.

El fichero está estructurado en 55 filas (individuos) y 15 columnas (variables). Usaremos solamente algunos de estos individuos (niños) y variables para realizar el Análisis de Correspondencias Múltiple. Las coordenadas de los individuos restantes y las variables en el mapa de factores se predecirán a partir de los resultados derivados del análisis MCA (Multiple Correspondence Analysis).

En la terminología MCA, nuestros datos contienen:

- Individuos activos (filas 1:55). Individuos que se utilizan en el Análisis de Correspondencias Múltiple.
- Variables activas (columnas 5:15). Variables que se usan en el MCA.
- Variables suplementarias. No participan en el ACM. Las coordenadas de estas variables serán predichas.
 - Variables cuantitativas suplementarias (quanti.sup). Columnas 1 y 2 correspondientes a las columnas edad y tiempo, respectivamente.
 - Variables cualitativas suplementarias (quali.sup). Columnas 3 y 4 correspondientes a las columnas Sick (enfermos) y Sex (sexo), respectivamente. Estas variables se utilizarán para colorear individuos por grupos.

```
library("FactoMineR")
library("factoextra")
data(poison)

# Data set que utilizaremos para el análisis
poison.active <- poison[1:55, 5:15]
```

En primer lugar, realizaremos un análisis exploratorio de los datos mediante un resumen sumario y gráficos de barras. Las categorías de variables con una frecuencia muy baja pueden distorsionar el análisis y, por tanto, dichas variables deberían eliminarse.

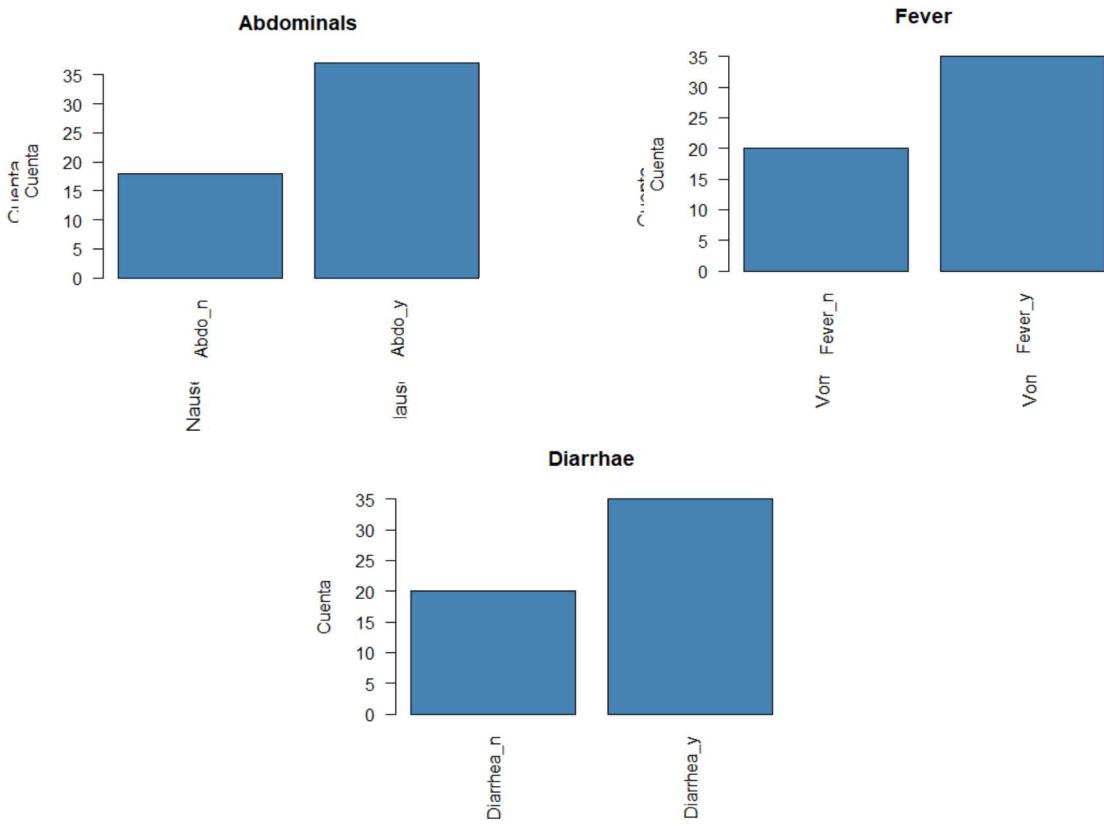
```
# Estadística descriptiva de los datos
summary(poison)
```

Age	Time	Sick	Sex	Nausea
Min. : 4.0	Min. : 0.0	Sick_n:17	F:28	Nausea_n:43
1st Qu.: 6.0	1st Qu.: 0.0	Sick_y:38	M:27	Nausea_y:12
Median : 8.0	Median :12.0			
Mean :16.9	Mean :10.2			
3rd Qu.:10.0	3rd Qu.:16.5			
Max. :88.0	Max. :22.0			

Vomiting	Abdominals	Fever	Diarrhoea	Potato
Vomit_n:33	Abdo_n:18	Fever_n:20	Diarrhea_n:20	Potato_n: 3
Vomit_y:22	Abdo_y:37	Fever_y:35	Diarrhea_y:35	Potato_y:52

Fish	Mayo	Courgette	Cheese	Icecream
Fish_n: 1	Mayo_n:10	Courg_n: 5	Cheese_n: 7	Icecream_n: 4
Fish_y:54	Mayo_y:45	Courg_y:50	Cheese_y:48	Icecream_y:51

```
# Gráficos de síntomas
for (i in 1:5) {
  plot(poison.active[,i], main=colnames(poison.active)[i], ylab = "Cuenta",
       col="steelblue", las = 2)
}
```



2.6.2. Estimación del modelo

Para realizar el análisis usaremos la función *MCA* () [paquete FactoMiner], con parámetros simplificados:

CA (X, ncp = 5, graph = TRUE)

- X: un marco de datos con n filas (individuos) y p columnas (variables categóricas).
- ncp: número de dimensiones guardadas en los resultados finales (por defecto, 5).
- graph: un valor lógico. Si es TRUE (por defecto), se muestra un gráfico.

```

library("FactoMineR")
res.mca <- MCA(poison.active, graph = FALSE)
print(res.mca)
  **Results of the Multiple Correspondence Analysis (MCA)**
  The analysis was performed on 55 individuals, described by 11 variables
  *The results are available in the following objects:

    name           description
 1  "$eig"         "eigenvalues"
 2  "$var"         "results for the variables"
 3  "$var$coord"   "coord. of the categories"
 4  "$var$cos2"    "cos2 for the categories"
 5  "$var$contrib" "contributions of the categories"
 6  "$var$v.test"  "v-test for the categories"
 7  "$ind"         "results for the individuals"
 8  "$ind$coord"   "coord. for the individuals"
 9  "$ind$cos2"    "cos2 for the individuals"
10  "$ind$contrib" "contributions of the individuals"
11  "$call"        "intermediate results"
12  "$call$marge.col" "weights of columns"
13  "$call$marge.li"  "weights of rows"

```

Usaremos las siguientes funciones [en factoextra] para ayudar en la interpretación y la visualización del análisis de correspondencia:

- *get_eigenvalue(res.mca)*: Extrae los valores propios / varianzas retenidas por cada dimensión (eje).
- *fviz_eig(res.mca)*: Visualiza los valores propios.
- *get_mca_ind(res.mca), get_mca_var(res.mca)*: Extrae los resultados para individuos y variables, respectivamente.
- *fviz_mca_ind(res.mca), fviz_mca_var(res.mca)*: Visualiza los resultados para individuos y variables, respectivamente.
- *fviz_mca_biplot(res.mca)*: crea un biplot de filas y columnas.

2.6.3. Valores propios

Calculamos a continuación la proporción de varianza retenida por las diferentes dimensiones (ejes). Para visualizar los porcentajes de inercia explicados por cada dimensión de MCA, se puede usar la función *fviz_eig()* o *fviz_screeplot()*.

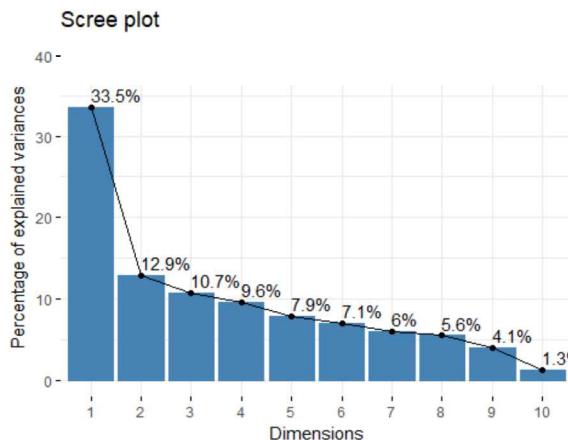
Como vemos en los resultados, alrededor del 88.6% de la variación se explica por las dos primeras dimensiones.

```

library("factoextra")
eig.val <- get_eigenvalue(res.mca)
eig.val
  eigenvalue  variance.percent cumulative.variance.percent
Dim.1      0.335231     33.523                33.523
Dim.2      0.129140     12.9140               46.437
Dim.3      0.107348     10.7348               57.172
Dim.4      0.095880      9.5880               66.760
Dim.5      0.078833      7.8833               74.643
Dim.6      0.071090      7.1090               81.752
Dim.7      0.060166      6.0166               87.769
Dim.8      0.055773      5.5773               93.346
Dim.9      0.041206      4.1206               97.467
Dim.10     0.013042      1.3042              98.771
Dim.11     0.012292      1.2292              100.000

fviz_screeplot(res.mca, addlabels = TRUE, ylim = c(0, 40))

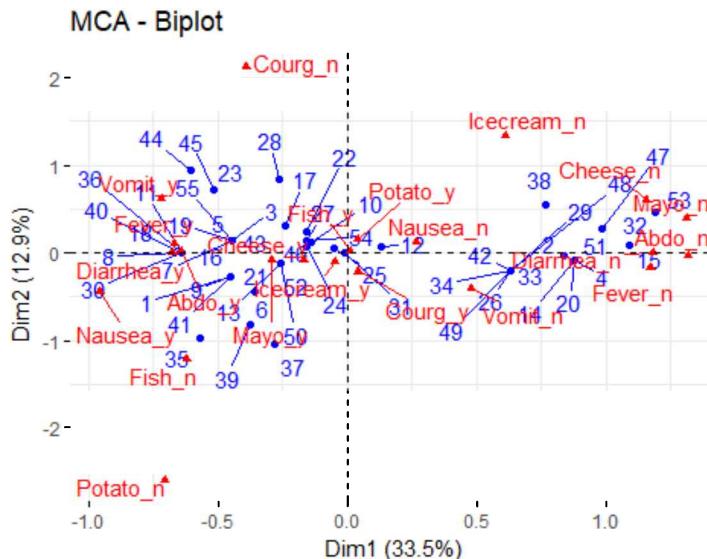
```



2.6.4. Biplot simétrico

Usaremos la función `fviz_mca_biplot()` [paquete factoextra] para dibujar el biplot de variables de individuos y categorías de las variables (método de normalización Principal). Los puntos azules representan a los individuos y los triángulos verdes a las categorías de las variables. La distancia entre cualquier punto de fila o columna da una medida de su similaridad (o disimilaridad). Los puntos de fila con un perfil similar están cercanos en el mapa de factores. Lo mismo es cierto para los puntos de columna.

```
# repel= TRUE para evitar la superposición de texto (lento si hay muchos puntos)
fviz_mca_biplot(res.mca, repel = TRUE, ggtheme = theme_minimal())
```



2.6.5. Análisis de las variables

Usaremos la función `get_mca_var()` [factoextra] para extraer los resultados de las categorías de las variables. Esta función devuelve una lista que contiene las coordenadas, el coseno cuadrado (calidad de la representación) y la contribución de las categorías de las variables.

Usaremos la función `fviz_mca_var()` [factoextra] para visualizar los resultados.

```

var <- get_mca_var(res.mca)
var
  Multiple Correspondence Analysis Results for variables
  =====
  Name      Description
  1 "$coord" "Coordinates for categories"
  2 "$cos2"   "Cos2 for categories"
  3 "$contrib" "contributions of categories"

# Coordenadas (primeras)
head(var$coord)
  Dim 1      Dim 2      Dim 3      Dim 4      Dim 5
Nausea_n  0.26739  0.121390 -0.2655833  0.033761  0.073705
Nausea_y -0.95815 -0.434982  0.9516733 -0.120978 -0.264110
Vomit_n   0.47903 -0.409195  0.0844928  0.273611  0.052453
Vomit_y   -0.71854  0.613792 -0.1267392 -0.410417 -0.078679
Abdo_n    1.31802 -0.035745 -0.0050942 -0.153610 -0.069870
Abdo_y   -0.64120  0.017389  0.0024783  0.074729  0.033991

# Cosenos al cuadrado: calidad en el mapa de factores. Contribución relativa
head(var$cos2)
  Dim 1      Dim 2      Dim 3      Dim 4      Dim 5
Nausea_n  0.25620  0.05280258 2.5275e-01  0.0040844  0.0194662
Nausea_y  0.25620  0.05280258 2.5275e-01  0.0040844  0.0194662
Vomit_n   0.34420  0.25116039 1.0709e-02  0.1122948  0.0041269
Vomit_y   0.34420  0.25116039 1.0709e-02  0.1122948  0.0041269
Abdo_n    0.84512  0.00062159 1.2625e-05  0.0114791  0.0023749
Abdo_y   0.84512  0.00062159 1.2625e-05  0.0114791  0.0023749

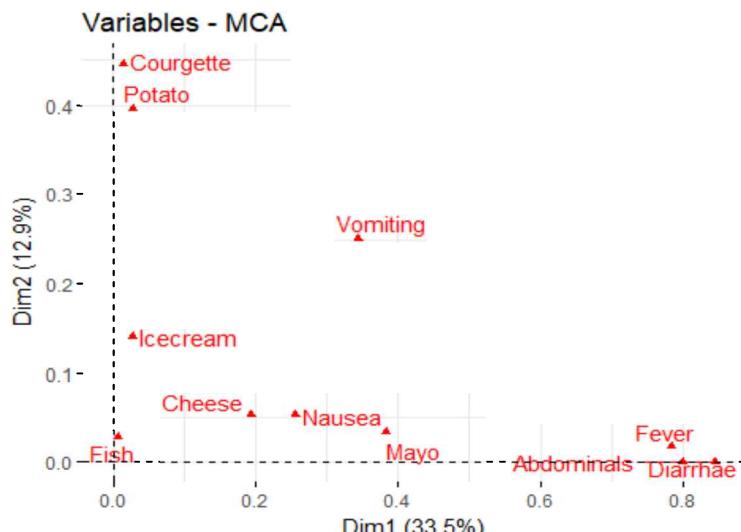
# Contribuciones a los componentes principales
head(var$contrib)
  Dim 1      Dim 2      Dim 3      Dim 4      Dim 5
Nausea_n  1.5159  0.811000 4.6700e+00  0.084494  0.489779
Nausea_y  5.4319  2.906084 1.6734e+01  0.302770  1.755042
Vomit_n   3.7337  7.072263 3.6275e-01  4.258937  0.190364
Vomit_y   5.6005  10.608394 5.4412e-01  6.388406  0.285546
Abdo_n    15.4176  0.029437 7.1925e-04  0.732196  0.184243
Abdo_y    7.5005  0.014321 3.4991e-04  0.356204  0.089632

```

Correlación entre variables y dimensiones principales

La forma de visualizar las correlaciones entre las variables y las dimensiones principales del MCA es la siguiente:

```
fviz_mca_var(res.mca, choice = "mca.cor", repel = TRUE, ggtheme = theme_minimal())
```



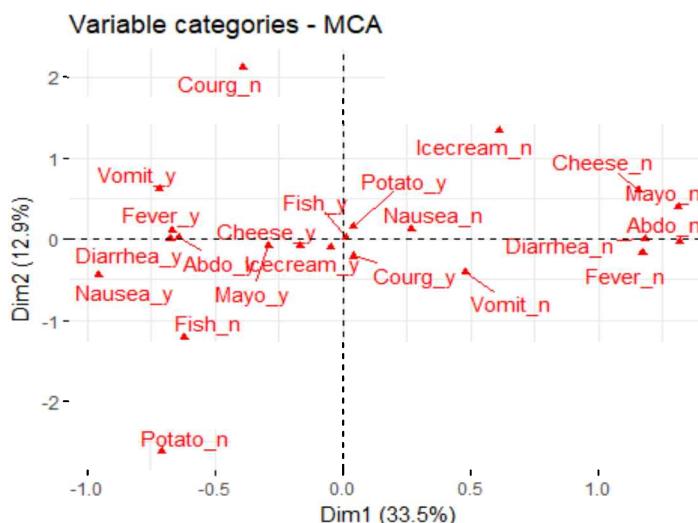
Con este gráfico se busca identificar las variables que están más correlacionadas con cada dimensión. Las correlaciones al cuadrado entre las variables y las dimensiones se usan como coordenadas.

Se puede observar que las variables Diarrhoe (diarrea), Abdominals (dolor abdominal) y Fever (fiebre) son las más correlacionadas con la dimensión 1. Del mismo modo, las variables Courgette (calabacín) y Potato (patata) son las más correlacionadas con la dimensión 2.

Coordenadas de categorías de variables

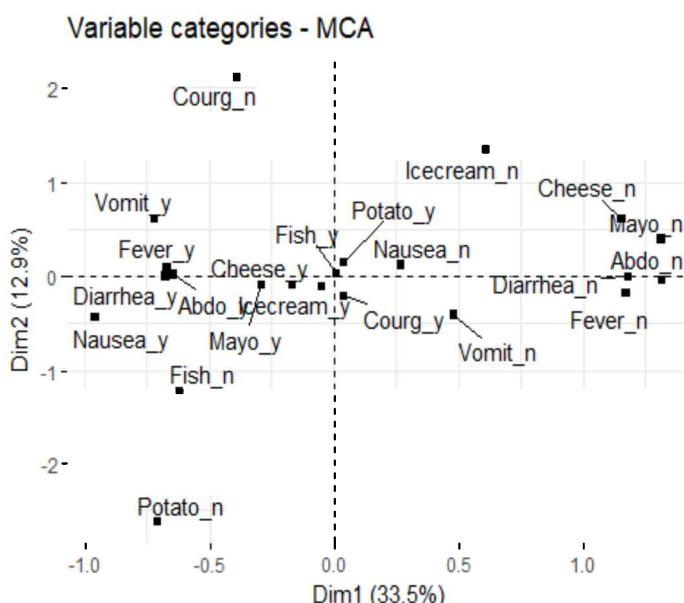
Representamos a continuación los puntos asociados a las categorías de las variables utilizando la función `fviz_mca_var()`.

```
fviz_mca_var(res.mca, repel = TRUE, ggtheme = theme_minimal())
```



Es posible cambiar el color y la forma de los puntos usando los argumentos `col.var` y `shape.var` de la siguiente manera.

```
fviz_mca_var(res.mca, col.var="black", shape.var = 15, repel = TRUE)
```



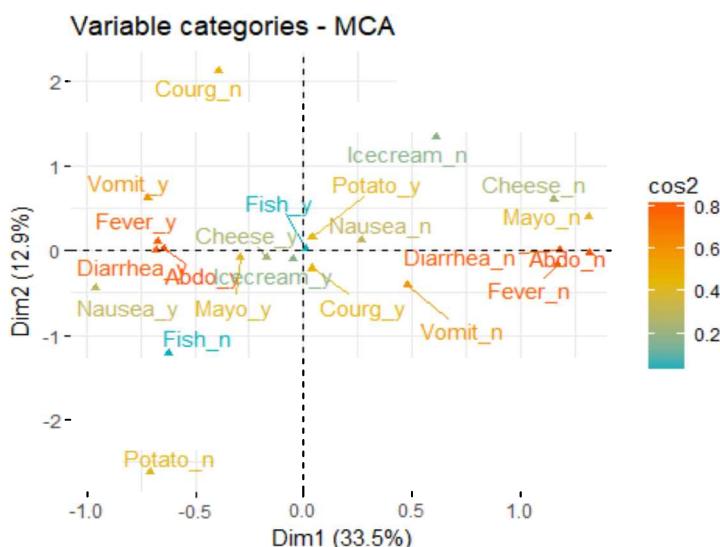
La gráfica anterior muestra las relaciones entre las categorías de variables. Se puede interpretar de la siguiente manera:

- Las categorías de las variables con un perfil similar se agrupan juntas.
- Las categorías de variables negativamente correlacionadas se ubican en lados opuestos del origen del gráfico (cuadrantes opuestos).
- La distancia entre los puntos y el origen mide la calidad de la categoría de la variable en el mapa de factores. Los puntos de categoría que están lejos del origen están bien representados en el mapa de factores.

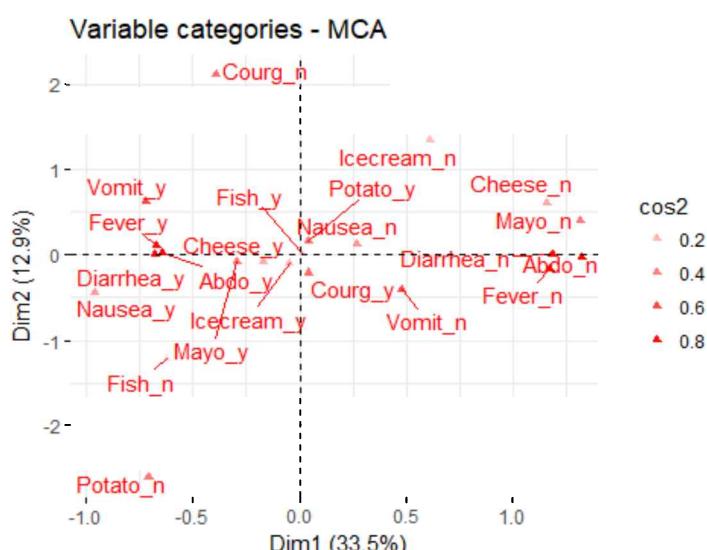
Calidad en la representación de las categorías de las variables

Se muestra a continuación las diferentes formas de analizar los valores de cos2.

```
# Color según valores de cos2: calidad sobre el mapa de factores
fviz_mca_var(res.mca, col.var = "cos2", gradient.cols = c("#00AFBB", "#E7B800",
"#"FC4E07"), repel = TRUE, ggtheme = theme_minimal())
```

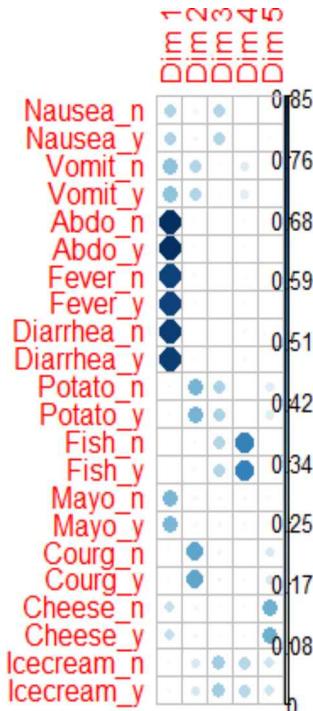


```
# Cambio de transparencia según valores de cos2
fviz_mca_var(res.mca, alpha.var="cos2", repel = TRUE, ggtheme = theme_minimal())
```

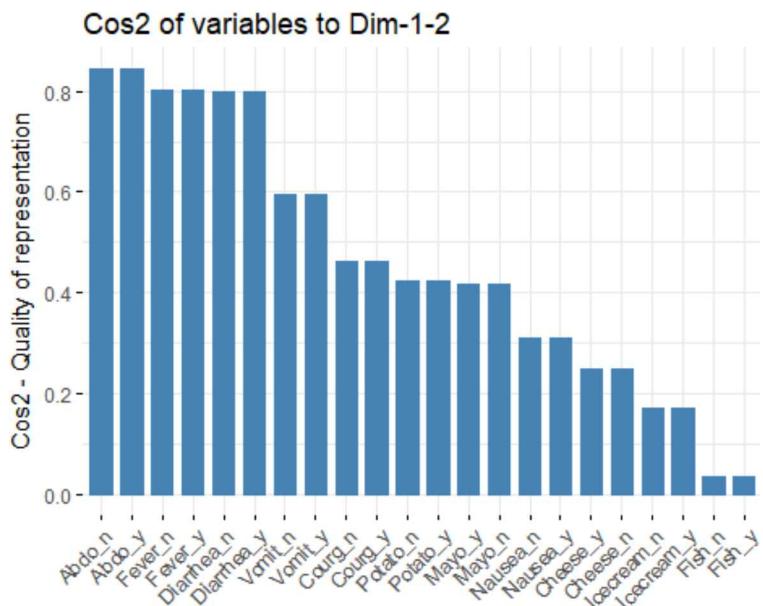


Usamos el paquete *corrplot* para visualizar los cosenos al cuadrado en todas las dimensiones, y la función *fviz_cos2()* para dibujar el gráfico de barras de la calidad en dos dimensiones, obteniendo:

```
library("corrplot")
corrplot(var$cos2, is.corr=FALSE)
```



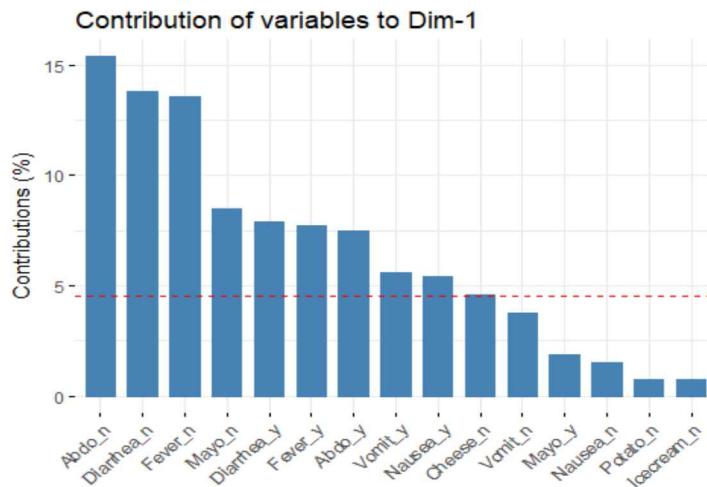
```
fviz_cos2(res.mca, choice = "var", axes = 1:2)
```



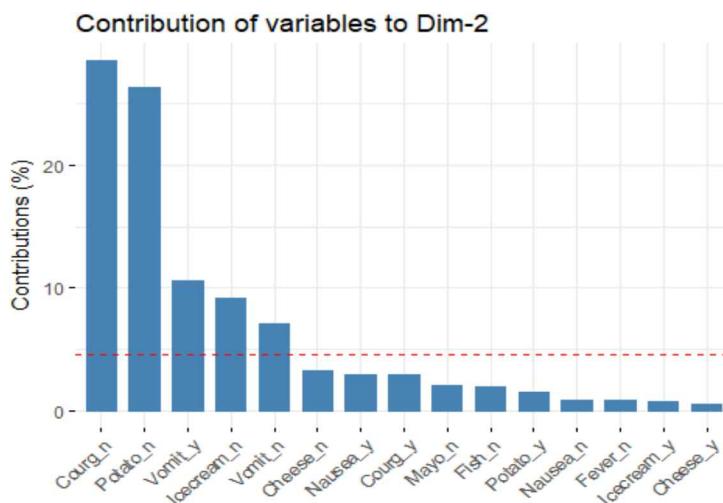
Contribuciones de las filas a las dimensiones

Analizamos a continuación la contribución de las variables (en %) a la definición de las dimensiones.

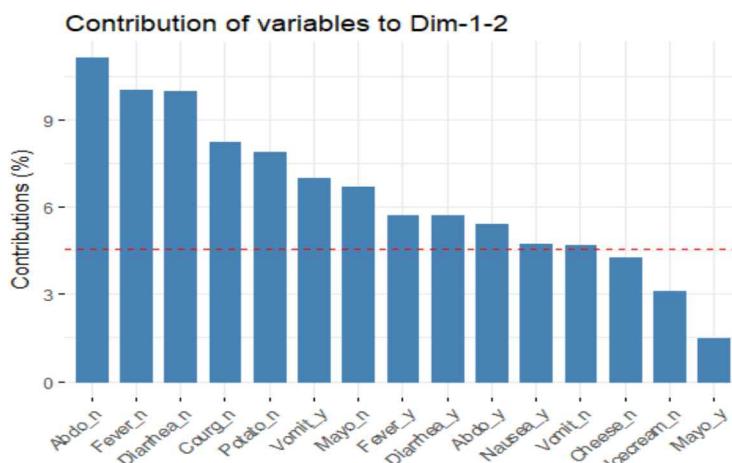
```
# Contribuciones de las categorías de las variables a la dimensión 1
fviz_contrib(res.mca, choice = "var", axes = 1, top = 15)
```



```
# Contribuciones de las categorías de las variables a la dimensión 2
fviz_contrib(res.mca, choice = "var", axes = 2, top = 15)
```



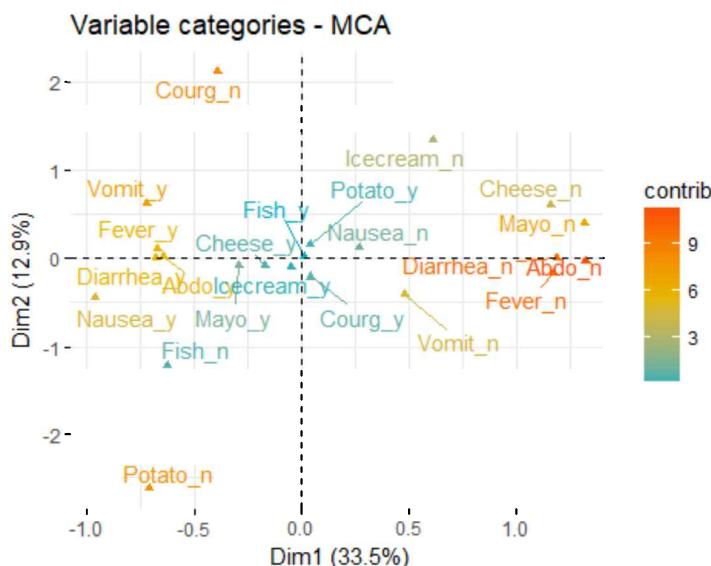
```
# Total contribución a las dimensiones 1 y 2
fviz_contrib(res.mca, choice = "var", axes = 1:2, top = 15)
```



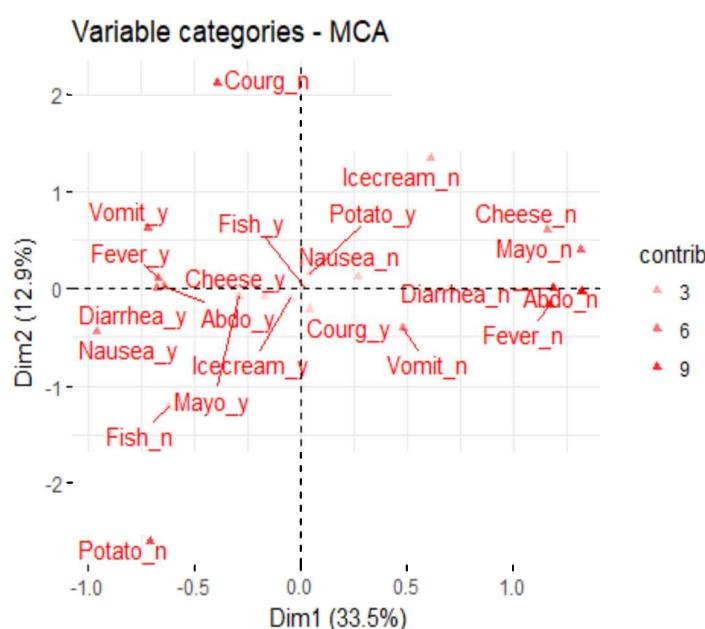
La línea roja discontinua indica el valor promedio esperado, si las contribuciones fueran uniformes. Según puede observarse, las categorías Abdo_n, Diarrhea_n, Fever_n and Mayo_n son las más importantes en la definición de la primera dimensión. Las categorías Courg_n, Potato_n, Vomit_y y Icecream_n son las que más contribuyen a la dimensión 2.

Las categorías de variables más importantes (o contribuyentes) se pueden resaltar en el diagrama de dispersión de las siguientes maneras.

```
# Cambio en los colores según los valores de contribución
fviz_mca_var(res.mca, col.var = "contrib", gradient.cols = c("#00AFBB", "#E7B800",
"FC4E07"), repel = TRUE, ggtheme = theme_minimal())
```



```
# Cambio en la transparencia según los valores de contribución
fviz_mca_var(res.mca, alpha.var="contrib", repel = TRUE, ggtheme = theme_minimal())
```



Es evidente que las categorías Abdo_n, Diarrhea_n, Fever_n y Mayo_n tienen una contribución importante al polo positivo de la primera dimensión, mientras que las categorías Fever_y y Diarrhea_y tienen una contribución importante al polo negativo; etc.

2.6.6. Análisis de los individuos

Se realiza a continuación el análisis de los individuos, utilizando para ello la función `get_mca_ind()`. La información derivada es la misma que para las categorías de las variables.

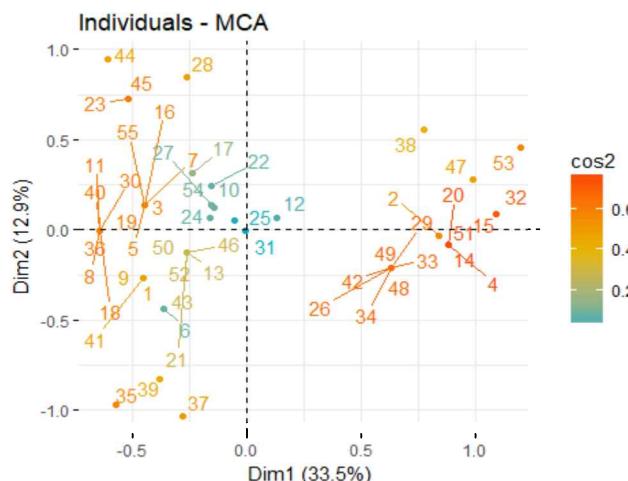
```
ind <- get_mca_ind(res.mca)
ind
  Multiple Correspondence Analysis Results for individuals
  -----
  Name      Description
1 "$coord" "Coordinates for the individuals"
2 "$cos2"   "Cos2 for the individuals"
3 "$contrib" "contributions of the individuals"

# Coordenadas
head(ind$coord)
  Dim 1     Dim 2     Dim 3     Dim 4     Dim 5
1 -0.45258 -0.264151 0.171516 0.013693 -0.116968
2 0.83617 -0.031935 -0.072082 -0.085504 0.519787
3 -0.44819 0.135387 -0.224840 -0.141702 -0.050048
4 0.88037 -0.085362 -0.020520 -0.072759 -0.229350
5 -0.44819 0.135387 -0.224840 -0.141702 -0.050048
6 -0.35943 -0.436044 -1.209322 1.724646 0.043482

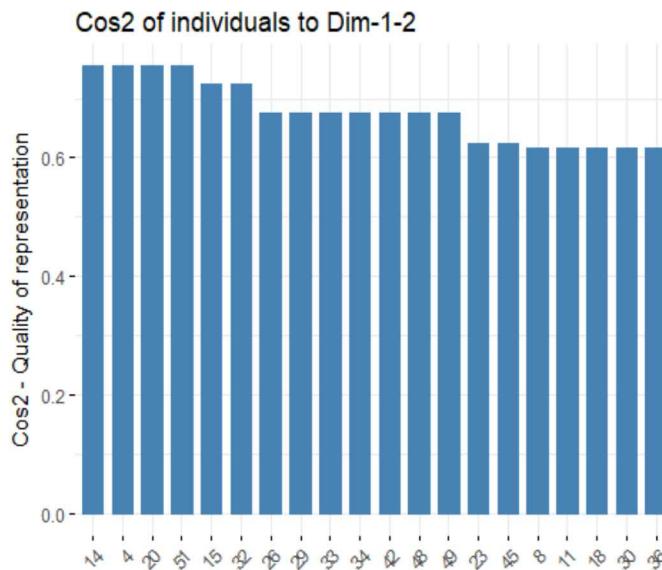
# Calidad de representación (contribución relativa)
head(ind$cos2)
  Dim 1     Dim 2     Dim 3     Dim 4     Dim 5
1 0.346526 0.11804472 0.04976832 0.00031723 0.02314608
2 0.555896 0.00081082 0.00413108 0.00581262 0.21481031
3 0.548139 0.05001768 0.13794849 0.05479209 0.00683492
4 0.747740 0.00702996 0.00040625 0.00510729 0.05074799
5 0.548139 0.05001768 0.13794849 0.05479209 0.00683492
6 0.024854 0.03657755 0.28134437 0.57220832 0.00036372

# Contribuciones
head(ind$contrib)
  Dim 1     Dim 2     Dim 3     Dim 4     Dim 5
1 1.11093 0.982383 0.4982547 0.0035558 0.315548
2 3.79212 0.014358 0.0880037 0.1386371 6.231341
3 1.08947 0.258067 0.8562299 0.3807690 0.057769
4 4.20361 0.102591 0.0071321 0.1003880 1.213190
5 1.08947 0.258067 0.8562299 0.3807690 0.057769
6 0.70069 2.676934 24.7699687 56.4042145 0.043605

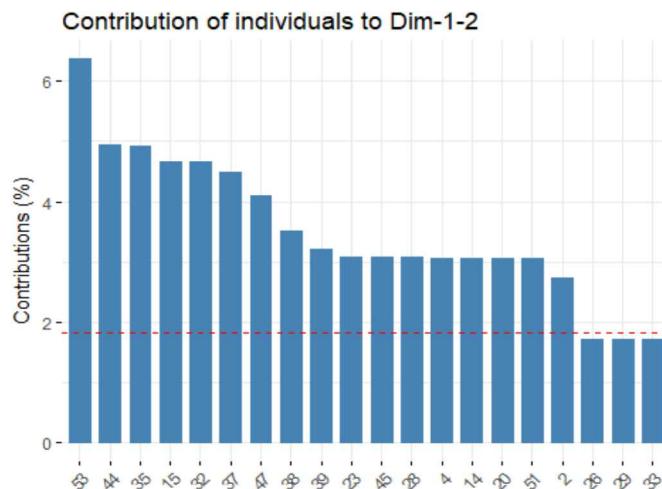
# Cambio de color según valores de cos2
fviz_mca_ind(res.mca, col.ind = "cos2", gradient.cols = c("#00AFBB", "#E7B800",
"##FC4E07"), repel = TRUE, ggtheme = theme_minimal())
```



```
# Gráfico de barras de cos2
fviz_cos2(res.mca, choice = "ind", axes = 1:2, top = 20)
```



```
# Contribución de los individuos a las dimensiones
fviz_contrib(res.mca, choice = "ind", axes = 1:2, top = 20)
```



2.6.7. Coloreando individuos por grupos

Vamos a colorear los individuos por grupos utilizando los niveles de la variable Vomiting (Vómitos).

Con el argumento "habillage" especificamos la variable factor que define los grupos, pudiendo ser especificada por nombre o índice de columna. Las dos siguientes expresiones son equivalentes:

- habillage = "Vomiting"
- habillage = 2

También sería posible especificar una variable externa, como por ejemplo:

- habillage = poison\$Vomiting.

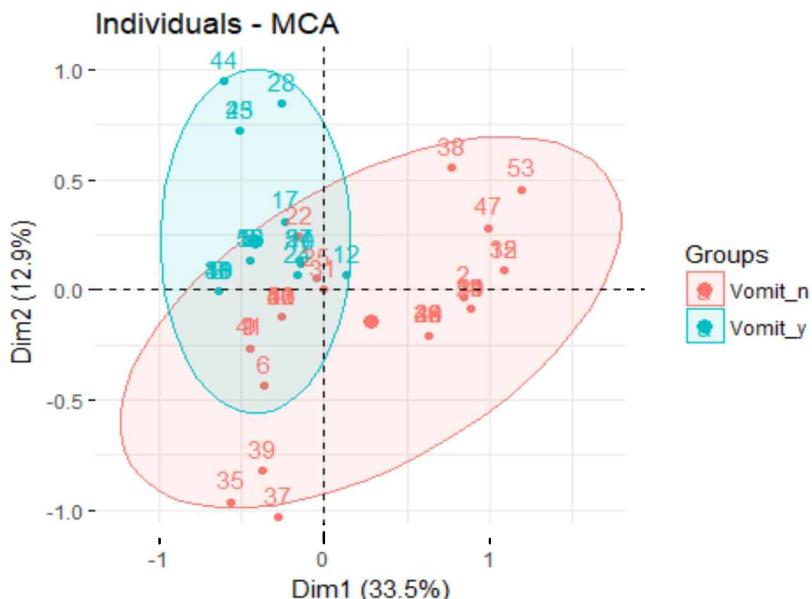
El argumento "addEllipses" agrega una elipse de concentración alrededor de cada grupo. Por defecto, se generarán elipses alrededor de los individuos. Si lo que buscamos es una elipse de confianza alrededor del punto medio de las categorías, debemos utilizar el argumento `ellipse.type = "confidence"`.

Finalmente, con el argumento "palette" se definen los colores de cada grupo.

```
# Con elipses de confianza y habillage = índice de columna
fviz_mca_ind(res.mca,
              label = "none", # oculta las etiquetas de los individuos
              habillage = 2, # color por grupos
              palette = c("#00AFBB", "#E7B800"),
              addEllipses = TRUE, ellipse.type = "confidence", ggtheme =
theme_minimal())
```

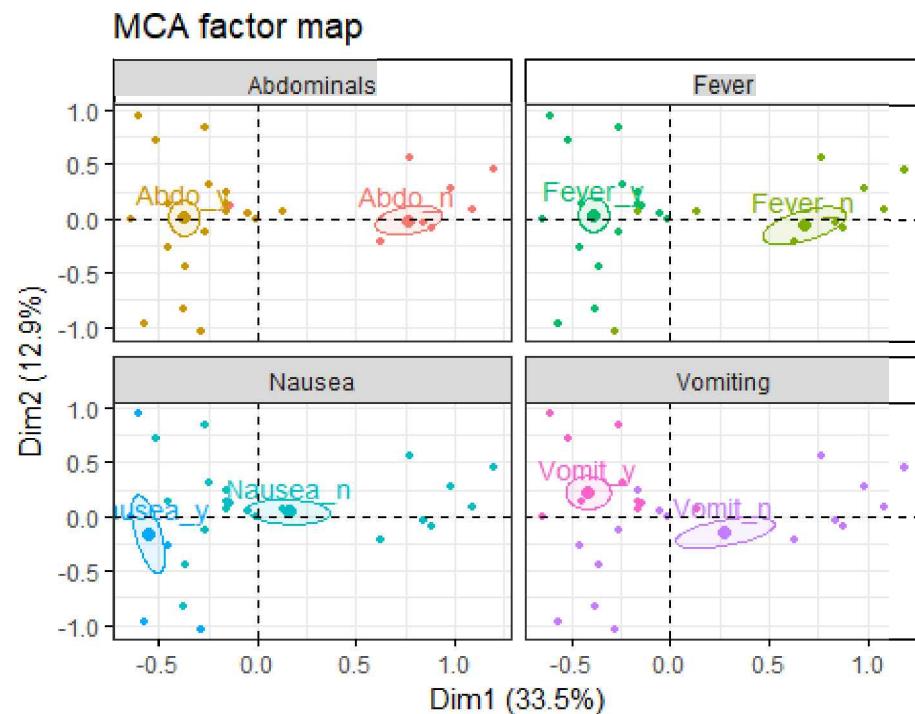


```
# Con elipses alrededor de puntos y habillage = variable de agrupación externa
fviz_mca_ind(res.mca, habillage = poison$Vomiting, addEllipses = TRUE)
```



La función `fviz_ellipses()` nos permite colorear individuos con varias variables, pudiendo especificar las variables por nombre o índice.

```
fviz_ellipses(res.mca, 1:4, geom = "point")
```



2.6.8. Descripción de la dimensión

Al igual que en el caso anterior, usamos la función dimdesc() para identificar las variables más correlacionadas con las dimensiones obtenidas.

```
# Descripción de la dimensión
res.desc <- dimdesc(res.mca, axes = c(1,2))

# Descripción de la dimensión 1
res.desc[[1]]
$quali
      R2    p.value
Abdominals 0.84512 4.0556e-23
Diarrhae   0.79947 3.9108e-20
Fever      0.78468 2.6006e-19
Mayo       0.38297 4.7562e-07
Vomiting   0.34420 2.5107e-06
Nausea     0.25620 8.0628e-05
Cheese     0.19442 7.5348e-04

$category
      Estimate    p.value
Abdo_n      0.56719 4.0556e-23
Diarrhea_n  0.53809 3.9108e-20
Fever_n     0.53309 2.6006e-19
Mayo_n      0.46450 4.7562e-07
Vomit_n    0.34669 2.5107e-06
Nausea_n   0.35479 8.0628e-05
Cheese_n   0.38300 7.5348e-04
Cheese_y   -0.38300 7.5348e-04
Nausea_y   -0.35479 8.0628e-05
Vomit_y    -0.34669 2.5107e-06
Mayo_y     -0.46450 4.7562e-07
Fever_y    -0.53309 2.6006e-19
Diarrhea_y -0.53809 3.9108e-20
Abdo_y     -0.56719 4.0556e-23
```

```
# Descripción de la dimensión 2
res.desc[[2]]
$quali
      R2      p.value
Courgette 0.44641 2.5002e-08
Potato    0.39575 2.6907e-07
Vomiting   0.25116 9.7280e-05
Icecream   0.14090 4.7439e-03

$category
      Estimate      p.value
Courg_n     0.41760 2.5002e-08
Potato_y    0.49775 2.6907e-07
Vomit_y     0.18381 9.7280e-05
Icecream_n   0.25972 4.7439e-03
Icecream_y  -0.25972 4.7439e-03
Vomit_n     -0.18381 9.7280e-05
Potato_n    -0.49775 2.6907e-07
Courg_y     -0.41760 2.5002e-08
```

2.6.9. Individuos y variables suplementarias

A continuación, vamos a realizar el análisis usando el data set "poison" declarando los tres últimos individuos como suplementarios y las correspondientes variables suplementarias, cualitativas y cuantitativas.

Los resultados para los individuos y variables suplementarias se obtienen:

```
res.mca <- MCA(poison, ind.sup=53:55, quanti.sup=1:2, quali.sup=3:4, graph=FALSE)

# Categorías de variables cualitativas suplementarias
res.mca$quali.sup
$coord
      Dim 1      Dim 2      Dim 3      Dim 4      Dim 5
Sick_n  1.418091  0.0020394  0.131991 -0.0160368 -0.083547
Sick_y  -0.630263 -0.0009064 -0.058663  0.0071275  0.037132
F       -0.031081  0.1123144  0.050331 -0.0559272 -0.068329
M       0.033568 -0.1212995 -0.054358  0.0604013  0.073796

$cos2
      Dim 1      Dim 2      Dim 3      Dim 4      Dim 5
Sick_n 0.8937703 1.8485e-06 0.0077430 0.0001143 0.0031022
Sick_y 0.8937703 1.8485e-06 0.0077430 0.0001143 0.0031022
F     0.0010433 1.3624e-02 0.0027359 0.0033781 0.0050424
M     0.0010433 1.3624e-02 0.0027359 0.0033781 0.0050424

$v.test
      Dim 1      Dim 2      Dim 3      Dim 4      Dim 5
Sick_n 6.75147  0.0097095  0.62840 -0.076351 -0.39776
Sick_y -6.75147 -0.0097095 -0.62840  0.076351  0.39776
F     -0.23067  0.8335514  0.37354 -0.415069 -0.50711
M     0.23067 -0.8335514 -0.37354  0.415069  0.50711

$\eta^2
      Dim 1      Dim 2      Dim 3      Dim 4      Dim 5
Sick 0.8937703 1.8485e-06 0.0077430 0.0001143 0.0031022
Sex  0.0010433 1.3624e-02 0.0027359 0.0033781 0.0050424

# Variables cuantitativas suplementarias
res.mca$quanti
$coord
      Dim 1      Dim 2      Dim 3      Dim 4      Dim 5
Age  0.0039349 -0.0074134 -0.264945  0.200155  0.029285
Time -0.8381585 -0.0833059 -0.087189 -0.084216 -0.023169
```

```
# Individuos supplementarios
```

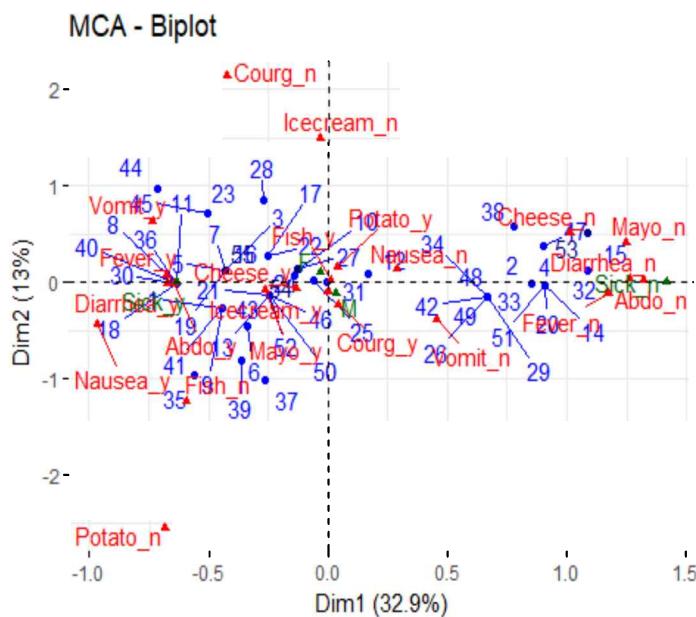
```
res.mca$ind.sup
$coord
  Dim 1   Dim 2   Dim 3   Dim 4   Dim 5
53  1.08357 0.51725 0.57941 0.53909 0.45537
54 -0.12495 0.14173 -0.17652 -0.15266 -0.27796
55 -0.43159 0.12705 -0.20716 -0.11868 -0.18918
```

```
$cos2
```

```
  Dim 1   Dim 2   Dim 3   Dim 4   Dim 5
53 0.363050 0.082728 0.103805 0.089862 0.064117
54 0.031577 0.040627 0.063025 0.047136 0.156266
55 0.502325 0.043527 0.115727 0.037983 0.096508
```

```
# Biplot de individuos y categorías de variables
```

```
fviz_mca_biplot(res.mca, repel = TRUE, ggtheme = theme_minimal())
```

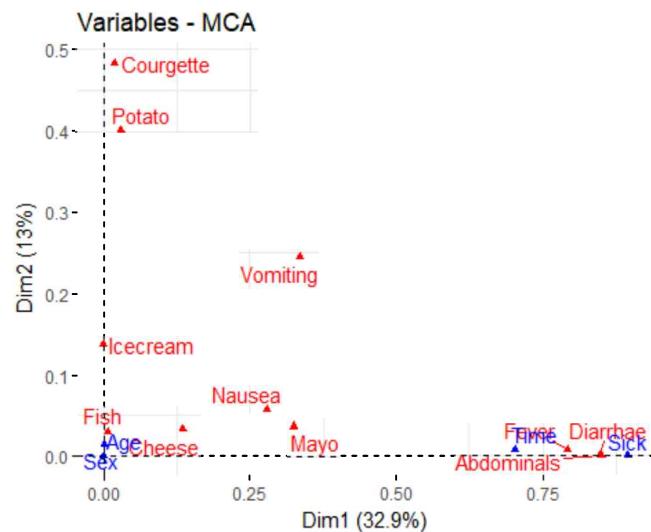


En el biplot de individuos y categorías de variables, los colores de los puntos corresponden a:

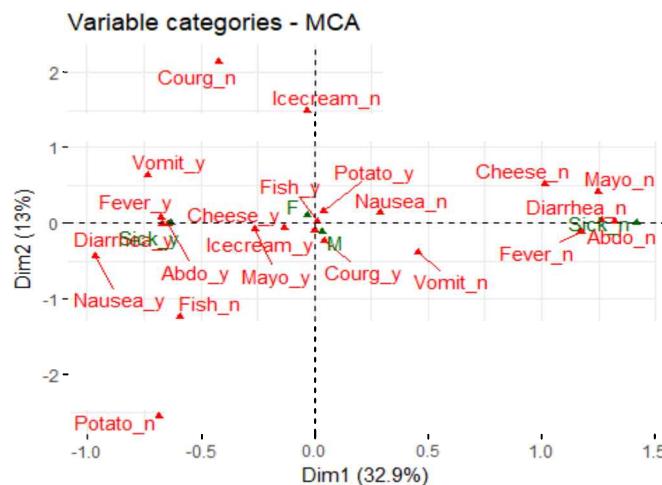
- Azul. Los individuos activos.
- Azul oscuro. Las individuos supplementarios.
- Rojo. Las categorías de las variables activas.
- Verde oscuro. Las categorías de las variables supplementarias.

La función fviz_mca_var() con el argumento choice = "mca.cor" resalta la correlación entre las variables (activas y supplementarias) y las dimensiones.

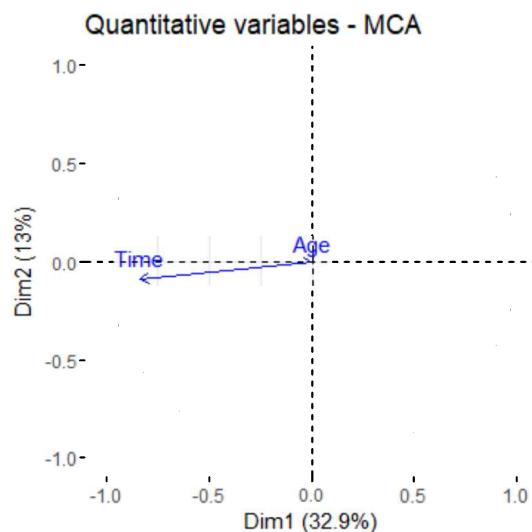
```
# Correlación entre las variables (activas y suplementarias) y las dimensiones
fviz_mca_var(res.mca, choice = "mca.cor", repel = TRUE)
```



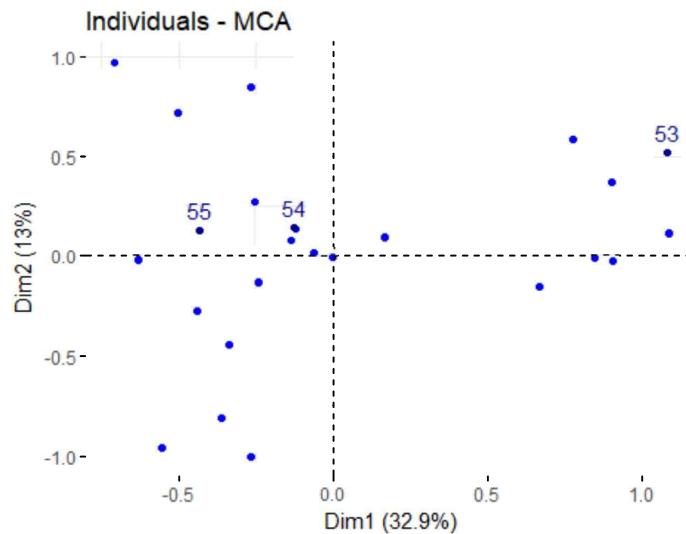
```
# Categorías de variables cualitativas (variables activas y suplementarias)
fviz_mca_var(res.mca, repel = TRUE, ggtheme= theme_minimal())
```



```
# Variables cuantitativas suplementarias
fviz_mca_var(res.mca, choice = "quanti.sup", ggtheme = theme_minimal())
```



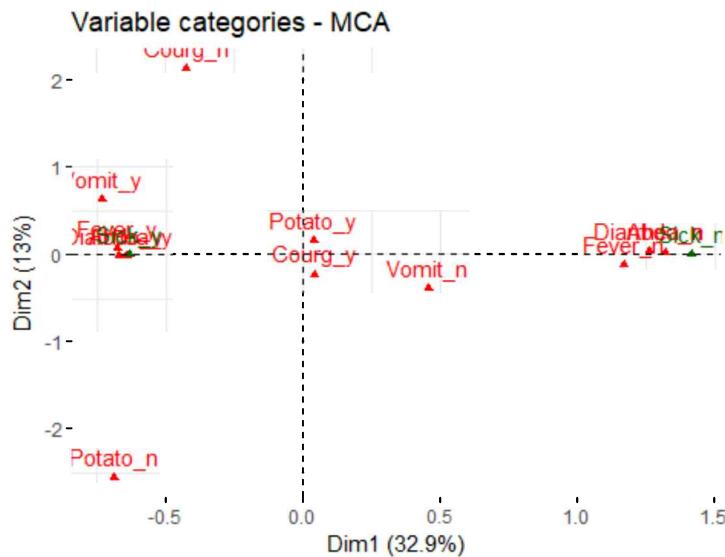
```
# Individuos suplementarios
fviz_mca_ind(res.mca,
              label = "ind.sup", # Mostrar solamente la etiqueta de ind.sup
              ggtheme = theme_minimal())
```



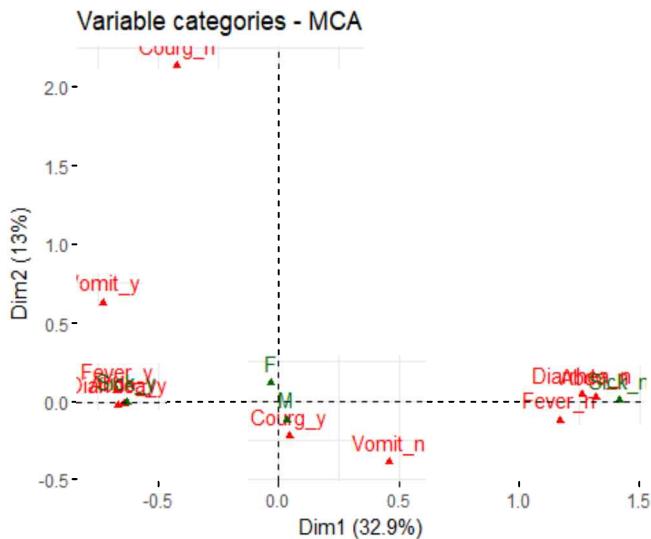
2.6.10. Filtrado de resultados

Es posible filtrar los resultados usando los argumentos **select.var** y **select.ind**. Como ejemplos:

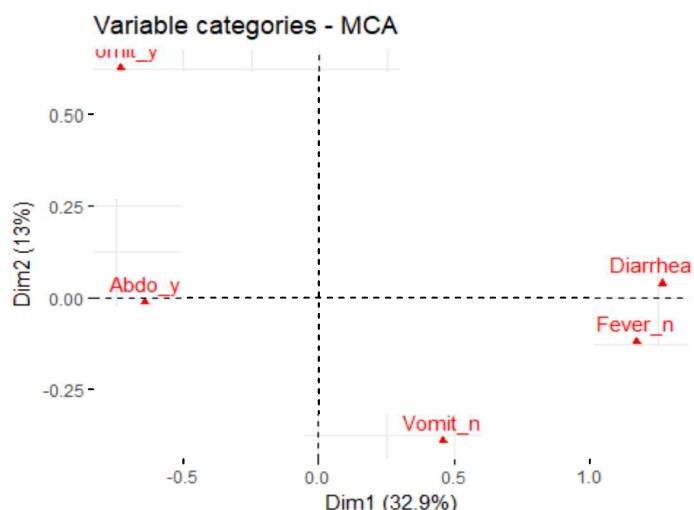
```
# Visualiza categorías de variables con cos2 >= 0.4
fviz_mca_var(res.mca, select.var = list(cos2 = 0.4))
```



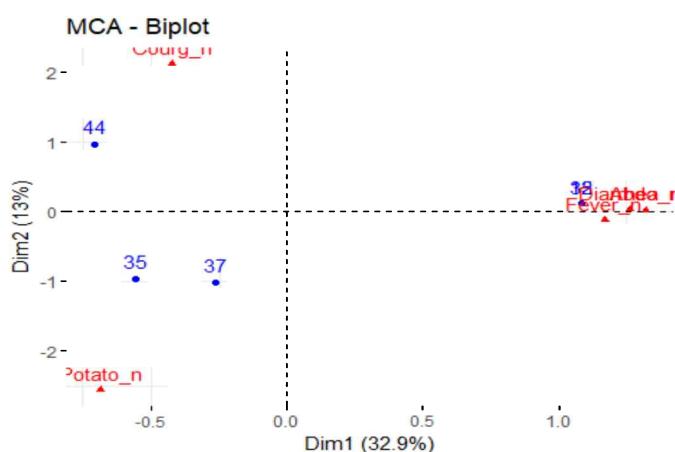
```
# Las 10 variables activas con el cos2 más alto
fviz_mca_var(res.mca, select.var= list(cos2 = 10))
```



```
# Selección por nombres
name <- list(name = c("Fever_n", "Abdo_y", "Diarrhea_n", "Fever_Y", "Vomit_y",
"Vomit_n"))
fviz_mca_var(res.mca, select.var = name)
```



```
# Los 5 individuos y categorías de variables que más contribuyen
fviz_mca_biplot(res.mca, select.ind = list(contrib = 5),
select.var = list(contrib = 5),
ggtheme = theme_minimal())
```



TEMA 5: MEDIDAS DE DISTANCIA Y AGRUPAMIENTO

ÍNDICE

1. Medidas de Distancia/Proximidad

- 1.1. Medidas de distancia o disimilaridad
- 1.2. Medidas de proximidad o similaridad
- 1.3. Distancia de Mahalanobis

2. Agrupamiento de la Información

- 2.1. Análisis Discriminante
- 2.2. Análisis Clúster
- 2.3. Escalamiento Multidimensional
- 2.4. Análisis de Correlación Canónica

OBJETIVOS	PALABRAS CLAVE
<ul style="list-style-type: none">· Manejar las distintas medidas de distancia/proximidad, aplicadas tanto a individuos como a variables, de cara a su utilización posterior para el agrupamiento (Análisis Clúster) y la reducción de la dimensionalidad (Escalaramiento Multidimensional).· Construir reglas de clasificación a través del Análisis Discriminante.· Reducir la dimensionalidad y analizar las relaciones existentes entre dos conjuntos de variables a través del Análisis de Correlación Canónica.	<ul style="list-style-type: none">· Similaridad / disimilaridad.· Proximidad / distancia.· Función discriminante de Fisher.· Tabla de clasificación.· Dendograma,· Agrupación jerárquica / no jerárquica.· Reducción de la dimensionalidad.· Correlación canónica.

1. MEDIDAS DE DISTANCIA/PROXIMIDAD

Tanto las técnicas de reducción de dimensiones como las de agrupamiento, están basadas en determinar la semejanza (proximidad, similaridad) o disparidad (distancia, disimilaridad) existente; entre las variables las primeras, entre los individuos/variables las segundas.

Lo primero a decidir será, pues, si optamos por centrar el análisis en medir disparidad o semejanza, lo cual dependerá en buena parte de los objetivos planteados en la investigación.

Otra cuestión a considerar a la hora de optar por una medida u otra es la escala de medida de la/s variable/s considerada/s (escala de intervalo, datos binarios, frecuencias).

Cuando se trabaja con $p > 2$ variables, la distancia euclídea (la medida de distancia más utilizada en escala de intervalo) presenta un inconveniente, que considera las p variables estocásticamente independientes. Este inconveniente puede resolverse introduciendo la distancia de Mahalanobis, la cual tiene en cuenta las correlaciones entre las variables, y por tanto la redundancia existente entre las mismas. En análisis discriminante, donde se trabaja con g poblaciones con distribuciones $N_p(\mu_i, \Sigma)$, la regla de la máxima verosimilitud para asignar a un sujeto a la población j correspondiente ($j = 1, \dots, g$) es equivalente a efectuar la asignación a la población más cercana considerando como medida la distancia de Mahalanobis, es decir, se elige la población cuya distancia sea mínima.

Dada la importancia de esta cuestión en el desarrollo de las técnicas estudiadas en este módulo, abordamos en primer lugar un pequeño apartado donde se describen las medidas de similaridad/disimilaridad más utilizadas.

1.1. MEDIDAS DE DISTANCIA O DISIMILARIDAD

Se llaman de distancia o de disimilaridad porque cuanto mayor es el valor de la medida, mayor será la diferencia entre los individuos. Los distintos tipos que existen dependen de la escala en la que éstas estén formuladas. A continuación se exponen las más empleadas en los distintos tipos de escalas.

1.1.1. Escala de intervalo

Cuando la variable está medida en una escala de intervalo las distancias más empleadas son las siguientes:

Distancia euclídea

Es la raíz cuadrada de la suma de las diferencias al cuadrado entre los dos elementos en la variable o variables consideradas.

$$EUCLID(X, Y) = \sqrt{\sum(X_i - Y_i)^2} \quad [1]$$

Distancia euclídea al cuadrado

$$SEUCLID(X, Y) = \sum(X_i - Y_i)^2 \quad [2]$$

Distancia métrica de Chebychev

Es la distancia máxima en valores absolutos entre los valores de los elementos.

$$CHEBYCHEV(X, Y) = \text{Max}_i |X_i - Y_i| \quad [3]$$

Distancia de Manhattan o Bloque+

La suma de las diferencias absolutas entre los valores de los elementos.

$$BLOCK(X, Y) = \sum |X_i - Y_i| \quad [4]$$

Distancia de Minkowski

La raíz p-ésima de la suma de las diferencias absolutas elevada a la potencia p-ésima entre los valores de los elementos.

$$MINKOWSKI(X, Y) = \sqrt[p]{\sum |X_i - Y_i|^p} \quad [5]$$

1.1.2. Frecuencias

Cuando se utilizan frecuencias agrupadas en tablas los tests más usados son los siguientes:

Medida de chi-cuadrado

Esta medida se basa en la prueba de chi-cuadrado de igualdad para dos conjuntos de frecuencias.

$$CHISQ(X, Y) = \sqrt{\sum \frac{(X_i - E(X_i))^2}{E(X_i)} + \frac{(Y_i - E(Y_i))^2}{E(Y_i)}} \quad [6]$$

donde:

- X_i, Y_i = Frecuencia observada
- $E(X_i), E(Y_i)$ = Frecuencia esperada

Medida de Phi-cuadrado

Esta medida es igual a la medida de chi-cuadrado normalizada por la raíz cuadrada de la frecuencia combinada.

$$PHI2(X, Y) = \frac{CHISQ(X, Y)}{\sqrt{n}} \quad [7]$$

1.1.3. Datos binarios

Este es el último caso que se da cuando se usan variables dicotómicas. Estas variables reflejan la presencia o ausencia de la característica medida. Generalmente, la presencia se codifica con valor 1 y la ausencia con valor 0. Se muestra a continuación la tabla de contingencia 2x2 con la notación utilizada:

TABLA 1. TABLA DE CONTINGENCIA 2x2

	$Y = 1$	$Y = 0$	Total
$X = 1$	a	b	$a + b$
$X = 0$	c	d	$c + d$
Total	$a + c$	$b + d$	n

Fuente: Elaboración propia

Los coeficientes más utilizados son los siguientes:

Distancia euclídea

Se calcula como la raíz cuadrada del número de casos discordantes.

$$BEUCLID(X, Y) = \sqrt{b + c} \quad [8]$$

Distancia euclídea al cuadrado

Se calcula como el número de casos discordantes.

$$BSEUCLID(X, Y) = b + c \quad [9]$$

Diferencia de tamaño

Se trata de un índice de asimetría. Oscila de 0 a 1.

$$SIZE(X, Y) = \frac{(b-c)^2}{(a+b+c+d)^2} \quad [10]$$

Diferencia de configuración

Medida que oscila de 0 a 1.

$$PATTERN(X, Y) = \frac{bc}{(a+b+c+d)^2} \quad [11]$$

Varianza

Medida que oscila entre 0 y 1.

$$VARIANCE(X, Y) = \frac{b+c}{4(a+b+c+d)} \quad [12]$$

Forma

Esta medida de distancia tiene un rango entre 0 y 1 y penaliza la asimetría de las no coincidencias.

$$BSHAPe(X, Y) = \frac{(a+b+c+d)(b+c)-(b-c)^2}{(a+b+c+d)^2} \quad [13]$$

Lance y Williams

Esta medida oscila entre 0 y 1. También se conoce como el coeficiente no métrico de Bray-Curtis.

$$BLW(X, Y) = \frac{b+c}{2a+b+c} \quad [14]$$

1.2. MEDIDAS DE PROXIMIDAD O SIMILARIDAD

En este caso la interpretación es al revés de las medidas de distancia. Es decir, un mayor valor indica una mayor cercanía entre las variables. Como en el caso anterior, dependen de la escala en la que estén formuladas las variables.

1.2.1. Escala de intervalo

Coeficiente de correlación de Pearson

Su valor varía de -1 a 1. La mayor cercanía en valor absoluto al uno indica una mayor relación o proximidad entre los individuos.

Coseno de vectores

Su fórmula es la siguiente:

$$\text{COS}(X, Y) = \frac{\sum(X_i Y_i)}{\sqrt{\sum X_i^2 \sum Y_i^2}} \quad [15]$$

1.2.2. Datos binarios

Coeficiente de Russell y Rao

No tiene en cuenta en el numerador las ocasiones en que en ambos casos no aparece una de las variables, pero sí en el denominador:

$$\text{RR}(X, Y) = \frac{a}{a+b+c+d} \quad [16]$$

Coeficiente de concordancia simple

Este coeficiente considera como semejanzas entre los dos casos tanto la presencia en ambos de una variable (a) como la ausencia de ésta en ambos (d).

$$\text{SM}(X, Y) = \frac{a+d}{a+b+c+d} \quad [17]$$

Va a tomar valores entre cero y uno. Si $\text{SM}(X, Y)$ vale cero será porque a y d valen cero, por lo que hay una ausencia total de similitud entre los dos casos. Si vale uno entonces c y b valdrán cero, es decir, ausencia de disimilitud, o lo que es lo mismo, similitud plena.

Coeficiente de Jaccard

$$\text{JACCARD}(X, Y) = \frac{a}{a+b+c} \quad [18]$$

Este coeficiente no tiene en cuenta las ocasiones en que en ambos casos no aparece una de las variables (d), no considera esa ausencia en ambos casos como un mayor parecido entre los casos. Se ofrece una ponderación igual a las coincidencias y a las no coincidencias. Se conoce también como razón de similaridad.

Coeficiente de Dice

Al igual que el de Jaccard, no tiene en cuenta las ocasiones en que en ambos casos no aparece una de las variables. Además, da doble importancia al hecho de que una variable se dé en ambos casos.

$$\text{DICE}(X, Y) = \frac{2a}{2a+b+c} \quad [19]$$

Coeficiente de Rogers-Tanimoto

Da doble importancia a la no coincidencia, es decir, a que una variable aparezca en un caso pero no en el otro.

$$RT(X, Y) = \frac{a}{a+d+2(b+c)} \quad [20]$$

Sokal y Sneath 1

Se trata de un índice en el que se ofrece una ponderación doble a las coincidencias.

$$SS1(X, Y) = \frac{2(a+d)}{2(a+d)+b+c} \quad [21]$$

Sokal y Sneath 2

Se trata de un índice en el que se ofrece una ponderación doble a las no coincidencias y no se toman en cuenta las ausencias conjuntas.

$$SS2(X, Y) = \frac{a}{a+2(b+c)} \quad [22]$$

Sokal and Sneath 3

Esta es la razón de coincidencias y no coincidencias. Este índice tiene un límite inferior de 0 y carece de límite superior. No está definido teóricamente cuando no existen no coincidencias.

$$SS3(X, Y) = \frac{a+d}{b+c} \quad [23]$$

Kulczynski 1

Se trata de la razón de presencias conjuntas sobre todas las no coincidencias. Este índice tiene un límite inferior de 0 y carece de límite superior. No está definido teóricamente cuando no existen no coincidencias.

$$K1(X, Y) = \frac{a}{b+c} \quad [24]$$

Kulczynski 2

Este índice está basado en la probabilidad condicional de que la característica esté presente en un elemento, siempre que esté presente en el otro. Para calcular este valor se promedian los distintos valores para cada elemento que actúa como predictor del otro. Toma valores entre 0 y 1.

$$K2(X, Y) = \frac{a/(a+b)+a/(a+c)}{2} \quad [25]$$

Sokal and Sneath 4

Este índice se basa en la probabilidad condicional de que la característica de un elemento coincida con el valor del otro. Para calcular este valor se promedian los distintos valores para cada elemento que actúa como predictor del otro.

$$SS4(X, Y) = \frac{a/(a+b)+a/(a+c)+d/(b+d)+d/(c+d)}{4} \quad [26]$$

Hamann

Este índice es el número de coincidencias menos el número de no coincidencias, dividido por el número total de elementos. Oscila entre -1 y 1.

$$HAMANN(X, Y) = \frac{(a+d)-(b+c)}{a+b+c+d} \quad [27]$$

Lambda de Goodman y Kruskal

Corresponde a la reducción proporcional del error (RPE o PRE) utilizando un elemento para pronosticar el otro (pronosticando en ambas direcciones). Los valores oscilan entre 0 y 1.

$$\text{LAMBDA}(X, Y) = \frac{t_1 - t_2}{2(a+b+c+d)} \quad [28]$$

donde:

$$t_1 = \text{Máx}(a, b) + \text{Máx}(c, d) + \text{Máx}(a, c) + \text{Máx}(b, d) \quad [29]$$

$$t_2 = \text{Máx}(a + c, b + d) + \text{Máx}(a + b, c + d) \quad [30]$$

D de Anderberg

Similar a lambda, este índice corresponde a la reducción de error real utilizando un elemento para predecir el otro (predice en ambas direcciones). Los valores oscilan entre 0 y 1.

$$D(X, Y) = \frac{t_1 + t_2}{2(a+b+c+d)} \quad [31]$$

Y de Yule

Este índice es una función de la razón cruzada para una tabla 2x2, y es independiente de los totales marginales. Varía entre -1 y 1. También se denomina coeficiente de coligación.

$$Y(X, Y) = \frac{\sqrt{ad} - \sqrt{bc}}{\sqrt{ad} + \sqrt{bc}} \quad [32]$$

Q de Yule

Este índice es un caso especial de gamma de Goodman y Krustal. Es una función de la razón cruzada y es independiente de los totales marginales. Varía entre -1 y 1.

$$Q(X, Y) = \frac{ad - bc}{ad + bc} \quad [33]$$

Ochiai

Este índice es la forma binaria de la medida de similaridad del coseno. Varía entre 0 y 1.

$$OCHIAI(X, Y) = \sqrt{\left(\frac{a}{a+b}\right)\left(\frac{a}{a+c}\right)} \quad [34]$$

Sokal y Sneath 5

Este índice es la media geométrica al cuadrado de las probabilidades condicionales de coincidencias positivas y negativas. Es independiente de la codificación de elementos. Varía entre 0 y 1.

$$SS5(X, Y) = \frac{ad}{\sqrt{(a+b)(a+c)(b+d)(c+d)}} \quad [35]$$

Correlación Phi de 4 puntos

Este índice es un análogo binario del coeficiente de correlación de Pearson. Varía entre -1 y 1.

$$SS5(X, Y) = \frac{ad-bc}{\sqrt{(a+b)(a+c)(b+d)(c+d)}} \quad [36]$$

Dispersión

Este índice tiene un rango de -1 a 1.

$$DISPER(X, Y) = \frac{ad-bc}{(a+b+c+d)^2} \quad [37]$$

1.3. DISTANCIA DE MAHALANOBIS

La distancia euclídea presenta algunos inconvenientes:

- a) No está acotada.
- b) No es invariante por cambios de escala.
- c) Si tenemos un conjunto de p variables, X_1, X_2, \dots, X_p con las cuales evaluar distancias entre individuos, considera las p variables estocásticamente independientes.

La invarianza por cambios de escala se resuelve dividiendo cada término (x_{ik}, x_{jk}) por la desviación típica de la variable k, lo que nos lleva a la distancia de K. Pearson. El inconveniente c) puede resolverse introduciendo la distancia de Mahalanobis.

1.3.1. Distancia euclídea normalizada

Dada una matriz de datos $X_{n \times p}$, con n individuos y p variables, la distancia euclídea normalizada $K(i, j)$ es la raíz cuadrada de:

$$K^2(i, j) = \sum_{k=1}^p \frac{(x_{ik} - x_{jk})^2}{\sigma_k^2} \quad [38]$$

La distancia $K(i, j)$ es invariante por cambios de escala y es una distancia entre individuos relacionada con el coeficiente de semejanza racial introducido por K. Pearson (1926), que ha sido utilizado en antropología para diferenciar cráneos. Dadas dos poblaciones representadas por (μ_1, Σ) y (μ_2, Σ) , donde (μ_1, μ_2) son los vectores de medias y Σ es la matriz de covarianzas (común) en relación a p variables aleatorias, el coeficiente de semejanza racial, también llamado distancia de K. Pearson, es proporcional a:

$$K^2 = (\mu_1 - \mu_2)' [diag(\Sigma)]^{-1} (\mu_1 - \mu_2) \quad [39]$$

K es también invariante por cambios de escala y puede considerarse un precedente de la distancia de Mahalanobis. Ambas distancias han sido comparadas por diversos autores.

1.3.2. Definición y propiedades de la distancia de Mahalanobis

Supongamos que una población Ω está caracterizada por p variables aleatorias, siendo $\mu = (\mu_1, \dots, \mu_p)'$ el vector de medias Σ la matriz de covarianzas no singular. La distancia de Mahalanobis $M(i, j)$ entre dos individuos i, j, representados por los vectores x_i, x_j , se define como:

$$M^2(i, j) = (x_i - x_j)' \Sigma^{-1} (x_i - x_j) \quad [40]$$

Análogamente, la distancia entre un individuo i y la población Ω es:

$$M^2(i,j) = (x_i - \mu)' \Sigma^{-1} (x_i - \mu) \quad [41]$$

La distancia entre dos poblaciones Ω_1, Ω_2 es

$$M^2(i,j) = (\mu_1 - \mu_2)' \Sigma^{-1} (\mu_1 - \mu_2) \quad [42]$$

La distancia fue introducida por Mahalanobis (1936), alegando criterios heurísticos. Sus propiedades son las siguientes:

1. $M(i,j) > 0$ y $M(i,j) = 0$ si y solo si $x_i = x_j$.
2. $M(i,j) = M(j,i)$.
3. $M(i,j) < M(i,k) + M(j,k)$.
4. $M(i,j)$ es invariante por transformaciones lineales no singulares de las variables. En particular, es invariante por cambios de escala.
5. Introduciendo el cambio de variable $y = \Sigma^{-\frac{1}{2}}x$, es fácil ver que la distancia de Mahalanobis es euclídea.
6. Es una distancia normalizada, que puede expresarse en unidades de desviación típica. Además, tiene en cuenta las correlaciones entre las variables, es decir, la redundancia entre las variables.
7. Indiquemos por M_p la distancia basada en p variables y por M_{p+q} la distancia basada en $p+q$ variables, conteniendo estas $p+q$ a las p primeras. Entonces $M_p \leq M_{p+q}$.
8. Sean M_p, M_q las distancias tomando las variables $X = (X_1, \dots, X_p)$ $Y = (Y_1, \dots, Y_q)$. Supongamos que las variables X están incorrelacionadas con las variables Y . Entonces $M_{p+q}^2 = M_p^2 + M_q^2$.

Es de esperar, por otra parte, que en las aplicaciones la distancia sea estable

$$\lim_{p \rightarrow \infty} M_p = \alpha < \infty \quad [43]$$

es decir, si el número de variables p es grande, la distancia de Mahalanobis no aumentará o al menos tenderá a un valor finito α , debido a que las variables añadidas serán redundantes respecto a las p anteriores.

1.3.3. Distancias singulares

Supongamos $rango(\Sigma) = r < p$ y que $\mu_1 - \mu_2$ es combinación lineal de las columnas de Σ . Se define la distancia de Mahalanobis singular entre las poblaciones de vectores de medias μ_1, μ_2 y matriz de covarianzas Σ como:

$$M^2(i,j) = (\mu_1 - \mu_2)' \Sigma^- (\mu_1 - \mu_2) \quad [44]$$

donde Σ^- es una g -inversa de Σ . La distancia singular tiene aplicaciones a la genética, al análisis factorial y a la comparación de curvas de crecimiento.

2. AGRUPAMIENTO DE LA INFORMACIÓN

2.1. ANÁLISIS DISCRIMINANTE

El Análisis Discriminante (AD), introducido por Fisher (1936), es una técnica que se utiliza para predecir la pertenencia a un grupo (variable dependiente) a partir de un conjunto de predictores (variables independientes). El objetivo del AD es entender las diferencias de los grupos y predecir la verosimilitud de que una persona o un objeto pertenezcan a una clase o grupo basándose en los valores que toma en los predictores. Ejemplos de análisis discriminante son distinguir entre innovadores y no innovadores de acuerdo a sus perfiles demográficos y sociales o el riesgo de impago de un préstamo a través de predictores económicos y sociodemográficos.

Existen dos enfoques en la clasificación discriminante:

- El basado en la obtención de funciones discriminantes de cálculo similar a las ecuaciones de regresión lineal múltiple.
- Empleando técnicas de correlación canónica y de componentes principales, denominado análisis discriminante canónico.

El primer enfoque es el más común y es el que abordamos a continuación. Su fundamento matemático está en conseguir, a partir de las variables explicativas, unas funciones lineales de éstas con capacidad para clasificar a otros individuos, donde la función de mayor valor define el grupo a que pertenece de forma más probable.

El AD solo admite variables cuantitativas como regresores, por lo que si alguna de las variables independientes es categórica, hay que utilizar otros métodos alternativos de clasificación.

2.1.1. Clasificación con dos grupos

2.1.1.1. Planteamiento del problema

El análisis discriminante es conceptualmente muy similar al análisis de varianza multivariante de un factor. El AD trata de establecer una relación entre una variable dependiente no métrica (dicotómica o multidicotómica) y un conjunto de p variables independientes métricas:

El propósito del AD consiste en aprovechar la información contenida en las variables independientes para crear una función Z combinación lineal de las p variables explicativas, capaz de diferenciar lo más posible a los 2 grupos. La combinación lineal para el análisis discriminante, *función discriminante de Fisher*, se formula:

$$z_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \dots + \beta_p x_{pi} \quad [45]$$

donde:

z_i es la puntuación Z discriminante para el objeto i .

β_0 es el término constante.

β_j es la ponderación discriminante para la variable independiente j .

x_{ji} es la variable independiente j para el objeto i .

Una vez hallada la función discriminante, el resultado es una única puntuación z_i discriminante compuesta para cada individuo en el análisis. Promediando las puntuaciones discriminantes para todos los individuos dentro de un grupo particular, obtenemos la media del grupo. Esta media es conocida como *centroide*.

Sustituyendo en la función discriminante el valor de las medias del grupo 1 en las variables X_1 y X_2 , obtenemos el centroide del grupo 1:

$$\bar{z}_I = \beta_0 + \beta_1 \bar{x}_{1,I} + \beta_2 \bar{x}_{2,I} + \cdots + \beta_p \bar{x}_{p,I} \quad [46]$$

De igual modo, sustituyendo las medias del grupo 2, obtenemos el centroide del grupo 2:

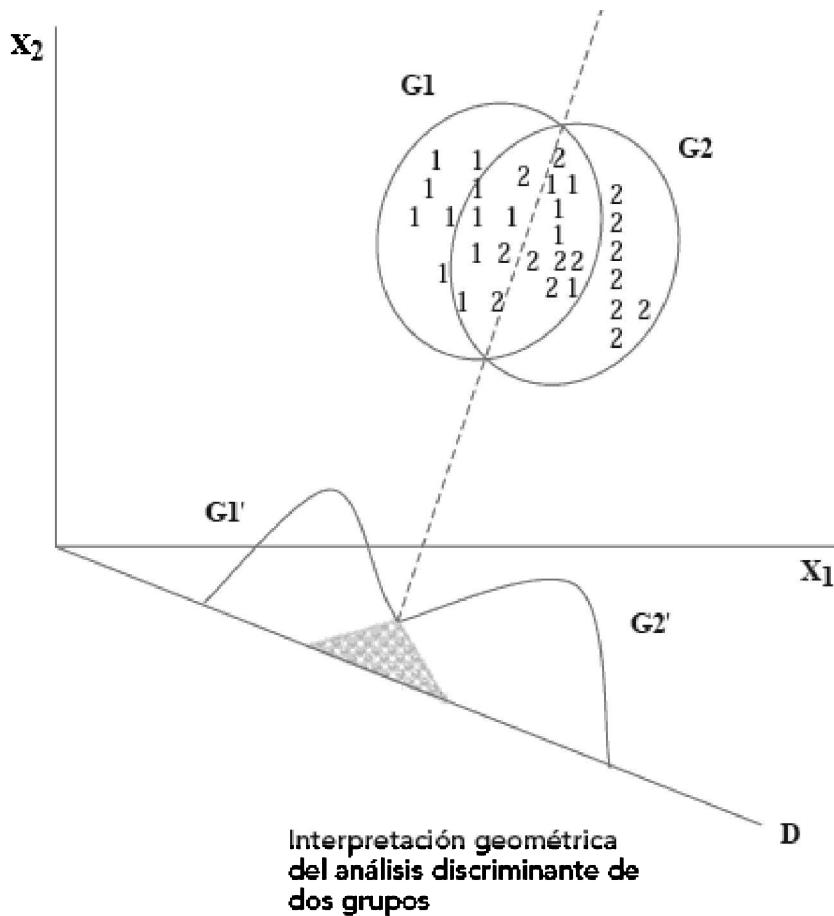
$$\bar{z}_{II} = \beta_0 + \beta_1 \bar{x}_{1,II} + \beta_2 \bar{x}_{2,II} + \cdots + \beta_p \bar{x}_{p,II} \quad [47]$$

La función Z debe ser tal que la distancia entre los dos centroides sea máxima, consiguiendo de esta forma que los grupos estén lo más distantes posible. Podemos expresar esta distancia de la siguiente manera:

$$h = \bar{z}_I - \bar{z}_{II} \quad [48]$$

Es importante señalar que los grupos deben diferenciarse de antemano en las variables independientes. El análisis busca diferenciar los dos grupos al máximo combinando las variables independientes pero si los grupos no difieren en las variables independientes, no podrá encontrar una dimensión en la que los grupos difieran (Figura 1).

FIGURA 1. ANÁLISIS DISCRIMINANTE



Fuente: Elaboración propia

Dicho de otro modo, si el solapamiento entre los casos de ambos grupos es excesivo, los centroides se encontrarán en la misma o parecida ubicación en el espacio p-dimensional y en esas condiciones, no será posible encontrar una función discriminante útil para la clasificación. Si los centroides están muy próximos, las medias de los grupos en la función discriminante serán tan parecidas que no será posible distinguir a los sujetos de uno y otro grupo.

La mayor utilidad de una función discriminante radica en su capacidad para clasificar nuevos casos. Ahora bien, la clasificación de casos es algo muy distinto de la estimación de la función discriminante. De hecho, una función perfectamente estimada puede tener una pobre capacidad clasificatoria.

Una vez obtenida la función discriminante podemos utilizarla, en primer lugar, para efectuar una clasificación de los mismos casos utilizados para obtener la función: esto permitirá comprobar el grado de eficacia de la función desde el punto de vista de la clasificación. Si los resultados son satisfactorios, la función discriminante podrá utilizarse, en segundo lugar, para clasificar futuros casos de los que, conociendo su puntuación en las variables independientes, se desconozca el grupo al que pertenecen.

Una manera de clasificar los casos consiste en calcular la distancia existente entre los centroides de ambos grupos y situar un punto de corte $z_0 = \frac{\bar{z}_1 + \bar{z}_2}{2}$ equidistante de ambos centroides. A partir de ese momento, los casos cuyas puntuaciones discriminantes sean mayores que el *punto de corte discriminante* z_0 , serán asignados al grupo superior, y los casos cuyas puntuaciones discriminantes sean menores que el punto de corte z_0 , serán asignados al grupo inferior.

La regla de clasificación descrita solo permite distinguir entre dos grupos, con lo que es difícilmente aplicable al caso de más de dos grupos e incluso a dos grupos con distinto tamaño. Con tamaños desiguales es preferible utilizar una regla de clasificación que desplace el punto de corte hacia el centroide del grupo de menor tamaño buscando igualar los errores de clasificación. Para calcular este punto de corte se utiliza una distancia ponderada:

$$z_0 = \frac{n_I \bar{z}_I + n_{II} \bar{z}_{II}}{n_I + n_{II}} \quad [49]$$

2.1.1.2. Cálculo de la Función Discriminante

Como se ha explicado, la función discriminante de Fisher Z se obtiene como función lineal de las p variables explicativas X, es decir:

$$z_i = \beta_1 x_{1i} + \cdots + \beta_p x_{pi} \text{ con } i = 1, \dots, n \quad [50]$$

donde z_i será la **puntuación discriminante**. En notación matricial, el modelo queda expresado como:

$$Z = X\beta \quad [51]$$

Donde las variables explicativas se expresarán en desviaciones respecto a la media y, de esta manera, las puntuaciones z_i también lo estarán.

La variabilidad de la función discriminante (suma de cuadrados de las variables discriminantes en desviaciones respecto a su media) se expresa entonces como:

$$Z'Z = \beta'X'X\beta \quad [52]$$

La matriz $X'X$ es una matriz simétrica, y puede considerarse como la Suma de Cuadrados Total (*SCT*) de las variables (explicativas) de la matriz X. Según la teoría del Análisis Multivariante de la Varianza, $X'X$ (*VT*) se puede descomponer en la suma de la matriz entre-grupos F y la matriz intra-grupos W (o residual). Por tanto, la variabilidad quedaría expresada como:

$$Z'Z = \beta'X'X\beta = \beta'F\beta + \beta'W\beta \quad [53]$$

donde F y W son las matrices correspondientes a las sumas de cuadrados entre-grupos e intra-grupos respectivamente, que se calculan con los datos muestrales, mientras que los coeficientes β_1, \dots, β_p están por determinar.

Fisher (1936) obtuvo los β_j maximizando la razón de la variabilidad entre-grupos respecto de la variación intra-grupos. Con este criterio se trata de determinar el eje discriminante de forma que las distribuciones proyectadas sobre el mismo estén lo más separadas posible entre sí (mayor variabilidad entre-grupos) y, al mismo tiempo, que cada una de las distribuciones esté lo menos dispersa (menor variabilidad intra-grupos).

Analíticamente, esto se expresa:

$$\text{Max}\lambda = \frac{\beta'F\beta}{\beta'W\beta} \quad [54]$$

La solución a este problema se obtiene derivando λ respecto de β e igualando a cero, es decir:

$$\frac{\partial\lambda}{\partial\beta} = \frac{2F\beta(\beta'W\beta) - 2W\beta(\beta'F\beta)}{(\beta'W\beta)^2} = 0 \Rightarrow F\beta(\beta'W\beta) - W\beta(\beta'F\beta) = 0 \quad [55]$$

Lo cual implica que:

$$\frac{F\beta}{W\beta} = \frac{\beta'F\beta}{\beta'W\beta} = \lambda \Rightarrow F\beta = W\beta\lambda \Rightarrow W^{-1}F\beta = \lambda\beta \quad [56]$$

En consecuencia, la ecuación para obtener el primer eje discriminante se traduce en la obtención de un vector propio β asociado a la matriz no simétrica $W^{-1}F$.

Dado que λ es la ratio a maximizar, cuando se calcule medirá el poder discriminante del primer eje discriminante. Como se está realizando un análisis discriminante con dos grupos, no se necesitan más ejes discriminantes.

Los coeficientes ($\hat{\beta}_1, \dots, \hat{\beta}_p$) normalizados correspondientes a las coordenadas del vector propio unitario asociado al mayor valor propio de la matriz $W^{-1}F$ obtenidos en el proceso de maximización, pueden contemplarse como un conjunto de cosenos directores que definen la situación del eje discriminante.

En general, al aplicar el análisis discriminante se le resta el valor de z_0 a la función discriminante, es decir, que vendrá dada por:

$$d_i = z_i - z_0 = \hat{\beta}_1 x_{1i} + \hat{\beta}_2 x_{2i} + \dots + \hat{\beta}_p x_{pi} - z_0 \quad [57]$$

Luego si $d_i < 0$ el individuo i se clasificará en el primer grupo y si $d_i > 0$ en el segundo.

2.1.2. Clasificación con más de dos grupos

En el caso general de un análisis discriminante con G grupos ($G > 2$), el número máximo de ejes discriminantes que se pueden obtener viene dado por $\text{Min}(G - 1, p)$. Por tanto, pueden obtenerse hasta $G - 1$ ejes discriminantes si el número de variables explicativas p es mayor que $G - 1$, lo cual suele ser cierto en la gran mayoría de las aplicaciones prácticas.

El resto de los ejes discriminantes vendrán dados por los vectores propios asociados a los valores propios de la matriz $W^{-1}F$ ordenados de mayor a menor. Así, el segundo eje discriminante tendrá menos poder discriminante que el primero, pero más que cualquiera de los restantes.

Como la matriz $W^{-1}F$ no es simétrica, los ejes discriminantes no serán en general ortogonales.

La generalización del procedimiento se realiza considerando la construcción de funciones discriminantes para cada grupo de la siguiente manera:

$$\left[\begin{array}{l} f_{1,i} = \alpha_{1,1}x_{1i} + \alpha_{1,2}x_{2i} + \cdots + \alpha_{1,p}x_{pi} - \bar{z}_1 \\ f_{2,i} = \alpha_{2,1}x_{1i} + \alpha_{2,2}x_{2i} + \cdots + \alpha_{2,p}x_{pi} - \bar{z}_2 \\ \vdots \\ f_{G,i} = \alpha_{G,1}x_{1i} + \alpha_{G,2}x_{2i} + \cdots + \alpha_{G,p}x_{pi} - \bar{z}_G \end{array} \right] \quad [58]$$

Y se clasifica a los individuos en el grupo para el que la función F_g sea mayor.

Si consideramos el caso de dos grupos y restamos las funciones:

$$f_{2,i} - f_{1,i} = (\alpha_{2,1} - \alpha_{1,1})x_{1i} + (\alpha_{2,2} - \alpha_{1,2})x_{2i} + \cdots + (\alpha_{2,p} - \alpha_{1,p})x_{pi} - (\bar{z}_2 - \bar{z}_1) \quad [59]$$

es fácil ver que sería equivalente a la situación anterior, ya que:

$$\begin{aligned} \beta_1 &= \alpha_{2,1} - \alpha_{1,1} \\ \beta_2 &= \alpha_{2,2} - \alpha_{1,2} \\ &\vdots \\ \beta_p &= \alpha_{2,p} - \alpha_{1,p} \end{aligned} \quad [60]$$

y

$$z_0 = \bar{z}_2 - \bar{z}_1 \quad [61]$$

Y por tanto:

$$f_{2,i} - f_{1,i} = z_i - z_0 = d_i \quad [62]$$

2.1.3. Ejemplos con el software R

La función R que realiza el Análisis Discriminante Lineal es “lda”. Para los 5 primeros datos, se dan los resultados de la clasificación (class) y las probabilidades posteriores de pertenecer a la clase cero (posterior.0) o de pertenecer a la clase 1 (posterior.1). La probabilidad posterior es la probabilidad condicional que es asignada después de que la evidencia es tomada en cuenta.

Con la base de datos de R, OJ de la librería ISLR, que contiene datos acerca de consumidores de zumos a partir de diferencias de precios, descuentos y grado de fidelización con una de las dos marcas, realizaremos un ejercicio de clasificación con AD, y evaluaremos los resultados con una métrica de porcentaje de aciertos.

```

library(MASS)
library(ISLR)
data("OJ")
str(OJ)
'data.frame':   1070 obs. of  18 variables:
 $ Purchase    : Factor w/ 2 levels "CH","MM": 1 1 1 2 1 1 1 1 1 ...
 $ WeekofPurchase: num  237 239 245 227 228 230 232 234 235 238 ...
 $ StoreID     : num  1 1 1 1 7 7 7 7 7 ...
 $ PriceCH     : num  1.75 1.75 1.86 1.69 1.69 1.69 1.69 1.75 1.75 1.75 ...
 $ PriceMM     : num  1.99 1.99 2.09 1.69 1.99 1.99 1.99 1.99 1.99 1.99 ...
 $ DiscCH      : num  0 0 0.17 0 0 0 0 0 0 ...
 $ DiscMM      : num  0 0.3 0 0 0 0 0.4 0.4 0.4 0.4 ...
 $ SpecialCH   : num  0 0 0 0 0 1 1 0 0 ...
 $ SpecialMM   : num  0 1 0 0 0 1 1 0 0 ...
 $ LoyalCH     : num  0.5 0.6 0.68 0.4 0.957 ...
 $ SalePriceMM : num  1.99 1.69 2.09 1.69 1.69 1.99 1.59 1.59 1.59 1.59 ...
 $ SalePriceCH : num  1.75 1.75 1.69 1.69 1.69 1.69 1.69 1.75 1.75 1.75 ...
 $ PriceDiff    : num  0.24 -0.06 0.4 0 0 0.3 -0.1 -0.16 -0.16 -0.16 ...
 $ Store7      : Factor w/ 2 levels "No","Yes": 1 1 1 1 2 2 2 2 2 2 ...
 $ PctDiscMM   : num  0 0.151 0 0 0 ...
 $ PctDiscCH   : num  0 0 0.0914 0 0 ...
 $ ListPriceDiff: num  0.24 0.24 0.23 0 0 0.3 0.3 0.24 0.24 0.24 ...
 $ STORE        : num  1 1 1 1 0 0 0 0 0 0 ...

```

Análisis Discriminante Lineal

```

compra=lda(Purchase~PriceDiff+PctDiscMM+PctDiscCH+LoyalCH,data=OJ)
compra
Call:
  lda(Purchase ~ PriceDiff + PctDiscMM + PctDiscCH + LoyalCH, data = OJ)

Prior probabilities of groups:
  CH      MM 
0.6102804 0.3897196

Group means:
  PriceDiff PctDiscMM PctDiscCH  LoyalCH
CH 0.20614089 0.04576862 0.03637068 0.7232937
MM 0.05306954 0.08048541 0.01313131 0.3191278

Coefficients of linear discriminants:
  LD1
PriceDiff -2.2109035
PctDiscMM -1.8823914
PctDiscCH  0.7482173
LoyalCH   -4.0076435

# Predicción
probs=predict(compra, newdata=OJ, type="prob")
data.frame(probs)[1:5,]
  class posterior.CH posterior.MM      LD1
1   CH     0.6361482    0.36385175  0.1480678
2   CH     0.6453310    0.35466903  0.1267965
3   CH     0.9203239    0.07967607 -0.8586670
4   MM     0.2336052    0.76639479  1.0794490
5   CH     0.9523435    0.04765653 -1.1509449

table(probs$class,OJ$Purchase)
  CH  MM
CH 567 101
MM 86 316

mean(probs$class==OJ$Purchase) # Porcentaje de bien clasificados
[1] 0.8252336

```

Para realizar este ejercicio vamos a realizar una minería de datos, dividiendo la base de datos en una muestra de entrenamiento y otra de test, la muestra de entrenamiento incluirá el 70% de las observaciones de la base de datos.

```

library(MASS)
data(iris)
attach(iris)

# División de la muestra en entrenamiento y validación
train=sample(seq(length(iris$Species)),length(iris$Species)*0.70,replace=FALSE)

# Análisis Discriminante Lineal
lda.tr=lda(Species[train]~.,data=iris[train,])

# Predicción
probs=predict(lda.tr,newdata=iris[-train,],type="prob")
data.frame(probs)[1:5,]
  class posterior.setosa posterior.versicolor posterior.virginica
2  versicolor      0.2895506      0.3635590      0.3468903
6  virginica       0.1660075      0.3392183      0.4947742
9  virginica       0.2744970      0.3253211      0.4001818
22 virginica       0.1898259      0.3644994      0.4456747
23 virginica       0.1802162      0.3515042      0.4682796
    x.LD1           x.LD2
2 -0.0233803  0.91752925
6 -1.4677214  0.47205794
9 -0.3056882 -0.02732207
22 -1.1011722  0.92950057
23 -1.2550874  0.67870284

table(probs$class,iris$Species[-train])

      setosa versicolor virginica
setosa      0        10       12
versicolor   4        0        1
virginica   11        6        1

mean(probs$class==iris$Species[-train]) # Porcentaje de bien clasificados
[1] 0.02222222

```

2.2. ANÁLISIS CLÚSTER

2.2.1. Introducción

El análisis clúster (AC) es un conjunto de técnicas multivariantes cuyo principal propósito es agrupar objetos basándose en las características que poseen. El AC clasifica los objetos en clases o conglomerados de tal forma que cada objeto sea parecido a los que hay en el conjunto de su conglomerado. Los conglomerados resultantes tendrán que tener un alto grado de homogeneidad interna (dentro del conglomerado) y de heterogeneidad externa (entre conglomerados).

Los principales objetivos perseguidos por un análisis clúster son:

- Elaboración de una tipología o clasificación.
- Investigación de esquemas conceptuales útiles para agrupar sujetos.
- Generación de hipótesis a través de exploraciones de datos.
- Comprobar si las hipótesis generadas a través de otros procedimientos se cumplen en la muestra de datos.

2.2.2. Etapas a seguir en el desarrollo del Análisis Clúster

Podemos distinguir tres pasos básicos para todos los estudios realizados con análisis cluster:

2.2.2.1. Selección de la muestra que se pretende dividir en grupos.

Por ejemplo países de la Unión Europea, compradores de un producto, etc. Pueden ser elegidos individuos, asociaciones, empresas o países. También es posible agrupar variables.

2.2.2.2. Selección de las variables que se van a utilizar para realizar el análisis.

Éste es uno de los pasos más críticos. Dentro de este paso se pueden distinguir los siguientes puntos:

- Número de variables: La elección del mismo es una decisión muy importante que va a condicionar en gran parte el éxito en los resultados de la investigación pero, a su vez, plantea grandes dificultades. Se tiende a pensar que con mayor número de variables se va a conseguir una mayor homogeneidad dentro de cada grupo. Esto es un error, ya que se pueden estar introduciendo variables que estén desvirtuando las semejanzas entre los sujetos al no estar estas relacionadas con el objetivo del estudio.
- Utilización de variables transformadas: La decisión aquí reside en si se utilizarán las variables tal cual son obtenidas o si se debe realizar alguna modificación. Una posible modificación en las variables es estandarizarlas, es decir, convertirlas en variables con media igual a cero y varianza igual a uno. Para hacer esto basta con restar a las variables la media y dividir el resultado por la desviación típica.

$$z = \frac{x_i - \bar{x}}{s} \quad [63]$$

donde:

\bar{x} = Media

s = Desviación típica

Realizar esta modificación puede reducir las diferencias entre grupos en aquellas variables que pudieran ser el mejor discriminante para las diferencias entre grupos. La estandarización puede suponer una transformación no equivalente entre las variables y podría cambiar las relaciones entre ellas. La decisión de estandarizar o no debe ser, por tanto, una de las primeras decisiones a tomar, y debe tenerse en cuenta que los resultados de la investigación pueden variar en función de que se haya realizado o no dicha estandarización.

La estandarización es útil especialmente en el caso de variables que están medidas en distintas escalas o magnitudes. Por ejemplo, si se quiere estudiar la semejanza entre los distintos países de la UE y se tienen en cuenta variables como el número de televisores por 1.000 habitantes, la renta per cápita, o el porcentaje de usuarios de Internet son variables que están medidas en unidades muy distintas la estandarización puede servir para compararlas entre sí.

Otro tema que se discute es si se pueden ponderar las variables para dar más importancia a alguna de ellas.

2.2.2.3. Cálculo de las similitudes o disimilitudes entre los casos o sujetos y validación de los resultados obtenidos.

Para realizar un AC se necesita una medida de similitud, de correspondencia, o parecido entre objetos que van a ser analizados. Las distintas medidas disponibles, en función de los tipos de datos que estemos analizando, son las enumeradas en el primer apartado de este módulo.

Una vez se tiene una matriz de similitud calculada, hay que realizar el proceso de partición de los datos, para ello hay que decidir el algoritmo de aglomeración utilizado en la formación de conglomerados, para después tomar una decisión acerca del número de conglomerados que se van a formar.

Los algoritmos de obtención de conglomerados se subdividen en:

- Métodos jerárquicos (encadenamiento simple, encadenamiento completo, encadenamiento medio, método de Ward, método del centroide).
- Métodos no jerárquicos (umbral secuencial, umbral paralelo, k-medias).
- O una combinación de los dos.

El dendrograma o gráfico en forma de árbol, es una herramienta visual que ayuda a decidir el número de conglomerados que podrían representar mejor la estructura de los datos.

Para determinar el número final de conglomerados a formar o regla de parada, no hay un procedimiento determinado, ha de decidirlo el investigador en la fase de interpretación de los datos.

2.2.3. Modelos jerárquicos

En el análisis clúster jerárquico se parte del número de individuos (países, empresas, etc.) y posteriormente se van uniendo en función de la mayor o menor proximidad de los individuos entre sí, formando grupos. Éstos a su vez se van uniendo entre sí hasta llegar a un único grupo. Obviamente, las dos decisiones que existen son:

1. La determinación de la medida de distancia o proximidad a usar: como hemos visto, la opción entre una u otra vendrá determinada por la medida en que los datos estén referidos.
2. El método que determinará el modo de unión sucesiva de los distintos grupos entre sí. Es decir, el que determinará la distancia existente entre los sucesivos grupos. Entre ellos encontramos los siguientes:
 - Vinculación inter-grupos: Según ella se define la distancia entre dos clústers (grupos) como la media de las distancias entre todas las combinaciones posibles dos a dos de los elementos de uno y otro grupo. Por ejemplo, si en primer lugar se han agrupado los individuos 1 y 2 la distancia entre el grupo (clúster) formado por éstos y el grupo 3 vendría dada por la media de las distancias 1-3 y 1-2. Usa pues los pares de distancias.
 - Vinculación intra-grupos: combina los grupos (clústers) de manera que la media de las distancias entre todos los pares de sujetos dentro del resultante sea la menor posible.

- Vecino más próximo o distancia mínima: los individuos que se combinan en cada grupo son aquellos que tienen una menor distancia o mayor similitud. Posteriormente se recalcula la distancia del clúster respecto al resto de casos formándose el siguiente mediante el mismo criterio.
- Vecino más lejano o distancia máxima: la distancia se calcula a partir de la distancia de los dos puntos más alejados.
- Método de Ward: El objetivo de este método es minimizar la varianza intra-grupos. Su funcionamiento es el siguiente: Se parte de n grupos formados todos ellos por un único punto (todos los individuos). En este momento la suma de las varianzas intra-grupo es cero. A continuación, se unirán dos grupos (individuos) en uno solo. Más concretamente, se unirán aquellos dos puntos que minimicen el incremento en la suma de las varianzas intra-grupo. El proceso continua del mismo modo sucesivamente.

2.2.3.1. Ejemplos con R

R incluye la función hclust para la realización de clasificaciones jerárquicas.

El investigador debe de suministrar dos opciones básicas a R:

- Medida de similitud/disimilitud para calcular la matriz de distancias.
- Estrategia de fusión.

Posteriormente, en la interpretación de los datos el investigador deberá decidir el punto de corte y, con ello, el número de grupos.

Se puede usar la función nativa de R dist para calcular la matriz de distancias; por defecto el programa usa la distancia euclídea (euclidean), siendo posible elegir varias más con la opción method="": maximum, manhattan, canberra, binary, minkowski.

Las estrategias de fusión disponibles en hclust son las siguientes:

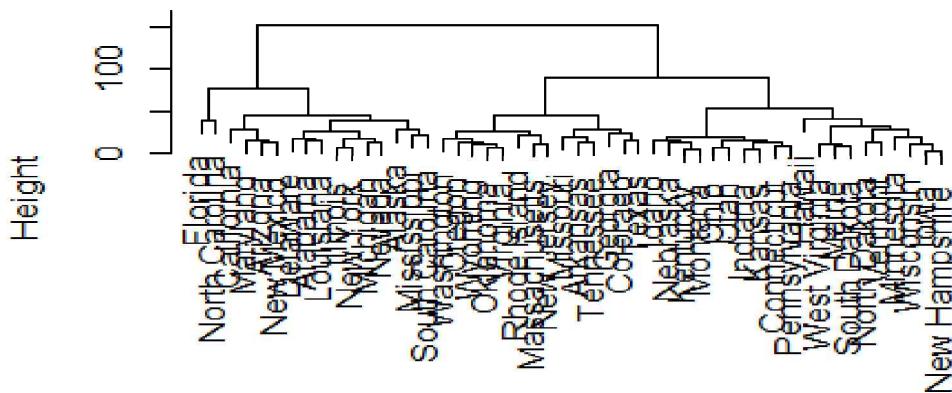
- "ward": la distancia entre dos conglomerados es la suma de los cuadrados entre dos conglomerados sumados para todas las variables. En cada paso del procedimiento de aglomeración se minimiza la suma de cuadrados dentro del conglomerado para todas las particiones. Tiende a combinar los conglomerados con un número reducido de dimensiones.
- "single": distancia mínima o vecino más próximo, encuentra dos objetos separados por la distancia más corta y los coloca en un primer conglomerado. A continuación se encuentra la distancia más corta, y se une a él un tercer objeto o se forma un nuevo conglomerado de dos miembros. El proceso continua hasta que todos los objetos están en un conglomerado.
- "complete": Parecido al "single"(simple) pero utilizando como criterio de agregación la distancia máxima (aproximación al vecino más lejano).
- "average": comienza igual que los métodos "single" y "complete", pero el método de aproximación es la distancia media entre todos los individuos de un conglomerado y los de otro.
- "mcquitty".
- "median".
- "centroid": la distancia entre dos conglomerados es la distancia (euclidia) entre centroides.

Realizamos un Clúster Jerárquico con las tasas de crimen en USA (USArrests), utilizando una medida de similaridad de distancias euclídeas y el método de agrupación de encadenamiento medio:

```
require(graphics)
str(USArrests)
'data.frame': 50 obs. of 4 variables:
 $ Murder : num 13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
 $ Assault : int 236 263 294 190 276 204 110 238 335 211 ...
 $ UrbanPop: int 58 48 80 50 91 78 77 72 80 60 ...
 $ Rape    : num 21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...

hc <- hclust(dist(USArrests), "ave")
plot(hc)
```

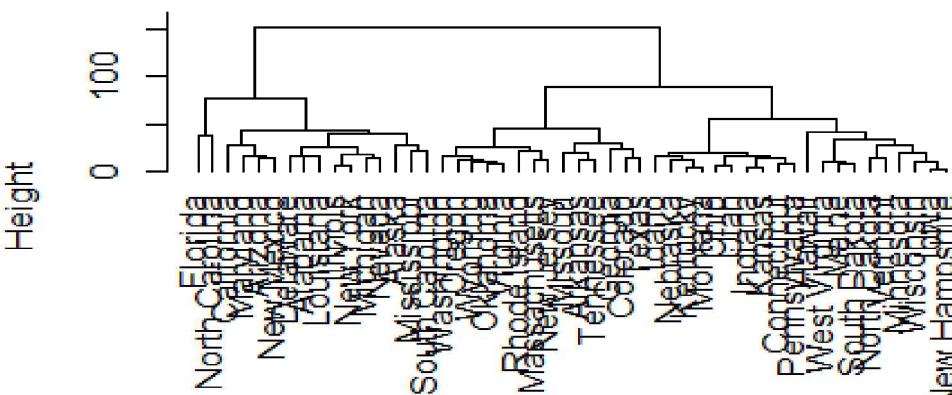
Cluster Dendrogram



```
dist(USArrests)
hclust (*, "average")
```

```
plot(hc, hang = -1)
```

Cluster Dendrogram



```
dist(USArrests)
hclust (*, "average")
```

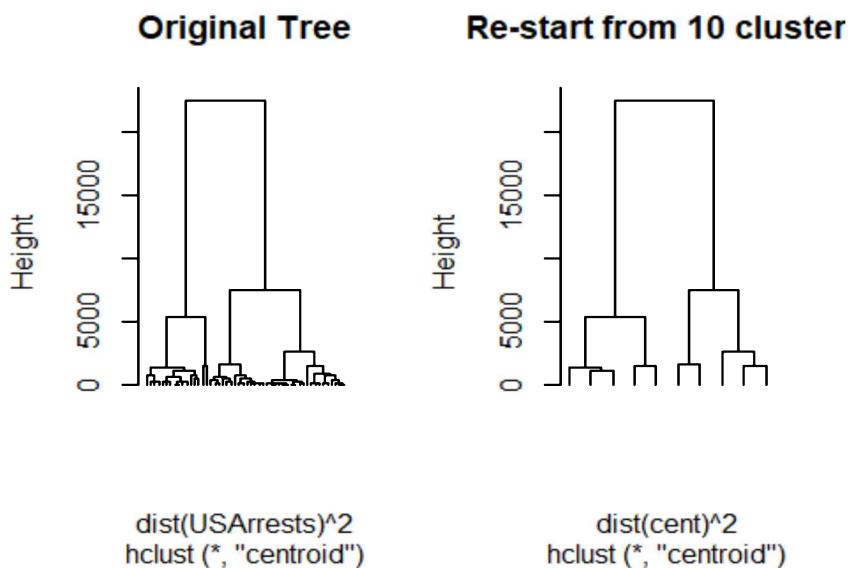
Se realiza el mismo clúster Jerárquico, se establece una parada de 10 clúster (cutree), y se realiza un nuevo clúster con los 10 conglomerados.

```
hc <- hclust(dist(USArrests)^2, "cen")

memb <- cutree(hc, k = 10)
cent <- NULL
for(k in 1:10){
  cent <- rbind(cent, colMeans(USArrests[memb == k, , drop = FALSE]))
}
str(cent)
num [1:10, 1:4] 11.47 13.5 9.95 11.5 5.59 ...
- attr(*, "dimnames")=List of 2
..$ : NULL
..$ : chr [1:4] "Murder" "Assault" "UrbanPop" "Rape"

hc1 <- hclust(dist(cent)^2, method = "cen", members = table(memb))

opar <- par(mfrow = c(1, 2))
plot(hc, labels = FALSE, hang = -1, main = "Original Tree")
plot(hc1, labels = FALSE, hang = -1, main = "Re-start from 10 clusters")
```



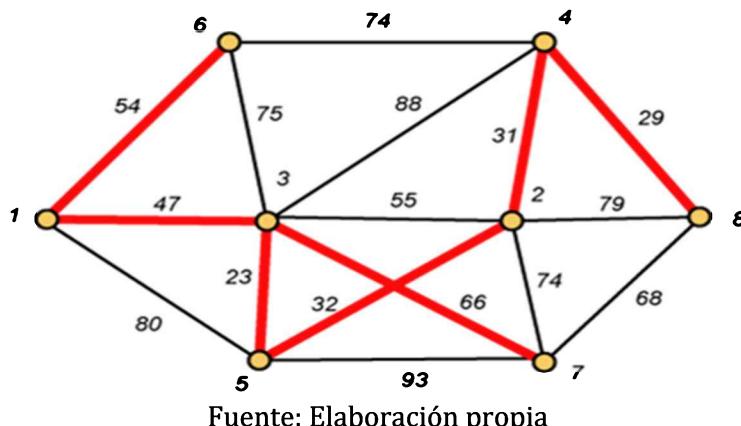
Clúster Jerárquico con span-tree

Si al clasificar un conjunto de datos no encontramos una decisión adecuada para establecer una regla de parada, y obtenemos como resultado un conjunto inclasificable, es útil combinar la metodología de jerarquía y topología de los árboles de expansión mínima (MST, Minimal Spanning Tree).

Dado un grafo, su árbol mínimo generador (o árbol de peso mínimo o árbol mínimo de expansión) es un árbol que pasa por todos los vértices y que la suma de sus aristas es la de menor peso.

La gráfica siguiente ilustra el proceso, se trata de un árbol de expansión de 7 nodos (acciones), con las medidas de distancia que hay entre ellos. Partiendo de la distancia menor, 29, entre el nodo 8 y 4, se inicia el árbol, que en el nodo 4, encuentra la segunda mínima distancia en su enlace con el nodo 2, 31, y en el nodo 2, la menor distancia la encuentra con el nodo 5, operando de esta manera hasta tener enlazados todos los nodos. Como se puede observar en la Figura 2, un nodo puede tener, más de un enlace con otros nodos, tal y como ocurre en el gráfico, nodo 3.

FIGURA 2. ENLACES ENTRE NODOS



Fuente: Elaboración propia

Basandonos en el trabajo pionero de Mantegna (1999) sobre tasas de retornos en los mercados financieros, realizamos un ejercicio clasificatorio con la función span-tree, de la librería vegan, que ordena y establece dependencias entre diversos activos financieros utilizando datos de rendimientos mensuales de activos procedentes de Berndt's The Practice of Econometrics.

```
# read prices from csv file
bolsa.df = load("berndt.RData")
colnames(berndt.df)
[1] "Año"      "MOBIL"    "TEXACO"   "IBM"     "DEC"      "DATGEN"   "CONED"
[8] "PSNH"     "WEYER"    "BOISE"    "MOTOR"   "TANDY"    "PANAN"    "DELTA"
[15] "CONTIL"   "CITGRP"   "GERBER"  "GENMIL"   "MARKET"   "RKFREE"
#[1] "Año"      "MOBIL"    "TEXACO"   "IBM"     "DEC"      "DATGEN"   "CONED"   "PSNH"    "WEYER"
"BOISE"
#[11] "MOTOR"    "TANDY"    "PANAN"    "DELTA"   "CONTIL"   "CITGRP"   "GERBER"  "GENMIL"
"MARKET"   "RKFREE"

# create zooreg object - regularly spaced zoo object
library(zoo)

Attaching package: 'zoo'
The following objects are masked from 'package:base':
  as.Date, as.Date.numeric

berndt.z = zooreg(berndt.df[,-1], start=c(1978, 1), end=c(1987,12),frequency=12)
index(berndt.z) = as.yearmon(index(berndt.z))

start(berndt.z)
[1] "ene. 1978"

end(berndt.z)
[1] "dic. 1987"

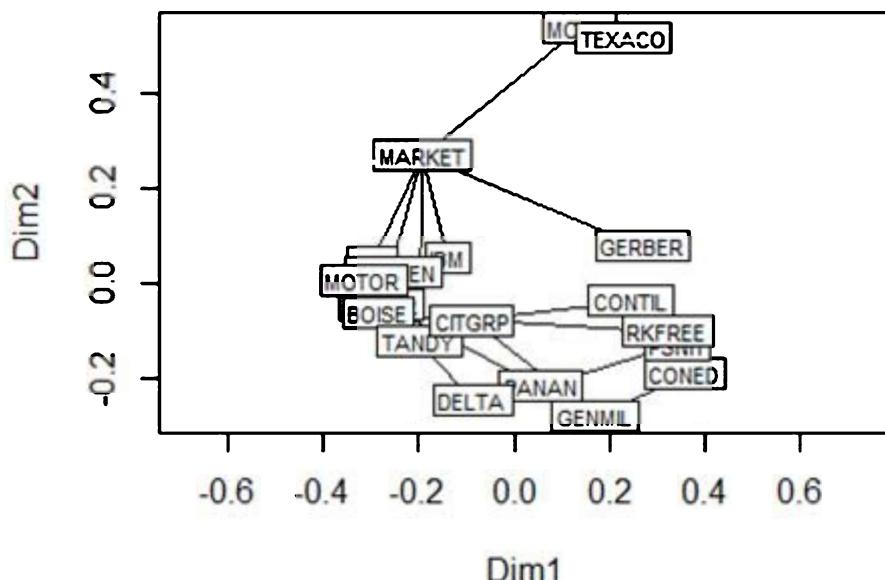
nrow(berndt.z)
[1] 120

# note: coredata() function extracts data from zoo object
returns.mat = as.matrix(coredata(berndt.z))

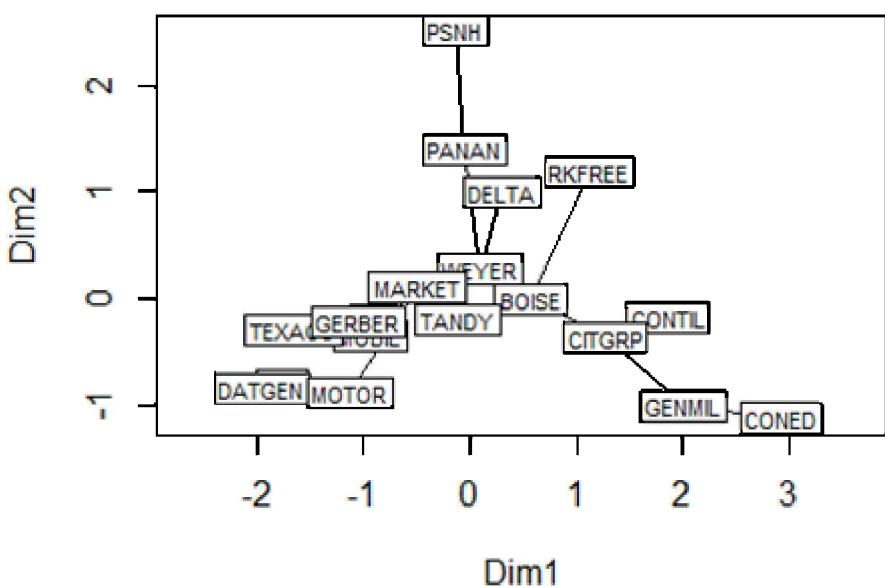
# create the correlation coefficients
coef.corr <- cor(returns.mat)
coef.d <- (1-coef.corr^2) # compute distance (Mantegna, 1998)

# hierarchical cluster whir hclust
d <- as.dist(as.matrix(coef.d)) # find distance matrix
```

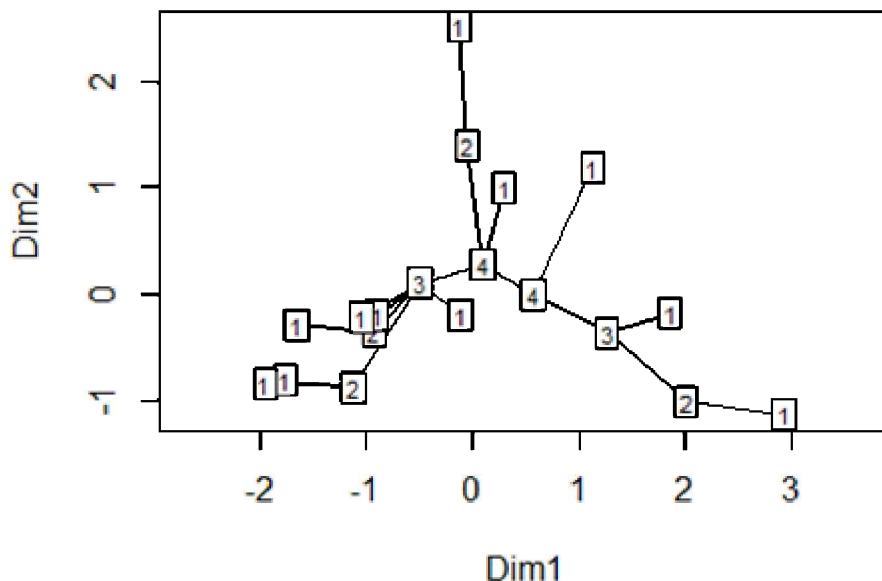
```
# Function spantree finds a minimum spanning tree (MTA) connecting all points, but
disregarding dissimilarities that are at or above the threshold or NA.
library(vegan)
tr <- spantree(d)
plot(tr, cmdscale(d), type = "t")
```



```
# Find a configuration to display the tree neatly
plot(tr, type = "t")
Initial stress      : 0.10907
stress after  2 iters: 0.06448
```

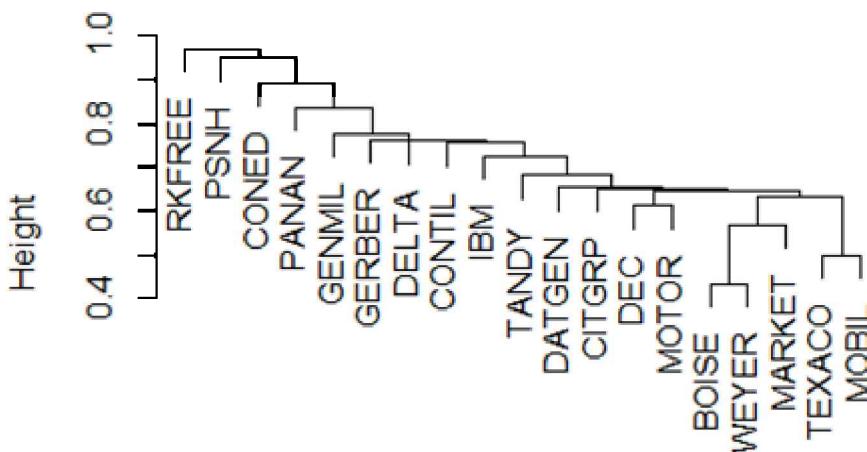


```
# Depths of nodes
depths <- spandepth(tr)
plot(tr, type = "t", label = depths)
Initial stress      : 0.10907
stress after  2 iters: 0.06448
```



```
# Plot as a dendrogram
cl <- as.hclust(tr)
plot(cl)
```

Cluster Dendrogram



```
tr
as.hclust.spantree (*, "spantree")
```

2.2.4. Modelos no jerárquicos

Tienen por objeto realizar una sola partición de los individuos en K grupos, lo que implica que previamente se debe fijar el número de grupos. Ésta es la principal diferencia con los modelos jerárquicos. El procedimiento a seguir es el siguiente:

1. Se comienza dividiendo los casos o puntos en un número prefijado de grupos.
2. Se calcula el centro (media) de cada uno de esos grupos.
3. Se reasigna cada punto al grupo de cuyo centro se encuentre más cercano.

4. Se vuelven a calcular los centros de cada grupo.
5. Se vuelven a repetir los pasos 3 y 4 hasta que ningún punto cambie de grupo.

A diferencia de los métodos de aglomeración jerárquica, que requieren el cálculo de una matriz de similitud, los métodos iterativos que usa este análisis trabajan directamente sobre la matriz inicial de datos.

El principal problema que presentan estos métodos es que el número de grupos debe ser previamente especificado por el investigador.

Uno de estos modelos es el análisis clúster de K medias que, al igual que el análisis discriminante, busca la maximización de $W - 1F$.

2.2.4.1. Ejemplo con R. Determinar el número de clúster con k-means

Determinar el número de clúster o la regla de parada es una de las tareas más sensibles en el AC. Se muestra a continuación una forma de incluir un test para decidir el número de conglomerados al utilizar la función k-means de R, y representar gráficamente el vector la suma de cuadrados entre conglomerados (withinss), para diferentes agrupamientos.

```
d=dist(USArrests)^2

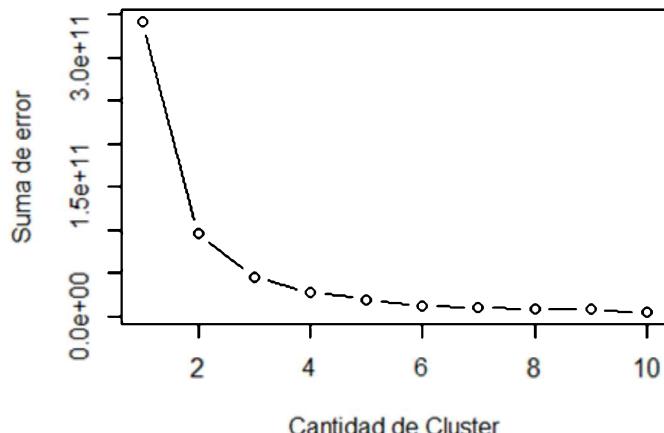
# Determinar el número de cluster

# Fija semilla inicial de números pseudoaleatorios
# para obtener misma serie aleatoria en ejecuciones.
set.seed(123)

# Crea vector "Errores", sin datos
# Crea variable "K_Max" con la cant. maxima de k a analizar
Errores <-NULL
K_Max   <-10

# Ejecuta kmeans con diferentes clúster, desde 1 hasta 10
# Luego guarda el error de cada ejecución en el vector "Errores"
for (i in 1:K_Max)
{
  Errores[i] <- sum(kmeans(d[-1], centers=i)$withinss)
}

# Grafica el vector "Errores"
plot(1:K_Max, Errores, type="b",
      xlab="Cantidad de Cluster",
      ylab="Suma de error")
```



2.3. ESCALAMIENTO MULTIDIMENSIONAL

El Análisis de Componentes Principales (ACP) se utiliza para obtener un “mapa” de los datos en dimensión reducida que preserve tanto como sea posible las distancias euclídeas entre las observaciones en su espacio original.

El objetivo del Escalamiento Multidimensional (Multidimensional Scaling, MDS) es similar, ya que pretende obtener una representación gráfica en dimensión reducida, pero a partir de una matriz de distancias entre los objetos o individuos. Esta matriz de distancias se puede calcular desde una matriz de datos (X) sobre algunas variables medidas o puede ser una matriz de distancias o similaridades obtenida directamente como, por ejemplo, de un juicio de expertos.

Existen muchos métodos de escalamiento multidimensional.

2.3.1. *Modelo general o método clásico*

Como los otros métodos, el método clásico tiene el objetivo de representar una matriz de distancias (o de similaridades) en un mapa o modelo geométrico.

Un modelo geométrico no es más que un conjunto de puntos (x_1, x_2, \dots, x_n) en un espacio de (q) dimensiones de forma que cada punto represente uno de los objetos o individuos y que las distancias entre dichos puntos se ajusten lo máximo posible a las distancias entre los objetos.

Así pues, el objetivo es doble, el primero es determinar la dimensión (q) del modelo y además obtener las coordenadas (de dimensión q) de los puntos (x_1, x_2, \dots, x_n) que proporcionan un “buen” ajuste a las distancias observadas o propuestas. Ese “buen” ajuste se deberá medir con algún tipo de medida o índice. La idea es que si dos objetos tienen una distancia grande entre ellos, los puntos que los representan estén suficientemente lejos y si, por el contrario, los objetos son cercanos sus representantes también lo sean.

La solución se basa en un trabajo de Young y Householder (1938) y hay que decir que no es única. El conjunto de coordenadas cuyas distancias se aproximan a las distancias originales se puede trasladar, rotar o reflectar, sin que las distancias se distorsionen, de manera que no se pueden determinar completamente. El problema de localización se resuelve tomando el punto con las medias como el origen de coordenadas. En cuanto a las rotaciones, el algoritmo proporciona una, pero cualquier otra puede servir para facilitar la interpretación de las soluciones.

Supongamos que la matriz de datos (X) sea tal que las distancias euclídeas entre sus filas sea la solución que buscamos. Dada una matriz como ésta, calcular las distancias euclídeas es fácil. El problema ahora es justamente el contrario: Dada una matriz de distancias (D) ($n \times n$), ¿cómo hallamos (X)?

Para empezar, definimos la matriz (B) ($n \times n$) como [$B = XX'$].

Los elementos de (B) son [$b_{ij} = \sum_{k=1}^q x_{ik} x_{jk}$].

Se puede comprobar que el cuadrado de las distancias euclídeas entre las filas de (X) puede escribirse en términos de los elementos de (B) como:

$$d_{ij}^2 = b_{ii} + b_{jj} - 2b_{ij} \quad [64]$$

Si en lugar de las distancias euclídeas ponemos las distancias de la matriz (B), entonces tenemos un sistema de ecuaciones para resolver. Dicho sistema de ecuaciones no es trivial, pero con algunas restricciones los elementos de (B) se pueden hallar en función de las distancias:

$$b_{ij} = -\frac{1}{2} [d_{ij}^2 - d_{i\cdot}^2 - d_{\cdot j}^2 + d_{\cdot \cdot}^2] \quad [65]$$

donde:

$$d_{i\cdot}^2 = \frac{1}{n} \sum_{j=1}^n d_{ij}^2 \quad [66]$$

$$d_{\cdot j}^2 = \frac{1}{n} \sum_{i=1}^n d_{ij}^2 \quad [67]$$

$$d_{\cdot \cdot}^2 = \frac{1}{n^2} \sum_{i=1}^n \sum_{j=1}^n d_{ij}^2 \quad [68]$$

Una vez hallados los elementos de B , solo nos queda factorizar para obtener las coordenadas (X).

La descomposición en valores singulares de (B) se puede escribir [$B = U\Lambda U'$] donde ($\Lambda = \text{diag}(\lambda_1, \dots, \lambda_n)$) es la matriz diagonal con los valores propios de (B) y (U) sus correspondientes vectores propios normalizados de forma que la suma de los cuadrados de los elementos de cada columna es 1. Los valores propios se ordenan de mayor a menor de forma que ($\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$).

Cuando (D) proviene de una matriz ($n \times q$) de rango máximo, entonces la matriz (B) es de rango (q) y sus últimos ($n - q$) valores propios son cero. Así que (B) se puede escribir como [$B = U_q \Lambda_q U_q'$] donde (Λ_q) contiene los primeros (q) valores propios y (U_q) sus correspondientes vectores propios.

Las coordenadas de los puntos que buscamos son [$X = U_q \Lambda_q^{1/2}$] donde ($\Lambda_q^{1/2} = \text{diag}(\lambda_1^{1/2}, \dots, \lambda_q^{1/2})$).

La mejor representación de dimensión (k) viene dada por los primeros (k) valores propios de (B) y sus correspondientes vectores propios. El ajuste de la representación en dimensión (k) a los datos originales se puede medir con el criterio [$P_k = \frac{\sum_{i=1}^k \lambda_i}{\sum_{i=1}^q \lambda_i}$]. Valores de (P_k) superiores a 0,8 sugieren un buen ajuste.

Cuando la matriz de disimilaridades no es euclídea, la matriz (B) no es definida positiva. En estos casos algunos de los valores propios de (B) serán negativos y, en consecuencia, algunas coordenadas serán números complejos. Sin embargo, si (B) tiene un número pequeño de valores propios negativos y éstos son pequeños, la representación asociada a los (k) mayores valores propios positivos, todavía es válida. En estos casos el criterio de ajuste puede ser [$P_k^{(1)} = \frac{\sum_{i=1}^k \text{abs}(\lambda_i)}{\sum_{i=1}^n \text{abs}(\lambda_i)}$] [$P_k^{(2)} = \frac{\sum_{i=1}^k \lambda_i^2}{\sum_{i=1}^n \lambda_i^2}$].

Algunas recomendaciones alternativas son:

1. *Criterio de la traza:* Elegir el número de coordenadas de forma que la suma de las que corresponden a valores propios positivos sumen aproximadamente lo mismo que la suma de todos los valores propios (positivos y negativos).

2. *Criterio de la magnitud:* Aceptar como genuinamente positivos los valores propios cuya magnitud excede substancialmente la del mayor valor propio negativo.

2.3.1.1. Ejemplos con R

Consideremos las distancias entre cinco objetos, dadas por la siguiente matriz:

```
d <- matrix(c(0, 1, 2, 4, 5, 1, 0, 2.5, 4, 5, 2, 2.5, 0, 4, 5, 4, 4, 4, 0, 1.2, 5, 5,
5, 1.2, 0), nrow = 5)
rownames(d) <- colnames(d) <- c("A", "B", "C", "D", "E")
d
  A   B   C   D   E
A 0 1.0 2.0 4.0 5.0
B 1 0.0 2.5 4.0 5.0
C 2 2.5 0.0 4.0 5.0
D 4 4.0 4.0 0.0 1.2
E 5 5.0 5.0 1.2 0.0
```

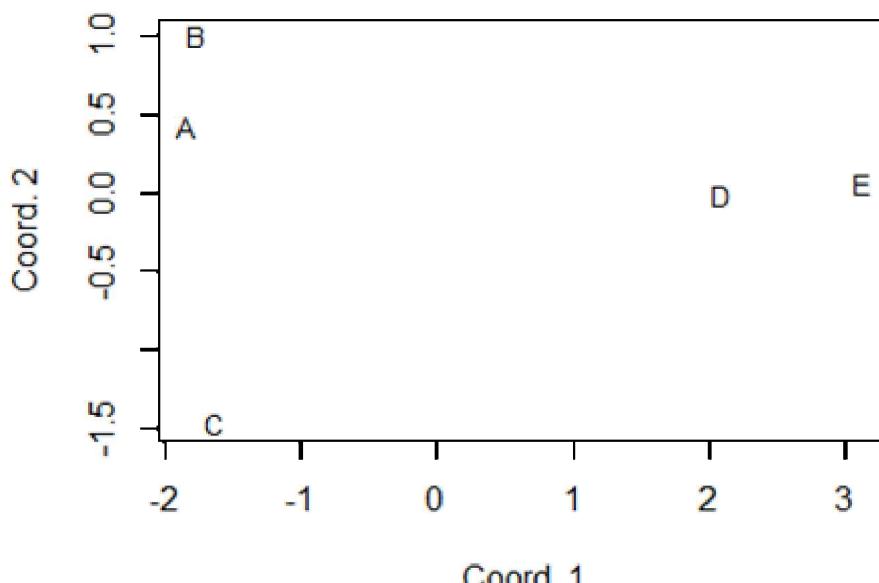
El escalamiento multidimensional se obtiene con la función `cmdscale()` de R. Las distancias obtenidas del mds las obtenemos a partir de la función `dist` aplicada al objeto mds.

```
mds <- cmdscale(d)
mds
  [,1]      [,2]
A -1.837444  0.42826525
B -1.757255  1.00509080
C -1.620166 -1.47854596
D  2.086176 -0.01557718
E  3.128689  0.06076709

dist(mds)
      A         B         C         D
B 0.5823727
C 1.9191505 2.4874173
D 3.9486436 3.9766471 3.9846260
E 4.9797124 4.9763642 4.9921049 1.0453055
```

Finalmente, podemos representar los objetos como puntos de un mapa de dimensión dos:

```
plot(mds, type = "n", xlab = "Coord. 1", ylab = "Coord. 2")
text(mds[, 1], mds[, 2], labels = rownames(mds), cex = 0.8)
```



Los dos primeros valores propios y el ajuste obtenido:

```
mds <- cmdscale(d, eig = T)
mds$eig[1:2]
[1] 23.229909 3.383652

mds$GOF
[1] 0.9806751 0.9806751
```

El siguiente ejemplo, representa en un mapa de coordenadas las distancias entre las principales ciudades europeas.

```
loc <- cmdscale(eurodist)
x <- loc[, 1]
y <- -loc[, 2] # para reflejar el norte en la parte superior

# asp = 1, para asegurar que las distancias euclidianas estén representadas
correctamente
plot(x, y, type = "n", xlab = "", ylab = "", asp = 1, axes = FALSE, main =
"cmdscale(eurodist)")
text(x, y, rownames(loc), cex = 0.6)
```

cmdscale(eurodist)



2.3.2. Otros modelos de escalamiento

La forma más habitual de introducción de los datos en un análisis de escalamiento multidimensional es mediante una matriz de proximidades o distancias. Es decir, se parte de las cercanías o lejanías de los distintos objetos o sujetos entre sí.

Esta matriz de proximidades puede ser básicamente de tres clases:

- a) Cuadrada simétrica.
- b) Cuadrada asimétrica.
- c) Rectangular.

El ejemplo clásico de esta técnica, el que acabamos de ver, es el de la tabla en la que se representan las distancias en kilómetros. La información se ofrece a través de una matriz cuadrada simétrica de datos. Es cuadrada al tener el mismo número de filas que de columnas y es simétrica porque la distancia de Madrid a Valencia es igual que de Valencia a Madrid y así sucesivamente.

Una segunda posibilidad es la de introducir los datos a través de una matriz cuadrada asimétrica. En ella la información no es equivalente. Supongamos que se pide a cuatro individuos que evalúen la mayor o menor simpatía hacia otros cuatro usando una escala ordinal que va de 1 a 5. En este caso, la información no tiene porqué ser simétrica ya que el nivel de simpatía que tiene el individuo 1 por el 5 no tiene porqué ser igual que la que tiene el 5 por el 1 y así sucesivamente.

En las matrices rectangulares no coincide el número de filas con el de columnas. Un ejemplo clásico es el de distintos individuos que evalúan distintos aspectos de un producto o bien. Los datos procedentes de una matriz rectangular son los que se conocen como datos multivariados o “datos de perfil” porque cada fila representaría el perfil que corresponde a un individuo. En este caso la matriz de datos no sería de proximidades por lo que habrá que transformarla a una matriz de proximidades. Una excepción a esta regla la constituyen las matrices de datos de preferencia, donde cada elemento p_{ij} de la matriz representa la preferencia que siente el sujeto i por el estímulo j.

Las proximidades pueden estar medidas en una escala ordinal, de intervalo o de proporción. El modelo de escalamiento estará determinado por la medida, de tal modo que cuando nos encontramos ante escalas de intervalo o razón el modelo a aplicar será métrico, mientras que en el caso ordinal será no métrico.

Cuando los datos están en escala nominal es necesario transformar la medida usando cualquiera de las medidas de distancia que se vieron en el primer apartado de este módulo.

En el modelo de escalamiento *métrico*, partimos del supuesto de que la relación entre las proximidades y las distancias es de tipo lineal. El procedimiento consiste en transformar la matriz de proximidades en una matriz de distancias de tal forma que verifique los tres axiomas de la distancia euclídea:

1. No negatividad: $d_{ij} \geq 0 = d_{ii}$.
2. Simetría: $d_{ij} = d_{ji}$.
3. Desigualdad triangular: $d_{ij} \leq d_{ik} + d_{kj}$.

Los dos primeros axiomas son fáciles de cumplir, pero el tercer axioma no se cumple siempre. Este problema se conoce con el nombre de “estimación de la constante aditiva”. Torgerson solucionó este problema, estimando el valor mínimo de c que verifica la desigualdad triangular de la siguiente forma:

$$c_{min} = \max_{(i,j,k)} \{ \delta_{ij} - \delta_{ik} - \delta_{kj} \} \quad [69]$$

De esta forma las distancias se obtienen sumando a las proximidades la constante c, es decir, $d_{ij} = \delta_{ij} + c$.

A diferencia del escalamiento métrico, el modelo de escalamiento **no métrico** no presupone una relación lineal entre las proximidades y las distancias, sino que establece una relación monótona creciente entre ambas, es decir, si $\delta_{ij} < \delta_{kl} \Rightarrow d_{ij} \leq d_{kl}$. Su desarrollo se debe a Shepard (1962) quién demostró que es posible obtener soluciones métricas asumiendo únicamente una relación ordinal entre proximidades y distancias. Posteriormente Kruskal (1964) mejoró el modelo.

Tanto para el modelo métrico como para el modelo no métrico es necesario obtener un coeficiente que nos informe sobre la bondad del modelo. Sabemos que las distancias son una función de las proximidades, es decir:

$$f: \delta_{ij}(x) \rightarrow d_{ij}(x) \quad [70]$$

De esta forma se tiene que $d_{ij} = f(\delta_{ij})$. Esto no deja ningún margen de error, sin embargo, en las proximidades empíricas es difícil que se dé la igualdad, con lo que generalmente ocurre que $d_{ij} \approx f(\delta_{ij})$. A las transformaciones de las proximidades por f se le denomina disparidades. A partir de aquí podemos definir el error cuadrático como:

$$e_{ij}^2 = (f(\delta_{ij}) - d_{ij})^2 \quad [71]$$

Como medida que nos informa de la bondad del modelo podemos utilizar el Stress que Kruskal definió como:

$$\text{Stress} = \sqrt{\frac{\sum_{i,j} (f(\delta_{ij}) - d_{ij})^2}{\sum_{i,j} d_{ij}^2}} \quad [72]$$

Mientras mayor sea la diferencia entre las disparidades y las distancias, es decir, entre $f(\delta_{ij})$ y d_{ij} , mayor será el Stress y por tanto peor será el modelo. Por tanto, el Stress no es propiamente una medida de la bondad del ajuste, sino una medida de la no bondad o "maldad" del ajuste. Su valor mínimo es 0, mientras que su límite superior para n estímulos es $\sqrt{1 - 2/n}$.

Kruskal (1964) sugiere las siguientes interpretaciones del Stress:

- 0.2 → Pobre
- 0.1 → Aceptable
- 0.05 → Bueno
- 0.025 → Aceptable
- 0.0 → Excelente

También se suele utilizar una variante del Stress que se denomina S-Stress, definida como:

$$S - \text{Stress} = \sqrt{\frac{\sum_{i,j} (f(\delta_{ij})^2 - d_{ij}^2)^2}{\sum_{i,j} d_{ij}^2}} \quad [73]$$

Otra medida que se suele utilizar es el coeficiente de correlación al cuadrado (RSQ), que nos informa de la proporción de variabilidad de los datos de partida que es explicada por el modelo. Los valores que puede tomar oscilan entre 0 y 1, al ser un coeficiente de correlación al cuadrado. Valores cercanos a 1 indican que el modelo es bueno y valores cercanos a 0 indican que el modelo es malo. Su expresión es:

$$RSQ = \frac{[\sum_{i,j} (d_{ij} - d_{..})(f(\delta_{ij}) - f(\delta_{..}))]^2}{[\sum_{i,j} (d_{ij} - d_{..})^2][\sum_{i,j} (f(\delta_{ij}) - f(\delta_{..}))^2]} \quad [74]$$

La mayoría de los paquetes estadísticos tienen implementados tanto los algoritmos para obtener soluciones con MDS así como las medidas para determinar si el modelo es adecuado

o no. En la actualidad todos los algoritmos implementados en los paquetes estadísticos son reiterativos, de forma que se alcance la mejor solución posible.

Junto al modelo general, en función de la matriz de proximidades de entrada y la escala de medida, encontramos otros muchos.

Entre ellos, cabe mencionar el modelo desdoblado, el modelo INDSCAL, y el modelo con replicación. Sin embargo, existen muchas otras variantes como el modelo GEMSCAL, modelos para matrices asimétricas como el modelo ASCAL o el modelo AINDS, que por su extensión no abordaremos. Cada uno de ellos se confeccionó para resolver un tipo de problema distinto .

2.3.2.1. Ejemplos con R

✓ **Medición estandarizada de la fecundidad e indicadores socioeconómicos. MDS no métrico**

Para este ejemplo vamos a utilizar el dataset **swiss** que contiene la siguiente información para cada una de las 47 provincias francófonas de Suiza alrededor de 1888:

- Medida de fertilidad estandarizada común.
- % de hombres involucrados en la agricultura como ocupación.
- % reclutas que reciben la calificación más alta en el examen del ejército.
- % educación más allá de la escuela primaria para reclutas.
- % católico' (en oposición a 'protestante').
- Nacidos vivos que viven menos de 1 año.

Todas las variables excepto 'Fertilidad' dan proporciones de la población. Para este análisis, se excluye dicha variable.

La función que aplica el MDS no métrico es *isoMDS* del paquete MASS.

```
library(MASS)
# MDS no métrico
# 47 filas (objectos) x 5 columnas (variables) cada una identificada por un nombre
único

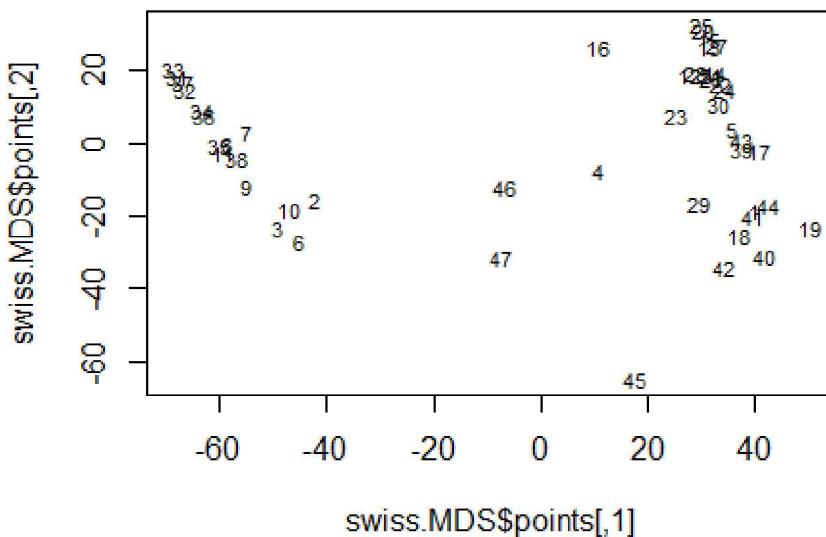
# Cargo datos y elimino primera variable, fertilidad
datos.swiss <- as.matrix(swiss[, -1])

# Distancias euclídeas entre las filas
d <- dist(datos.swiss)

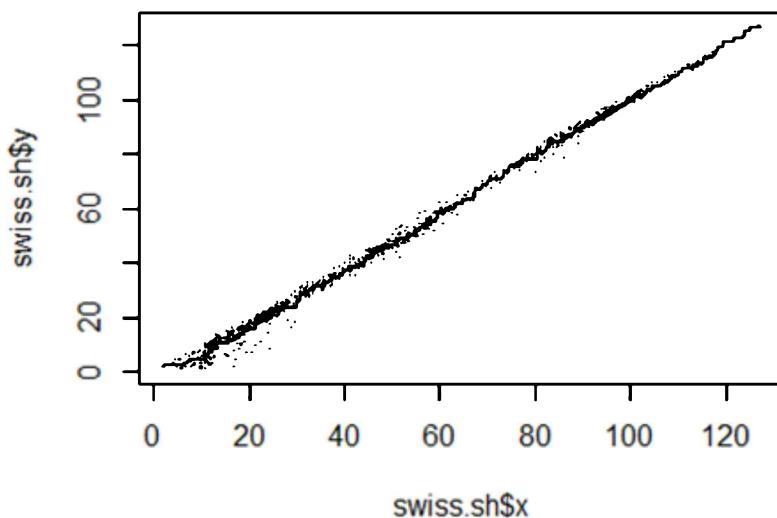
# Ajuste de modelo, con k igual a número de dimensiones
swiss.MDS <- isoMDS(d, k=2)
  initial value 2.979731
  iter  5 value 2.431486
  iter 10 value 2.343353
  final value 2.338839
  converged

# Representación gráfica
plot(swiss.MDS$points, type = "n", main="MDS no métrico")
text(swiss.MDS$points, labels = as.character(1:nrow(datos.swiss)), cex = 0.7)
```

MDS no métrico



```
# Ajuste
swiss.sh <- Shepard(d, swiss.MDS$points)
plot(swiss.sh, pch = ".")
lines(swiss.sh$x, swiss.sh$yf, type = "S")
```



✓ Estudio de mercado sobre vinos. Modelo INDSCAL de diferencias individuales

Los datos utilizados aquí se refieren a 10 vinos franceses diferentes evaluados por 11 panelistas. Se les pidió posicionar los vinos en un mantel de dimensiones (60,40).

Así, cada fila del data set *napping.don* de dimensión (10,22) representa un vino francés, cada pareja (X_i, Y_i) representa las coordenadas de los vinos colocados en una mesa, para un panelista determinado.

Por otra parte, también se les pidió que describieran cada vino usando su propia lista de palabras.

Cada fila del data set *napping.words* de dimensión (10, 14) representa un vino francés, cada columna es un atributo, y cada celda es la cantidad de veces que un atributo determinado se citó para un vino determinado.

Esta versión del modelo Indscal está especialmente adaptada al tipo de datos Napping, es decir, los productos (estímulos) se colocan en una mesa por los panelistas, y luego sus coordenadas se usan como entrada para el modelo Indscal.

Las funciones devuelven los tres gráficos siguientes:

- Una representación de estímulos, es decir, una representación de los productos.
- Una representación de los pesos calculados por el modelo Indscal.
- Un círculo de correlación de las variables mejorado por variables ilustrativas (columnas suplementarias).

```
library(SensoMineR)
data(napping)
nappeplot(napping.don)

resindscal<- indscal(napping.don, napping.words)
resindscal
$W
      Dim 1      Dim 2
Y1  0.8587087 0.1317172
Y2  0.1304253 0.2977809
Y3  0.7873545 0.0000000
Y4  0.0000000 0.8521910
Y5  0.6114389 0.3283329
Y6  0.0000000 0.2658146
Y7  0.1449778 0.7667908
Y8  0.3512626 0.2483905
Y9  0.6774071 0.2622057
Y10 0.0000000 0.8643238
Y11 0.7066128 0.0000000

$points
      Dim 1      Dim 2
1 T Michaud   -0.11940626 0.2838063
2 T Renaudie   0.02032120 0.3700653
3 T Trotignon  -0.29235413 0.4788932
4 T Buisse Domaine 0.01605128 0.2076785
5 T Buisse Cristal 0.16843998 0.1036647
6 V Aub. Silex   0.11007659 -0.1529218
V Aub. Marigny   0.45268744 -0.3373842
8 V Font. Domaine -0.47591517 -0.1219007
V Font. Brules   -0.39846909 -0.3915849
10 V Font Coteaux 0.51856817 -0.4403164

$subvar
      Y1      Y2      Y3      Y4      Y5      Y6
0.77927937 0.11411387 0.60843766 0.71907295 0.52523305 0.07061606
      Y7      Y8      Y9      Y10     Y11
0.63311502 0.20402052 0.56618379 0.73900713 0.49850733

$r2
[1] 0.4961442

$dfr
[1] 0.07676768

attr("class")
[1] "indscal"

prefpls(cbind(resindscal$points, napping.words))
```

2.3.3. Relación con otras técnicas multivariantes

El MDS puede ser utilizado en muchas investigaciones junto a otras técnicas multivariantes, bien como una alternativa a dichas técnicas o bien como un complemento a las mismas. La utilización de cada una de ellas va a depender de los objetivos que se persigan en la investigación. Por tanto, no hay una técnica mejor que otra, sino que en algunos casos será más apropiado utilizar una técnica que en otros.

Entre las ventajas de utilizar el MDS en comparación con otras técnicas multivariantes están:

- Los datos en MDS pueden estar medidos en cualquier escala, mientras que en el análisis factorial deben estar medidos en escala de razón o intervalo.
- El MDS proporciona soluciones para cada individuo, lo cual no es posible con el análisis factorial ni con el análisis cluster.
- En el MDS el investigador no necesita especificar cuáles son las variables a emplear en la comparación de objetos, algo que es fundamental en el análisis factorial y en el análisis cluster, con lo que se evita la influencia del investigador en el análisis.
- Las soluciones proporcionadas por MDS suelen ser de menor dimensionalidad que las proporcionadas por el análisis factorial (Schiffman, Reynolds y Young, 1981).
- En MDS pueden ser interpretados directamente las distancias entre todos los puntos, mientras que en el análisis de correspondencias solamente pueden ser interpretadas directamente las distancias entre filas o bien entre columnas.

2.4. ANÁLISIS DE CORRELACIÓN CANÓNICA

2.4.1. Introducción

El Análisis de Correlación Canónica (CCA) es un método estadístico exploratorio multidimensional en la misma línea que el análisis de componentes principales (PCA): ambos métodos se basan en el mismo fondo matemático (álgebra matricial y análisis propio) y los resultados se pueden ilustrar mediante representaciones gráficas similares.

El propósito principal de CCA es la exploración de correlaciones muestrales entre dos conjuntos de variables cuantitativas observadas en las mismas unidades experimentales.

2.4.2. Modelo

Consideremos dos matrices X e Y de dimensiones $(n \times p)$ y $(n \times q)$ respectivamente. Las columnas de X e Y corresponden a variables y las filas corresponden a unidades experimentales.

La j -ésima columna de la matriz X se denota por X^j , asimismo la k -ésima columna de Y se denota por Y^k .

Sin pérdida de generalidad, se supondrá que las columnas de X e Y están estandarizadas (media 0 y varianza 1).

Además, se supone que $p \leq q$ (en otras palabras, el grupo que contiene las menos variables se denota por X). Denotamos por S_{XX} y S_{YY} las matrices de covarianza muestral para los

conjuntos de variables X e Y respectivamente, y por $S_{XY} = S_{YX}$ la matriz de covarianza cruzada muestral entre X e Y.

El CCA clásico asume que $p \leq n$ y $q \leq n$, entonces las matrices X e Y son de rango de columna completo p y q, respectivamente. El CCA se presenta entonces como un problema resuelto a través de un algoritmo iterativo.

La primera etapa de CCA consiste en encontrar dos vectores, a^1 y b^1 que maximicen la correlación entre las combinaciones lineales:

$$U^1 = Xa^1 = a_1^1 X^1 + a_2^1 X^2 + \cdots + a_p^1 X^p \quad [75]$$

y

$$V^1 = Yb^1 = b_1^1 Y^1 + b_2^1 Y^2 + \cdots + b_q^1 Y^q \quad [76]$$

asumiendo que los vectores a^1 y b^1 están normalizados, es decir: $\text{var}(U^1) = \text{var}(V^1) = 1$.

El problema consiste en resolver:

$$\rho_1 = \text{cor}(U^1, V^1) = \max_{a,b} \text{cor}(Xa, Yb) \quad [77]$$

sujeto a la restricción:

$$\text{var}(Xa) = \text{var}(Yb) = 1 \quad [78]$$

Las variables resultantes U^1 y V^1 se denominan las primeras variables canónicas y ρ_1 se denomina la primera correlación canónica.

Las variables y correlaciones canónicas de orden superior se pueden encontrar como un problema escalonado.

Para $s = 1, \dots, p$, podemos encontrar sucesivamente correlaciones positivas $\rho_1 \geq \rho_2 \geq \dots \geq \rho_p$ con vectores correspondientes $(a^1, b^1), \dots, (a^p, b^p)$, maximizando:

$$\rho_s = \text{cor}(U^s, V^s) = \max_{a^s, b^s} \text{cor}(Xa^s, Yb^s) \quad [79]$$

bajo la restricción adicional:

$$\text{cor}(U^s, U^t) = \text{cor}(V^s, V^t) = 0 \quad 1 \leq t < s \leq p \quad [80]$$

Las correlaciones canónicas se calculan a partir de los valores positivos de las raíces cuadradas de los autovalores de $P_X P_Y$, siendo éstas las matrices de proyección ortogonal sobre las columnas de X e Y, definidas como:

$$P_X = X(X^T X)^{-1} X^T = \frac{1}{n} X S_{XX}^{-1} X^T \quad [81]$$

$$P_Y = Y(Y^T Y)^{-1} Y^T = \frac{1}{n} Y S_{YY}^{-1} Y^T \quad [82]$$

$$\rho_s = \sqrt{\lambda_s} \quad [83]$$

Los vectores U^1, \dots, U^p son los vectores propios estandarizados correspondientes a los valores propios en orden descendente ($\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_p$) de $P_X P_Y$.

Los vectores V^1, \dots, V^p son los vectores propios estandarizados correspondientes a los mismos valores propios ($\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_p$) de $P_Y P_X$.

El CCA no se puede realizar cuando el número de unidades experimentales es menor que la mayor cantidad de variables en ambos conjuntos de datos ($n \leq \text{Max}(p, q)$). De hecho, cuando aumenta el número de variables, las correlaciones canónicas más altas son casi 1 debido a la recuperación de subespacios canónicos que no proporcionan ninguna información significativa. Por lo tanto, una condición estándar generalmente recomendada para CCA (Eaton y Perlman, 1973) es $n \geq p + q + 1$.

Además, cuando las variables $X_1 \dots X_p$ y/o $Y_1 \dots Y_q$ están altamente correlacionados, es decir, casi colineales, las matrices S_{XX} y/o S_{YY} respectivamente, tienden a estar mal condicionadas y sus inversas no son confiables.

Una forma de lidiar con este problema consiste en incluir un paso de regularización en los cálculos. Dicha regularización en el contexto de CCA fue propuesta por primera vez por Vinod (1976), luego desarrollada por Leurgans *et al.* (1993).

En este marco, S_{XX} y S_{YY} se reemplazan respectivamente por $\Sigma_{XX}(\lambda_1)$ y $\Sigma_{YY}(\lambda_2)$ definidas como:

$$\Sigma_{XX}(\lambda_1) = S_{XX} + \lambda_1 I_p \quad [84]$$

$$\Sigma_{YY}(\lambda_2) = S_{YY} + \lambda_2 I_q \quad [85]$$

Este método plantea un nuevo problema: ¿cómo establecer valores adecuados para los parámetros de regularización? Este problema se aborda a través de un procedimiento de validación cruzada.

2.4.3. Interpretación de resultados

Al igual que en el análisis de componentes principales, se recomienda elegir un valor pequeño para la dimensión d ($1 \leq d \leq p$). En la práctica, este valor suele ser 2, 3 o 4. Obsérvese que las pequeñas correlaciones canónicas no son relevantes: no expresan relaciones lineales entre columnas de X e Y y pueden desecharse.

Para valores grandes de p , se sugiere un enfoque empírico para elegir la dimensión basada en el examen conjunto de dos representaciones gráficas: el gráfico de las correlaciones canónicas y las gráficas de dispersión de las variables. Una brecha clara entre dos valores sucesivos del gráfico de las correlaciones canónicas versus la dimensión sugiere seleccionar para d el rango del más grande. Por otro lado, consideramos el diagrama de dispersión de variables de acuerdo con los ejes (U^s, U^{s+1}) para los primeros valores de s y desechar los ejes en los que casi todos los puntos que representan variables X o Y se encuentran dentro del círculo de radio 0.5 (es decir, las correlaciones entre las variables X^j o Y^k y las variables canónicas U^s o V^s son menores que 0.5).

✓ Gráficos de variables

La gráfica de variables es interesante porque permite discernir la estructura de correlación entre los dos conjuntos de variables X e Y . Las coordenadas de las variables X^j e Y^k en el eje definido por U^s son las correlaciones de Pearson entre estas variables y U^s . Como se supone que las variables X^j e Y^k son de varianza unitaria, sus proyecciones en el plano definido por los ejes (U^s, U^t) se encuentran dentro de un círculo de radio 1 centrado en el origen, llamado *círculo de correlación*. En este gráfico, se trazan dos circunferencias

correspondientes al radio 0.5 y 1 para revelar los patrones más sobresalientes en el anillo definido entre estas dos circunferencias. Las variables con una relación fuerte se proyectan en la misma dirección desde el origen. Cuanto mayor es la distancia desde el origen, más fuerte es la relación.

✓ Gráficos de unidades

La representación de las unidades puede ser útil para aclarar la interpretación de la correlación entre variables.

Esta representación de unidades es posible usando los ejes definidos por (U^s, U^t) : la coordenada de la i-ésima unidad en el eje U^s es U_i^s (la i-ésima coordenada de la s-ésima variable canónica).

Las relaciones entre los dos gráficos (variables y unidades) dibujadas en los ejes coincidentes pueden revelar asociaciones entre variables y unidades.

2.4.4. Ejemplo en R

Como ejemplo de análisis, veremos el desarrollado en la revista *Journal of Statistical Software*.

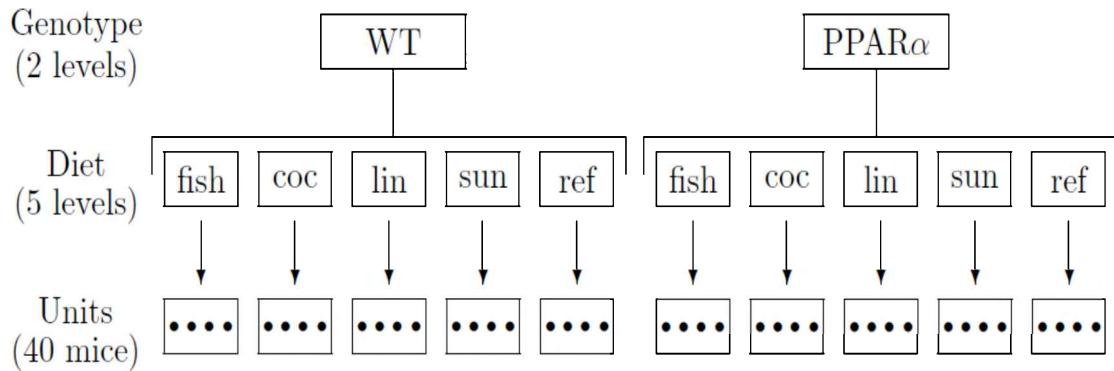
Utilizaremos el data set *nutrimouse*, que proviene de un estudio de nutrición en ratones (Martin *et al.*, 2007). Cuarenta ratones fueron estudiados. Se tomaron dos conjuntos de variables:

- expresiones de 120 genes medidos en células hepáticas, seleccionadas (entre aproximadamente 30000) como potencialmente relevantes en el contexto del estudio de nutrición.
- concentraciones de 21 ácidos grasos hepáticos (FA) medidos por cromatografía de gases.

Las unidades biológicas (ratones) se clasifican de forma cruzada de acuerdo con dos factores:

- Genotipo: se realizó estudio en tipo salvaje (*WT*) y ratones deficientes (*PPAR α*).
- Dieta: los aceites utilizados para la preparación de dietas experimentales fueron aceites de maíz y colza (50/50) para una dieta de referencia (REF), aceite de coco hidrogenado para una dieta saturada de FA (COC) y aceite de girasol para una dieta rica en $\omega 6$ FA (SUN), aceite de linaza para una dieta rica en $\omega 3$ FA (LIN) y aceites de pescado enriquecidos con maíz / colza para la dieta (FISH) (42.5 / 42.5 / 15).

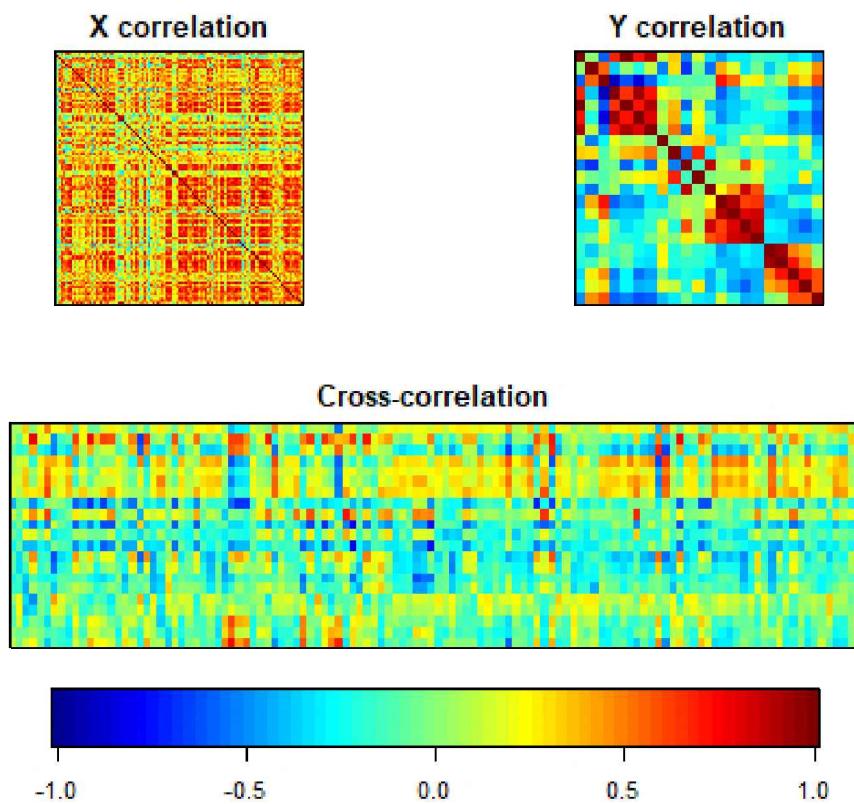
El esquema del experimento es el siguiente:



Convertimos en primer lugar los datos en matriz y efectuamos el análisis preliminar.

```
library(CCA)
data(nutrimouse)
X <- as.matrix(nutrimouse$gene)
Y <- as.matrix(nutrimouse$lipid)

correl <- matcor(X, Y)
img.matcor(correl, type = 2)
```



El gráfico resalta algunas correlaciones importantes no solo dentro de cada conjunto de variables (matrices cuadradas 120 x 120 y 21 x 21 en la parte superior), sino también entre ambos conjuntos (la matriz rectangular 21 x 120 en la parte inferior), es decir, entre expresión génica y concentración de ácidos grasos.

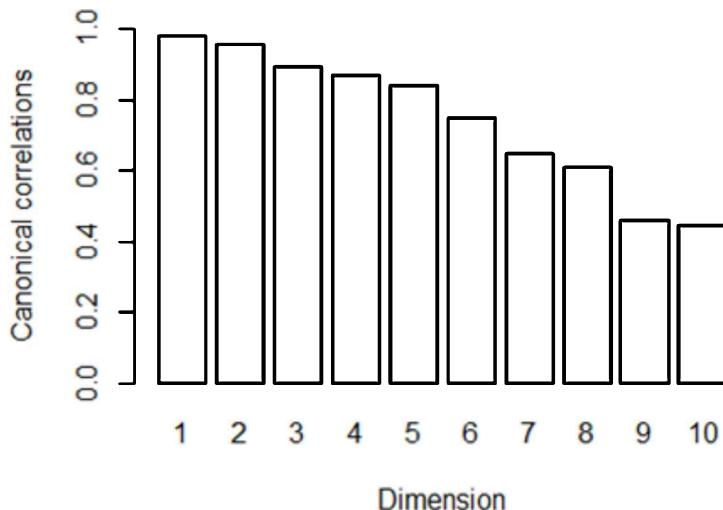
El trabajo debe detenerse aquí si las imágenes obtenidas son uniformes en color verde claro correspondiente a una correlación casi nula.

Cuando se trata de conjuntos de datos en los cuales el número de unidades experimentales es mayor que el número de variables, el CCA clásico se puede realizar con la función cc(). Las siguientes líneas de comando ilustran su uso con un número restringido de genes del conjunto de datos X.

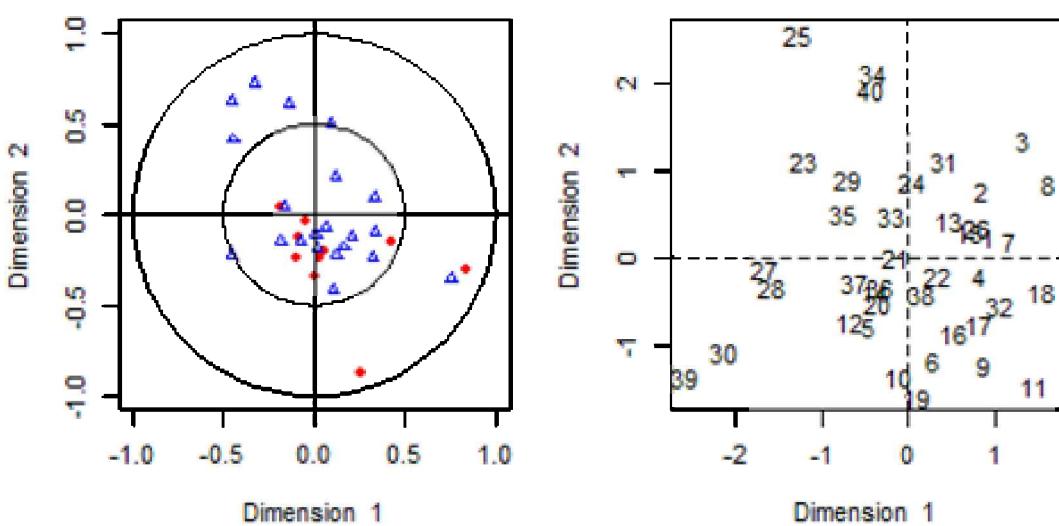
```
# elegimos al azar 10 genes entre los 120. Se almacenan como una matriz
# Xr (40 x 10) e Y (40 x 21)
Xr <- as.matrix(nutrimouse$gene[, sample(1:120, size = 10)])

# Análisis
res.cc <- cc(Xr, Y)

# Gráficos
barplot(res.cc$cor, xlab = "Dimension", ylab = "Canonical correlations", names.arg =
1:10, ylim = c(0,1))
```



```
plt.cc(res.cc)
```



Cuando se trata de todo el conjunto de datos (X (40 x 120) e Y (40 x 21)), no se puede realizar el CCA clásico y debe incluirse en el procesamiento de datos el paso de regularización.

Elección de los parámetros de regularización: se opta por construir una cuadrícula alrededor de los valores razonables para λ en el cual detectamos la celda donde la puntuación CV alcance su máximo.

La función `estim.regul()` implementa el proceso de validación cruzada de leave-one-out. La cuadrícula predeterminada en la que se calcula el criterio CV es regular con 5 puntos de discretización igualmente espaciados en el intervalo [0.001, 1] en cada dimensión.

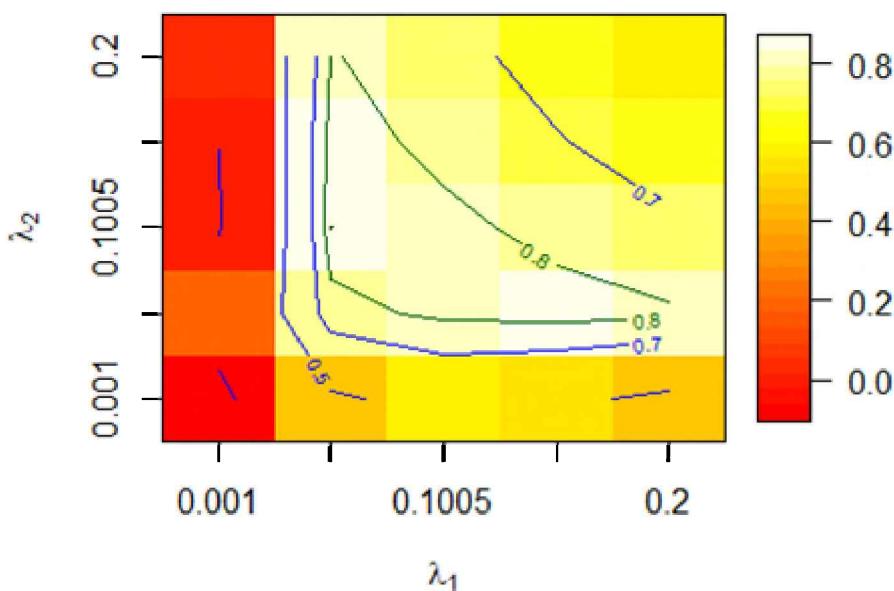
La experiencia puede guiar al usuario a refinar la cuadrícula de discretización, pero también se puede usar la función `estim.regul()` de forma recursiva. Primero, utilizando la cuadrícula predeterminada y ubicando un área donde se pueda alcanzar el valor óptimo para λ_1 y λ_2 y luego, determinando una nueva cuadrícula alrededor de estos primeros valores óptimos.

Utilizamos a continuación una rejilla de 5 x 5 para calcular el criterio CV. Con una de 50 x 50 los cálculos se hacen demasiado pesados.

```
res.regul <- estim.regul(X, Y, plt = TRUE, grid1 = seq(0.001, 0.2, 1=5), grid2 =
seq(0.001, 0.2, 1=5))
lambda1 = 0.05075
lambda2 = 0.1005
CV-score = 0.8502436
contour(res.regul$grid1, res.regul$grid2, res.regul$mat, add = TRUE, levels =
c(0,0.5,0.7), col = "blue")

contour(res.regul$grid1, res.regul$grid2, res.regul$mat, add = TRUE, levels =
c(0.8,0.85,0.88), col = "darkgreen")
```

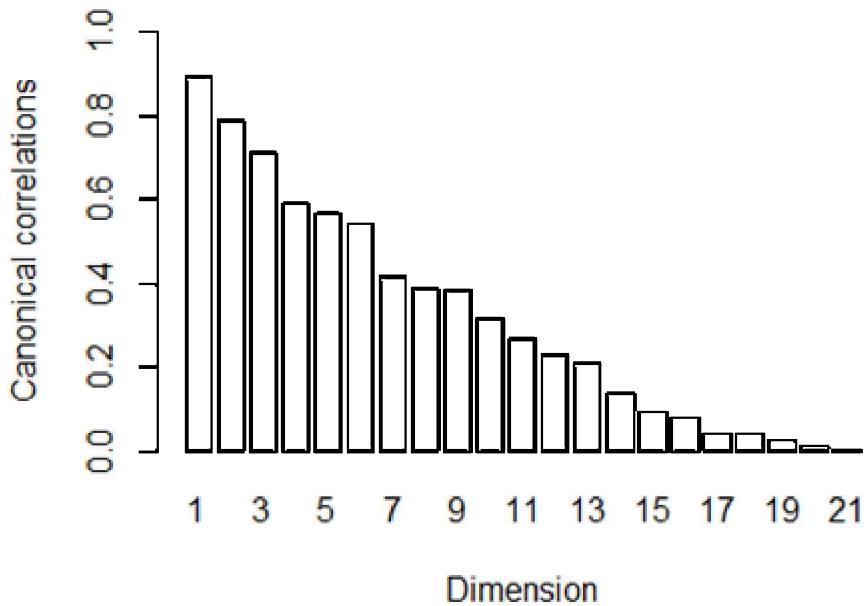
$$\text{CV}(\lambda_1, \lambda_2)$$



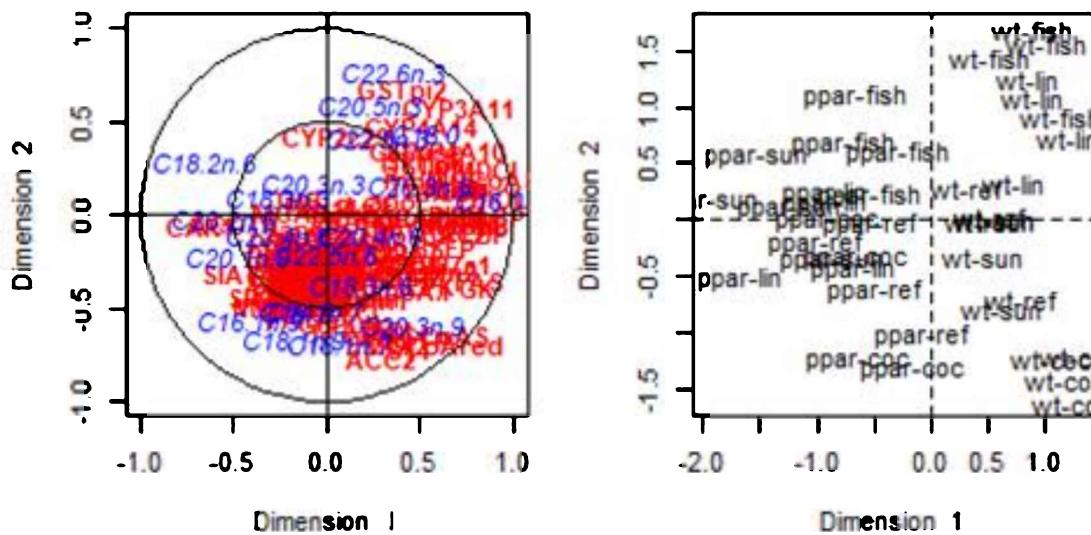
Una vez que los parámetros de regularización están fijados, se utilizará la función `rcc()` con los valores correspondientes para los parámetros λ_1 y λ_2 .

```
res.rcc <- rcc(X, Y, 0.05075, 0.1005)

barplot(res.rcc$cor, xlab = "Dimension", ylab = "Canonical correlations", names.arg =
1:21, ylim = c(0,1))
```



```
plt.cc(res.rcc, var.label = TRUE, ind.names = paste(nutrimouse$genotype,  
nutrimouse$diet, sep = "-"))
```



TERCER BLOQUE. INTRODUCCIÓN AL MACHINE LEARNING

TEMA 6: REGRESIÓN Y CLASIFICACIÓN: ÁRBOLES DE DECISIÓN Y REDES NEURONALES

ÍNDICE

1. Uso de Muestras para el Entrenamiento, Validación y Test

- 1.1. Muestras de entrenamiento, validación y test
- 1.2. Validación cruzada

2. Árboles de Decisión y Clasificación

- 2.1. Introducción
- 2.2. Aplicabilidad de los árboles de decisión para clasificación
- 2.3. Características de los algoritmos de clasificación
- 2.4. Árbol CHAID (CHi-square Automatic Interaction Detection) y CHAID exhaustivo
- 2.5. Árbol CRT (Classification and Regression Trees)
- 2.6. Árbol QUEST (Quick, Unbiased, Efficient Statistical Tree)
- 2.7. Árbol C5.0
- 2.8. Otros algoritmos de clasificación
- 2.9. Árboles de decisión con R

3. Redes Neuronales Artificiales

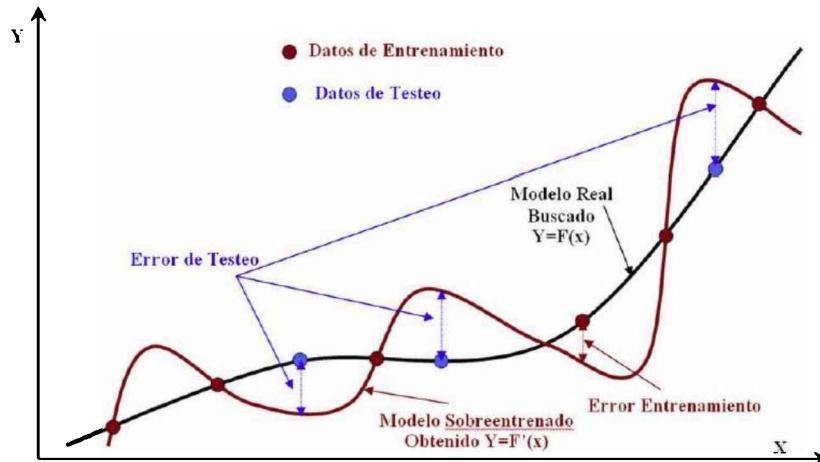
- 3.1. Introducción
- 3.2. Tipos de modelos de redes neuronales
- 3.3. Unidades de procesamiento de la información
- 3.4. Propiedades de los sistemas neuronales
- 3.5. Perceptrón multicapa
- 3.6. Funciones de base radial
- 3.7. Comparación entre las Funciones de Base Radial y el Perceptrón Multicapa
- 3.8. Análisis de sensibilidad e interpretación de los pesos de la red
- 3.9. Redes neuronales y modelos estadísticos clásicos
- 3.10. Otras arquitecturas de redes neuronales
- 3.11. Librería R Weka con redes neuronales

OBJETIVOS	PALABRAS CLAVE
<ul style="list-style-type: none"> · Validar modelos de Data Mining a través de la definición de muestras (entrenamiento, validación y test) o validación cruzada. · Construir modelos de regresión y clasificación a través de Árboles de Decisión. · Construir modelos de regresión y clasificación a través de Redes Neuronales Artificiales. 	<ul style="list-style-type: none"> · Árbol de decisión, ramas, nodos, poda. · Red neuronal, neuronas, capa oculta, pesos sinápticos, función de activación. - Tasa de aprendizaje, momento. - Perceptrón multicapa, función de base radial, análisis de sensibilidad

1. USO DE MUESTRAS PARA EL ENTRENAMIENTO, VALIDACIÓN Y TEST

En el proceso de estimación del modelo debemos de utilizar una estrategia para que nuestro método de clasificación/regresión se optimice. Es útil emplear un conjunto de datos de entrenamiento y otro conjunto de test.

FIGURA 1. ERRORES DE ENTRENAMIENTO Y DE TEST



Fuente: Grupo EDMANS. <http://edmans.webs.com/>.

1.1. MUESTRAS DE ENTRENAMIENTO, VALIDACIÓN Y TEST

Existen diferentes estrategias a la hora de utilizar el conjunto de datos disponible. Una práctica de muestreo, especialmente beneficiosa, es la partición (split) de la muestra en tres tablas de datos más pequeñas, que serán usadas con los siguientes fines: Training, Validación y Test.

La tabla de datos de training es utilizada para entrenar los modelos, es decir, para estimar los parámetros del modelo. Los datos de validación se emplean para ajustar y/o seleccionar el mejor modelo. En otras palabras, en base a algunos criterios, el modelo con el mejor valor del criterio establecido será seleccionado. Por ejemplo, el menor error cuadrático medio de la predicción es utilizado muy a menudo en modelos de regresión. La tabla de datos de test es utilizada para comprobar el comportamiento del modelo seleccionado. Después de que se haya seleccionado el mejor modelo y se haya comprobado puede utilizarse para puntuar (score) la base de datos completa.

Cada registro en la muestra puede aparecer solamente en una de las tres tablas. Cuando subdividimos la muestra de la tabla de datos, se puede utilizar un muestreo aleatorio simple, o un muestreo estratificado. Primero, se puede seleccionar aleatoriamente una pequeña fracción de los registros de una base de datos de tamaño muy grande, y en general, los modelos más simples requieren un menor número de elementos muestrales, mientras que los modelos más complejos requieren mayores tamaños de muestra. En segundo lugar, se puede usar el muestreo aleatorio de nuevo para subdividir la muestra, por ejemplo: 40% training, 30% validación, 30% test.

1.2. VALIDACIÓN CRUZADA

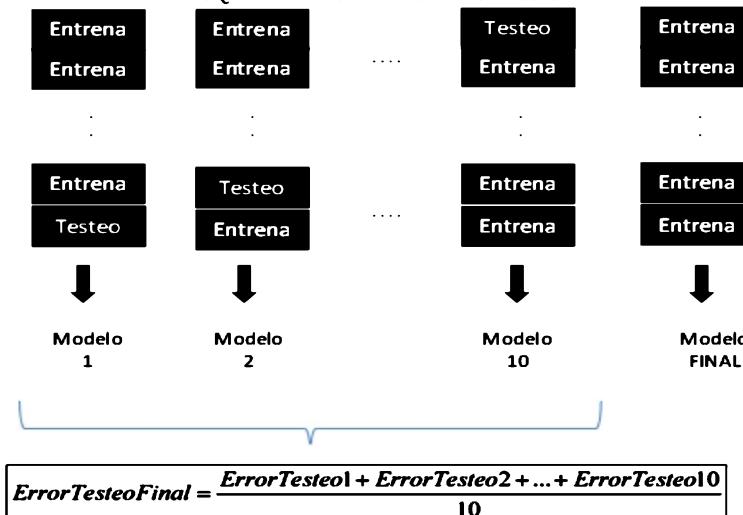
De manera ideal, siempre que elaboremos un modelo, lo óptimo es que lo entrenemos con un conjunto de entrenamiento y un conjunto de evaluación independientes que sean capaces de modelar la población original de manera individual.

Como esta situación en muchas ocasiones no es factible por falta de datos, o por la dificultad de obtener un muestreo adecuado, etcétera, se suele aplicar un esquema de validación denominado validación cruzada que consiste en dividir el conjunto de entrenamiento en k particiones, y repetir el procedimiento de entrenamiento y validación k veces, de forma que en cada una de ellas se entrene el modelo con $k-1$ particiones y se evalúe con la partición restante.

Los resultados finales se suelen obtener por agregación de los resultados originales.

Un esquema del desarrollo del proceso de cross validation se muestra en el gráfico siguiente:

FIGURA 2. ESQUEMA DE VALIDACIÓN CROSS VALIDATION



Fuente: Elaboración propia

CÓDIGO DE R CON LA LIBRERÍA CARET

```
library(mlbench)
library(caret)
data(BostonHousing)
str(BostonHousing)

set.seed(107)
inTrain <- createDataPartition(y = BostonHousing$medv, p = .75, list = FALSE)
train <- BostonHousing[inTrain,]
test <- BostonHousing[-inTrain,]
trainControl <- trainControl(method="repeatedcv", number=10, repeats=3,
classProbs=TRUE, summaryFunction=twoClassSummary)
```

En la función `trainControl` de la librería `caret` podemos especificar si queremos realizar el procedimiento a través de cross validation. El comando `summaryFunction=twoClassSummary` es un parámetro obligatorio si nos enfrentamos a un problema de clasificación y queremos sacar la curva ROC y el valor del área debajo de la curva.

2. ÁRBOLES DE DECISIÓN Y CLASIFICACIÓN

2.1. INTRODUCCIÓN

Tanto el problema de la clasificación o discriminación, como el de la regresión, se pueden abordar de varias formas. Desde el punto de vista estadístico se dispone de un amplio conjunto de elementos que pueden venir de dos o más poblaciones diferentes. Se observa un conjunto de características que vienen recogidas en una variable p-dimensional. El problema de clasificación se convierte en prever nuevos elementos, de acuerdo con la información disponible.

Los árboles de decisión o de clasificación no son modelos estadísticos basados en la estimación de los parámetros de una ecuación propuesta, no tenemos que estimar un modelo estadístico formal, son más bien algoritmos para clasificar utilizando particiones sucesivas, en general binarias, en los valores de una variable cada vez.

Esta técnica de clasificación es probablemente el modelo de clasificación más utilizado y popular, según Gehrke *et al.* (1999b) y Quinlan (1986). Existen algunas ventajas al utilizar esta técnica frente a otros modelos, Jiménez (2002). "Una de las ventajas más sobresalientes de los modelos de árbol de decisión es su carácter descriptivo, que permite entender e interpretar fácilmente las decisiones tomadas por el modelo, ya que tenemos acceso a las reglas que se utilizan en la tarea predictiva (aspecto no contemplado en otras técnicas, como las RNA). Además, los algoritmos utilizados para generar este tipo de modelos suelen incluir la opción de conversión de las rutas de decisión establecidas en el árbol a reglas lógicas del tipo 'si... entonces'. Con esta conversión se puede conseguir una mejor comprensión, si cabe, de las reglas predictivas del modelo."

Los árboles de decisión son particiones secuenciales de un conjunto de datos que maximizan las diferencias de la variable dependiente. Nos ofrecen una forma concisa de definir grupos que son consistentes en sus atributos pero que varían en términos de la variable dependiente. Esta herramienta puede emplearse tanto para la resolución de problemas de clasificación como de regresión: árboles de clasificación y árboles de regresión.

Desde otro punto de vista, podemos asegurar que los árboles de decisión o de clasificación son un modelo de predicción surgido en el ámbito del aprendizaje automático (Machine Learning) y de la Inteligencia artificial (Artificial Intelligence) que, partiendo de una base de datos, crea diagramas de construcciones lógicas que nos ayudan a resolver problemas.

Mediante esta técnica se representan de forma gráfica un conjunto de reglas sobre las decisiones que se deben de tener en cuenta para asignar un determinado elemento a una clase (valor de salida).

A esta técnica también se le denomina, siguiendo a Escobar (2007), segmentación jerárquica y se puede encuadrar, como técnica multivariante entre los métodos de dependencia, dado que como en el resto de los métodos estadísticos, se establece una distinción entre las variables que se pretenden explicar y aquellas otras que se utilizan para explicar las anteriores. La segmentación se realiza a través de un proceso (algoritmo) que está basado en criterios para identificar grupos homogéneos de una población.

La segmentación jerárquica es una técnica explicativa y descomposicional que utiliza un proceso de división secuencial, iterativo y descendiente que partiendo de una variable

dependiente que se pretende explicar, forma grupos homogéneos definidos específicamente mediante combinaciones de variables independientes en las que se incluyen la totalidad de los casos recogidos en la muestra.

Los modelos basados en árboles de clasificación suelen dar buenos resultados cuando muchas de las variables de clasificación son cualitativas. Sin embargo, algunos autores afirman que, en general, no son más eficaces que los procedimientos ofrecidos por la estadística clásica cuando las variables siguen distribuciones aproximadamente normales.

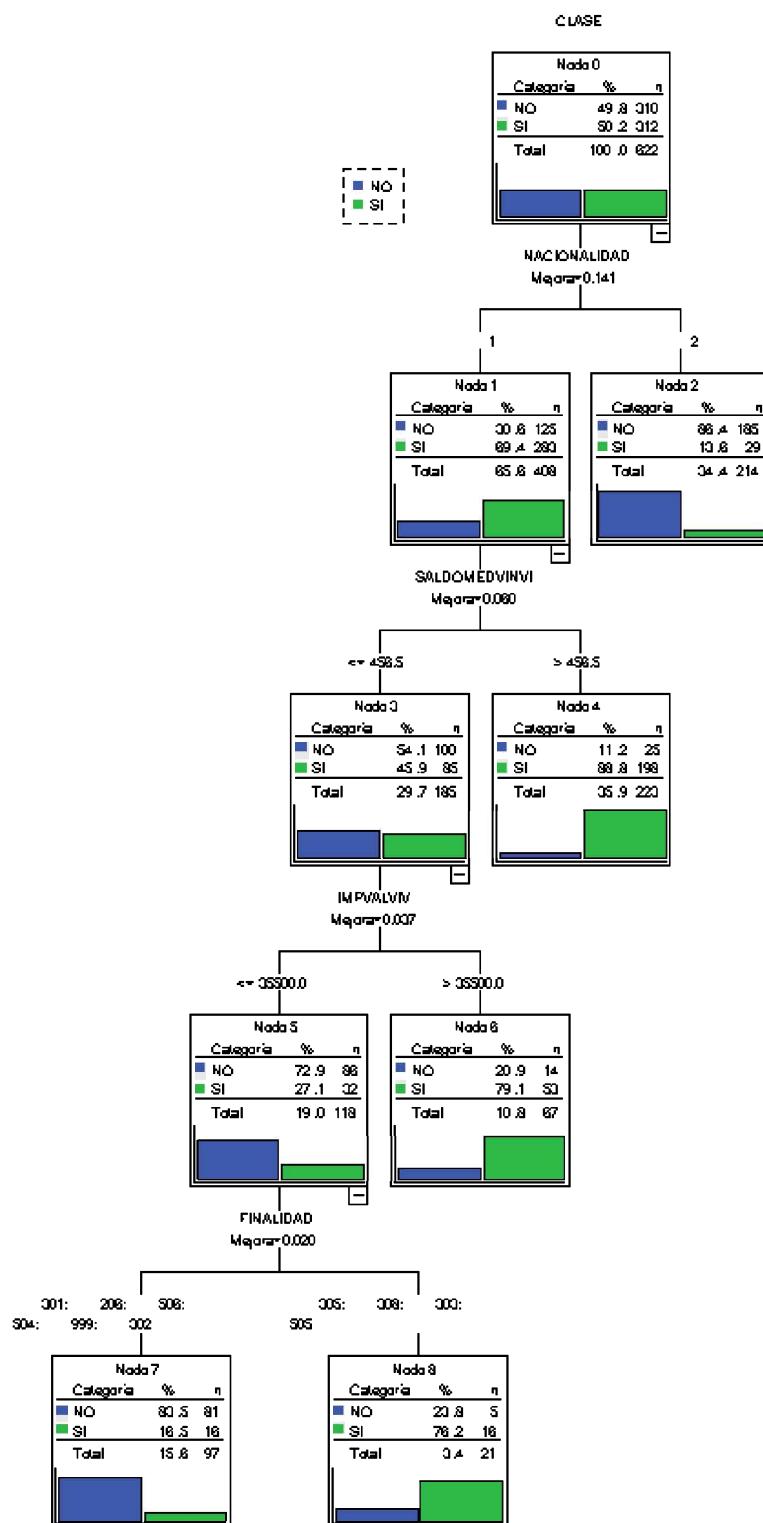
Diversos autores como Hernández *et al.* (2004) afirman que los árboles de decisión no solo son adecuados para resolver problemas de clasificación sino que abordan eficientemente otras tareas como la regresión, el agrupamiento, o la estimación de probabilidades.

En los árboles de decisión se encuentran los siguientes componentes: nodos, ramas y hojas. Los nodos son las variables de entrada, las ramas representan los posibles valores de la variable de entrada y las hojas son los posibles valores de la variable de salida. Como primer elemento de un árbol de decisión tenemos el llamado nodo raíz que va a representar a la variable de mayor relevancia en el proceso de clasificación.

En el gráfico siguiente se muestra una partición del árbol que genera el algoritmo CART y, como se observa en la Figura 1, se ven cuáles son las variables de segmentación más importantes a la hora de solicitar un crédito. El objetivo es identificar las variables que ofrecen la mejor escisión entre los peticionarios del crédito. La mejor división se produce con la variable nacionalidad con un 65,4% de individuos españoles que solicitan el préstamo frente a un 34,4% de extranjeros. A continuación, considerando la variable sexo, el algoritmo encuentra que la mejor variable para dividir es, en los de nacionalidad española, la variable relacionada con el saldo medio mantenido en la entidad. El importe del valor de la vivienda y la finalidad del crédito. Así el procedimiento continúa hasta que no existen variables independientes o no existen escisiones significativas pendientes de realizar.

Como se verá en el epígrafe siguiente hay diferentes algoritmos que pueden generar diversas estructuras de árbol de decisión.

FIGURA 1. EJEMPLO DE ÁRBOL DE CLASIFICACIÓN. MÉTODO CART



Fuente: Elaboración propia

2.2. APLICABILIDAD DE LOS ÁRBOLES DE DECISIÓN PARA CLASIFICACIÓN

En cuanto a la aplicabilidad de los árboles de decisión, en la construcción de árboles de clasificación se han desarrollado varios métodos y cada uno de ellos ofrece diferentes capacidades. En general, estos algoritmos son apropiados para problemas de clasificación que presenten las siguientes características:

- ✓ Cuando los ejemplos de aplicación vienen en forma de pares <atributos, valor>.
- ✓ Si la presentación de salida o función objetivo tiene valores discretos.
- ✓ Cuando es interesante el tipo de representación para la explotación posterior del modelo.
- ✓ Resulta conveniente si se necesitan descripciones disyuntivas.
- ✓ Los datos de aprendizaje pueden contener errores o valores nulos en algún atributo.

Esta técnica es la más utilizada por su sencillez. Los árboles de decisión ayudan a la toma de decisiones facilitando la interpretación de éstas, dado que resumen los ejemplos de partida permitiendo la clasificación de nuevos elementos siempre que no se alteren sustancialmente las condiciones iniciales. Los modelos de árboles de decisión, utilizados de forma exclusiva o en combinación con otras técnicas, se aplican a la resolución de numerosos problemas en el ámbito del marketing y, en general, son herramientas muy útiles en el control de la gestión empresarial, especialmente en las decisiones de segmentación de mercados, posicionamiento de productos y del comportamiento del consumidor, marketing directo, etcétera. Se ha empleado también en diagnósticos de enfermedades (clases) dependiendo de los síntomas (éstos representan en el modelo de árbol los atributos de entrada). También se utiliza para problemas de concesión de créditos, en la gestión de la relación con el cliente, CRM (Customer Relationship Management) y en otras múltiples actividades pertenecientes a las Ciencias Sociales en su conjunto.

2.3. CARACTERÍSTICAS DE LOS ALGORITMOS DE CLASIFICACIÓN

Los algoritmos que se encuentran, o bien solos o bien integrados en diferentes paquetes informáticos, son los que determinan o generan el procedimiento de cálculo que establece el orden de importancia de las variables en cada interacción. También se pueden imponer ciertas limitaciones en el número de ramas en que se divide cada nodo.

Los elementos y las herramientas de los algoritmos que determinan la construcción de un árbol son varios:

- ✓ El criterio para determinar la partición de cada nodo.
- ✓ La regla que declara un nodo terminal.
- ✓ La asignación de una clase a cada nodo terminal, lo que determina la regla de clasificación.
- ✓ Fusión: En relación a la variable dependiente, las categorías de las variables predictoras no significativas se agrupan juntas para formar categorías combinadas que sean significativas.

- ✓ Partición. Selección del punto de división. La variable utilizada para dividir el conjunto de todos los datos se elige por comparación con todas las demás.
- ✓ Poda. Se eliminan las ramas que añaden poco valor de predicción al árbol.
- ✓ La evaluación de la bondad del clasificador obtenido. La estimación de la validación del árbol y el cálculo del riesgo. Los métodos utilizados son los mismos, independientemente del método que se utilice para la generación del árbol.

2.3.1. Particiones posibles y criterios de selección

Lo más razonable para resolver el problema de la partición adecuada de un nodo es basarse en una tasa de error o coste de clasificación del nodo. El criterio del coste se determina a través de la denominada función de impureza, seleccionándose aquella partición que dé lugar al mayor decrecimiento de la impureza. Dependiendo de la función de impureza que se tome se tendrán distintos criterios de selección del corte óptimo. Los dos criterios más conocidos son el índice de Gini y el criterio de Twoing.

Cuando se ha definido el criterio de corte adecuado se obtiene, mediante particionamiento recursivo, sucesivas segmentaciones del conjunto de datos que cada vez se van haciendo más finas.

Una vez que ya se disponen de los criterios de partición y de asignación de cada una de las clases a cada nodo terminal el proceso terminará cuando se encuentre la regla que nos indique el instante en que el proceso de segmentación de los datos se detiene y se declara un nodo como terminal.

Los criterios de división están generalmente basados en medidas denominadas de impureza, entendiendo por impureza el grado en el que el nodo incluye casos de distintas clases. Así se define un nodo puro a aquel que solo contiene casos que pertenecen a una única clase. Se considera la bondad de una partición como la medida del decrecimiento de la impureza, y por tanto, la maximización de la bondad del ajuste es equivalente a la minimización de la impureza del árbol generado por la partición.

La función de impureza, dado un problema de clasificación con J clases diferentes, suele ser no negativa y se define sobre las J -duplas (p_1, p_2, \dots, p_J) donde cada valor representa la probabilidad de que un caso sea de la clase j en el subárbol actual.

La medida adaptada de Breiman *et al.* (1984) de impureza de un árbol T puede lograrse a través de las impurezas de sus hojas o nodos terminales (\tilde{T})

$$\phi(T) = \sum_{t \in \tilde{T}} p(t)\phi(t) \quad [1]$$

$P(t)$ representa la probabilidad de que un registro dado corresponda a la hoja t y $\phi(t)$ es la impureza del nodo terminal t .

Cualquier función ϕ tiene las siguientes propiedades:

1. Esta función ϕ posee un único máximo en $(1/J, 1/J, \dots, 1/J)$. Esto quiere decir que la impureza de un nodo es máxima cuando los registros correspondientes a cada una de las clases del problema es el mismo.
2. La función es simétrica respecto al conjunto de las J -duplas (p_1, p_2, \dots, p_J) .

3. Un nodo se denomina puro cuando solo contiene ejemplo de una clase (la función ϕ es igual a cero). En este caso, la función ϕ alcanza sus J mínimos en $\phi(1,0,\dots,0).....\phi(0,0,\dots,1)$.

2.3.2. Ganancia de información

Otras medidas intentan maximizar la ganancia de información que consigue el atributo A_i al ramificar el árbol de clasificación mediante la siguiente función I:

$$I(A_{ij}) = \sum_{j=1}^{M_i} p(A_{ij})H(C | A_{ij}) \quad [2]$$

La entropía es una medida de la incertidumbre que hay en un sistema, es decir, trata de medir ante una situación determinada la probabilidad de que ocurra cada uno de los posibles resultados. La entropía de clasificación se define como:

$$H(C_k | A_{ij}) = -\sum_{k=1}^J p(C_k | A_{ij}) \log_2 p(C_k | A_{ij}) \quad [3]$$

La ganancia de información que se produce al dividir T en los subconjuntos T_j viene dada por:

$$H(T) - \sum p(T_j)H(T) \quad [4]$$

donde $H(T)$ es la entropía de T .

2.3.3. El criterio de proporción de ganancia

Se trata de normalizar el concepto de ganancia obtenida, dado que este criterio posee el inconveniente de que favorece a los atributos o variables con muchos valores:

$$R(A_i) = \frac{H(C) - \sum_{j=1}^{M_i} p(A_{ij})H(C | A_{ij})}{\sum_{j=1}^{M_i} p(A_{ij}) \log_2(A_{ij})} \quad [5]$$

2.3.4. Índice de diversidad de Gini

El índice de Gini es una medida de diversidad de las clases en un nodo del árbol. Este índice se emplea en diferentes algoritmos de árboles de clasificación:

$$G(A_i) = \sum_{j=1}^{M_i} p(A_{ij})G(C | A_{ij}) \quad [6]$$

Siendo $G(C | A_{ij})$ igual a:

$$G(C | A_{ij}) = -\sum_{j=1}^{M_i} p(C_k | A_{ij})p(1 - p(C_k | A_{ij})) \quad [7]$$

- ✓ A_{ij} es el atributo empleado para ramificar el árbol,
- ✓ J es el número de clases,
- ✓ M_i es el de valores distintos que tiene el atributo A_i
- ✓ $p(A_{ij})$ constituye la probabilidad de que A_i tome su j -ésimo valor y
- ✓ $p(C_k | A_{ij})$ representa la probabilidad de que un ejemplo sea de la clase C_k cuando su atributo A_i toma su j -ésimo valor.

El índice de diversidad de Gini toma el valor cero cuando un grupo es completamente homogéneo y el mayor valor lo alcanza cuando todas las $p(A_{ij})$ son contantes, entonces el valor del índice es $(J-1)/J$.

2.3.5. Otros criterios de selección

Existen otras medidas utilizadas por algunos autores: López de Mantaras (1991) sugiere una alternativa al criterio de normalización de proporción de ganancia que evita la fragmentación del conjunto de entrenamiento característica de algunas reglas de decisión. La métrica de distancia que propone es la siguiente:

$$LM(A_i) = \frac{H(C) - \sum_{j=1}^{M_i} p(A_{ij})H(C | A_{ij})}{-\sum_{j=1}^{M_i} \sum_{k=1}^J \frac{n(C_k | A_{ij})}{N} \log_2 \frac{n(C_k | A_{ij})}{N}} \quad [8]$$

Otro trabajo que representa una alternativa al índice de Gini lo proponen Taylor y Silverman (1993) a cuya fórmula la llaman MPI (Mean Posterior Improvement):

$$MPI(A_i) = \prod_{j=1}^{M_i} p(A_{ij}) * \left(1 - \sum_{k=1}^J \frac{\prod_{j=1}^{M_i} p(C_k | A_{ij})}{P(C_k)} \right) \quad [9]$$

En Berzal *et al.* (2001) encontramos dos medidas que son menos complejas: el criterio MaxDif y el índice generalizado de Gini (GG). Ambas medidas realizan una suma ponderada de las medidas de impureza de cada uno de los subárboles resultantes de ramificar el nodo actual del árbol.

MaxDif

$$D(A_i) = \sum_{j=1}^{M_i} p(A_{ij})D(C | A_{ij}) \quad [10]$$

$$D(C | A_{ij}) = \max_k \{p(C_k | A_{ij}) - p(1 - p(C | A_{ij}))\} \quad [11]$$

Índice Generalizado de Gini

$$GG = \sum_{j=1}^{M_i} p(A_{ij})GG(C | A_{ij}) \quad [12]$$

$$GG(C | A_{ij}) = 1 - \max_k \{p(C_k | A_{ij})\} \quad [13]$$

Estos mismos autores también proponen la utilización de un umbral de soporte mínimo para mejorar el comportamiento de los algoritmos de árboles de clasificación TDIDT (Top Down Induction Decision Trees) clásicos en presencia de ruido que nos sirva para no tener en cuenta, en la construcción del árbol, ramas poco pobladas.

Existen otros criterios a los anteriormente expuestos basados en el criterio de la impureza de los nodos que se adscriben a otras categorías, Martin (1997): algunos de los criterios utilizan distancias o ángulos para ver las diferencias entre los diferentes subconjuntos, y otros criterios emplean medidas como la χ^2 de Pearson entre los conjuntos de entrenamiento y las clases.

Tanto en Martin (1997), como en Shih (1999), se pueden encontrar estudios exhaustivos sobre distintas reglas de división.

Es importante señalar que la mayor parte de las reglas de división que se han propuesto por los diferentes autores mejoran solo de forma marginal la precisión de los árboles que se construyen, pero tan solo en situaciones muy concretas.

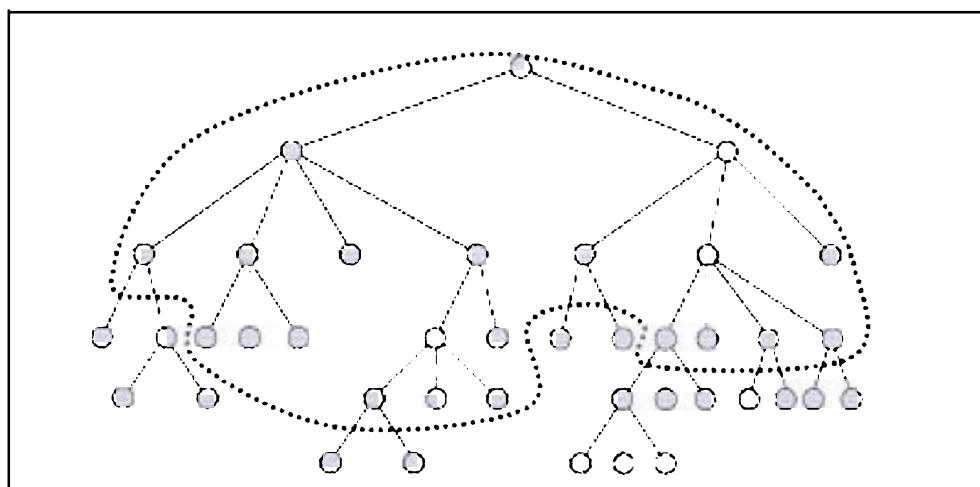
2.3.6. Poda en Árboles de clasificación

Todos los algoritmos de aprendizaje de árboles de clasificación obtienen modelos más o menos complejos y consistentes respecto a la evidencia: cubre todos los ejemplos y los cubre de una forma que puede parecer óptima.

Pero es demasiado ingenuo, porque el modelo es simplemente una aproximación al concepto de aprendizaje, y lo verdaderamente importante es que el modelo sirva para ejemplos nuevos, para clasificar bien al conjunto de test. Es especialmente importante si los datos contienen errores porque se ajustará el modelo a dichos errores y esto perjudicará a su comportamiento global, lo que se conoce como sobreajuste (overfitting), Hernández *et al.* (2004).

Para solucionar este problema es conveniente limitar el crecimiento del árbol modificando los algoritmos de aprendizaje de forma que se obtengan modelos más generales. Este proceso de poda se puede ver gráficamente en la siguiente ilustración:

FIGURA 2. EJEMPLO DE PODA. NODOS INFERIORES ELIMINADOS



Fuente: Hernández *et al.* (2004)

El concepto de poda en árboles de clasificación se puede dividir en dos métodos: prepoda y postpoda.

- ✓ **Prepoda.** Las reglas de parada se preguntan si merece la pena seguir o detener el proceso de crecimiento del árbol por la rama actual. Se denominan reglas de prepoda ya que reducen el crecimiento y la complejidad del árbol mientras se está construyendo, diferenciándose de las reglas de post poda que se utilizan cuando ya se ha construido el árbol.

Se pueden citar tres estrategias como reglas de prepoda:

- a. *Pureza del nodo.* Si el nodo solo contiene ejemplos o registros de una única clase se decide que la construcción del árbol ha finalizado. También se puede elegir un umbral de pureza y dejarlo para detener la realización del árbol de decisión.
 - b. *Cota de profundidad.* Previamente a la construcción se fija una cota que nos marque la profundidad del árbol que queremos. Cuando la alcanza se detiene el proceso.
 - c. *Umbral de soporte.* Podemos parar el proceso si especificamos un número de ejemplos mínimo para los nodos, ya que no consideramos fiables aquellos casos que no lleguen a alcanzar el valor fijado para el nodo.
- ✓ **Postpoda.** Normalmente se realiza la poda del árbol una vez que este ha sido construido. Son aquellas ramas del árbol con menor capacidad, las que suelen ser candidatas a ser podadas. Esta poda generalmente aumenta la capacidad de precisión del árbol. También hay que afirmar que la correcta estimación a priori del beneficio obtenido al simplificar el árbol durante su construcción resulta difícil y tan solo se ha empleado en algunos algoritmos recientes como el denominado PUBLIC, Rastogi y Shim (2000).

Existen dos formas de poda muy comunes utilizadas en los diferentes algoritmos: la poda por **coste-complejidad** y la **poda pesimista**.

- ✓ En la **poda por coste-complejidad** se trata de equilibrar la precisión y el tamaño del árbol. La complejidad está determinada por el número de hojas que posee el árbol (nodos terminales)

Siguiendo la notación anteriormente utilizada llamamos T al árbol de clasificación, N al número de ejemplos de entrenamiento y M al número de instancias que se clasifican mal, entonces la medida coste-complejidad del árbol T para un parámetro de complejidad especificado α toma la siguiente expresión:

$$R_\alpha(T) = R(T) + \alpha l(T) \quad [14]$$

Donde $R(T)$ es un estimador del error de $T = M/N$ (porcentaje de instancias mal clasificadas) y $l(T)$ es el número de hojas del árbol. El parámetro α es desconocido.

El árbol óptimo podado será aquel que haga mínima la expresión $R_\alpha(T)$

A la hora de trabajar se genera una secuencia de árboles con los distintos valores del parámetro desconocido α . Tal y como se describe en Breiman *et al.* (1984), al aumentar α se

tienden a podar menos nodos y de todos los árboles generados se escoge aquel que tenga asociado el menor error utilizando un conjunto independiente del de entrenamiento o el método de validación cruzada.

- ✓ **La poda pesimista** es utilizada por algunos algoritmos de construcción de árboles. Quinlan (1993) solo utiliza el conjunto de entrenamiento para construir el árbol. Con los casos clasificados incorrectamente (E) se saca su error de sustitución (E/N). El error de sustitución de un árbol es la suma de los errores de sus hojas. Aunque la probabilidad real del error cometido no se puede estimar exactamente, se puede asimilar utilizando una distribución de probabilidad binomial de errores y de éxitos en N experimentos. Dado un nivel de confianza se puede establecer un intervalo de confianza para esta supuesta distribución binomial y se puede asimilar el límite superior del intervalo como cota del error en el nodo.

Para llevar a cabo la poda pesimista se podaría el árbol si el intervalo de confianza del error de resustitución incluye el error de resustitución del nodo si se trata como una hoja. Procediendo de esta forma se eliminan los subárboles que no mejoren significativamente la precisión del clasificador. Esta heurística utilizada en este método suele producir buenos resultados.

A continuación, se realiza una breve descripción de los principales algoritmos más utilizados por los diferentes investigadores, y que podemos encontrar en la mayoría de los programas informáticos.

2.4. ÁRBOL CHAID (CHI-SQUARE AUTOMATIC INTERACTION DETECTION) Y CHAID EXHAUSTIVO.

El algoritmo en el que se basa el CHAID, el AID (Automatic Interaction Detection) o Detección Automática de Interacciones, fue uno de los más utilizados en la década de los años setenta y principios de los ochenta hasta que surgió el CHAID. Se le llama así porque la idea inicial no perseguía el objetivo de la clasificación, sino que estaban centrados en las interacciones entre las variables.

Las primeras ideas de la segmentación AID fueron recogidas por Morgan y Sonquist (1963) que propusieron la utilización recursiva del Análisis de la Varianza con todos los pares posibles de las variables candidatas.

Este algoritmo presenta dos limitaciones muy importantes, derivadas, por una parte, del elevado número de elementos muestrales que requieren para efectuar los análisis y, por otra, de la carencia de un modelo explícito que explique o determine la relación existente entre la variable dependiente y las variables explicativas.

En el algoritmo AID las variables explicativas han de estar medidas en escalas nominales u ordinales y la variable a explicar, variable criterio o dependiente, puede medirse en una escala métrica (medida con una escala proporcional o de intervalo) o ficticia (dicotómica con valores 0 y 1).

El análisis AID constituye un Análisis de la Varianza secuencial que se realiza mediante divisiones dicotómicas de la variable dependiente que busca en cada etapa la partición entre las categorías de la variable independiente que maximiza la varianza intergrupos y minimiza la varianza intragrupo.

La agrupación de categorías se efectúa probando todas las combinaciones binarias posibles de las variables. Es la prueba estadística F la que se utiliza para seleccionar las mayores diferencias posibles.

La media cuadrática externa (MCE) mide la heterogeneidad entre los grupos, es decir, aquellas muestras generadas con los pronosticadores, y toma la siguiente expresión:

$$MCE = \sum_{j=1}^J \frac{(\bar{x}_j - \bar{x})}{J-1} \quad [15]$$

La siguiente medida calcula la heterogeneidad dentro de cada muestra, es decir compara a cada individuo del grupo con la media del grupo:

$$MCI = \sum_{j=1}^J \sum_{i=1}^{n_j} \frac{(x_{ij} - \bar{x}_j)}{n-J} \quad [16]$$

El cociente entre ambas fórmulas sigue una distribución F de Snedecor que se distribuye con J-1 grados de libertad bajo la hipótesis nula $H_0 = \mu_1 = \mu_2 = \dots = \mu_J$

En este algoritmo, el proceso de subdivisión de la muestra en grupos dicotómicos continúa hasta que se verifica alguna de estas circunstancias:

- ✓ El tamaño de los grupos llega a un mínimo que se ha establecido de antemano.
- ✓ Las diferencias entre los valores medios de los grupos no son significativas, bien porque ninguna de las variables predictoras reduce significativamente la varianza residual, o bien porque los grupos son muy homogéneos y, por tanto, existe poca varianza intragrupo.

Las limitaciones de este algoritmo son importantes:

- ✓ Si se utilizan variables predictoras que difieren mucho en el número de categorías, el algoritmo tiende a seleccionar como más significativas y, por tanto, como más explicativas, aquellas variables que posean un número más elevado de categorías.
- ✓ Las particiones resultantes dependen de la variable que es elegida en primer lugar, lo que condiciona las sucesivas particiones.
- ✓ El carácter exclusivamente dicotómico de las particiones. Particiones con tres o más ramas reducen más la varianza residual y, además, pueden permitir una mejor selección de otras variables.

Muchas de las limitaciones del AID son corregidas por CHAID, que es un acrónimo de Chi-squared Automatic Interaction Detection (detector automático de interacciones mediante Ji cuadrado). Las ideas iniciales de Morgan y Sonquist (1963) fueron recogidas por otros autores que emplean, en lugar del Análisis de la Varianza, las tablas de contingencia y el estadístico χ^2 . Algunos de estos primeros pioneros en utilizar esta técnica son Cellard *et al.* (1967), Bourouche y Tennenhaus (1972), Kass (1980) y Madgison (1989). Aunque fue diseñado para trabajar solo con variables categóricas, posteriormente se incluyó la posibilidad de trabajar con variables categóricas nominales, categóricas ordinales y variables continuas, permitiendo generar tanto árboles de decisión para resolver problemas de clasificación como árboles de regresión.

En este algoritmo los nodos se pueden dividir en más de dos ramas. La construcción del árbol se basa en el cálculo de la significación de un contraste estadístico como criterio para definir la jerarquía de las variables predictoras o de salida, al igual que para establecer las agrupaciones de valores similares respecto a las variables de salida, a la vez que conserva inalterables todos los valores distintos. Todos los valores estadísticamente homogéneos son clasificados en una misma categoría y asignados a una única rama. Como medida estadística, si la prueba es continua, se utiliza la prueba F, mientras que si la variable predicha es categórica se utiliza la prueba Chi-cuadrado.

Para detectar si una relación es significativa, se utilizan varios métodos diferentes dependiendo del tipo de variables implicadas: variable dependiente nominal, ordinal, o de intervalo.

Para el caso de que la variable dependiente sea nominal disponemos de dos test estadísticos: el criterio de la χ^2 y la razón de verosimilitud (G^2).

Si cruzamos dos variables nominales, a una la llamamos X y a la otra Y, se construye la tabla de contingencia que estará formada por I filas (variable Y) y J columnas (Variable X) y en ella se encuentran las frecuencias conjuntas de ambas variables.

TABLA 1. NOTACIÓN DE UNA TABLA DE CONTINGENCIA.

	X ₁	X ₂	..	X _I	
Y ₁	n ₁₁	n ₁₂	..	n _{1I}	n _{1.}
Y ₂	n ₂₁	n ₂₂	..	n _{2I}	n _{2.}
:	:	:	..	:	:
Y _J	n _{J1}	n _{J2}	..	n _{JI}	n _{J.}
	n _{.1}	n _{.2}	..	n _{.I}	n

Fuente: Elaboración Propia

Las frecuencias marginales para cada uno de los valores j se obtienen a través del siguiente sumatorio:

$$n_{.j} = \sum_{i=1}^I n_{ij} \quad [17]$$

Igualmente se calculan las frecuencias marginales de los valores i con la fórmula siguiente:

$$n_{i.} = \sum_{j=1}^J n_{ij} \quad [18]$$

Si las categorías de la variable X y las categorías de la variable Y son independientes se cumple la siguiente condición:

$$P(I \cap J) = P(I) * P(J) \quad [19]$$

Las frecuencias esperadas debido a la independencia toman la siguiente expresión:

$$n_{ij}^* = n \frac{n_{i\cdot} n_{\cdot j}}{n} = \frac{n_{i\cdot} n_{\cdot j}}{n} \quad [20]$$

Una vez se hayan calculado las frecuencias empíricas y las teóricas disponemos de dos tests estadísticos muy similares. Definimos los residuos estandarizados a través de la siguiente expresión:

$$r_{ij}^s = \frac{n_{ij} - n_{ij}^*}{\sqrt{n_{ij}^*}} \quad [21]$$

El estadístico basado en la distribución de la χ^2 de Pearson adopta la siguiente expresión:

$$\chi^2 = \sum_{j=1}^J \sum_{i=1}^I \frac{(n_{ij} - n_{ij}^*)^2}{n_{ij}^*} \quad [22]$$

La otra medida es el estadístico razón de verosimilitud (G^2) que se fundamenta en el criterio de máxima verosimilitud, Haberman (1978) y Goodman (1979), que se calcula a través de la siguiente fórmula:

$$G^2 = 2 \sum_{j=1}^J \sum_{i=1}^I n_{ij} \ln \frac{n_{ij}}{n_{ij}^*} \quad [23]$$

Escobar (2007) afirma que, en el trabajo de comparación de modelos, el contraste a través de la G^2 ofrece ventajas adicionales. Aunque los resultados son muy similares, las ventajas se derivan de que la G^2 se calcula como la diferencia de las razones de verosimilitud entre dos modelos: el modelo saturado compuesto por efectos medios η de fila, de columna, y de asociación, frente al de independencia donde solo se consideran los efectos de fila y de columna.

$$n_{ij} = \eta \tau_i^A \tau_i^B \tau_i^{AB} \quad [24]$$

$$n_{ij}^* = \eta \tau_i^A \tau_i^B \quad [25]$$

Si la variable dependiente toma la forma ordinal se puede considerar un contraste diferente, donde solo se consideren los efectos columna, de acuerdo a los trabajos de Goodman (1979) y de Madgison (1992), ya que son los únicos efectos que representan a la variable ordinal. La expresión del modelo es la siguiente:

$$n_{ij} = \eta \tau_i^A \tau_i^B \delta_i^j \quad [26]$$

donde δ_i^j es un parámetro distinto para cada valor de la variable independiente.

Si se emplea este modelo, solo es adecuada la utilización del estadístico G^2 , que ahora toma la siguiente expresión:

$$G^2 = 2 \sum_{j=1}^J \sum_{i=1}^I n_{ij} \ln \frac{n_{ij}}{n_{ij}^*} \quad [27]$$

El algoritmo de segmentación de CHAID tiene tres fases: fusión, partición y detención.

- ✓ En la **fase de fusión**, cada predictor o variable independiente funde las categorías no significativas.
- ✓ En la **fase de partición**, para las variables independientes que tengan un valor p de Bonferroni ajustado significativo, hay que separar el grupo del predictor que tenga el menor valor p. Cada una de las categorías mezcladas del predictor se convierte en un nuevo subgrupo del grupo padre, si ningún predictor tuviese un valor significativo entonces no separar el grupo.
- ✓ La **fase de detención** se produce cuando se analizan todos los subgrupos o cuando contengan un número demasiado bajo de casos.

El ajuste de Bonferroni, Kass (1980) y Hawking y Kass (1982) establece que cuando se hagan B contrastes de significación, la significación total (p_T) debe ser menor o igual a la suma de cada una de las significaciones de los contrastes efectuados (p_i).

$$p_T \leq \sum_{i=1}^B p_i \quad [28]$$

El número de las posibles combinaciones de las pruebas de significación (B) se calcula a través de las fórmulas de la combinatoria. Escobar (2007) contempla tres posibilidades:

1. Si la opción escogida es sin restricciones, el número de pruebas para k grupos se determina con la siguiente fórmula:

$$B_n = \sum_{i=0}^{k-1} (-1)^i \frac{(k-i)}{i!(k-i)!} \quad [29]$$

2. Para variables dependientes ordinales, utilizando una función monótona, el número de pruebas para formar k grupos también depende del número de categorías c de la variable:

$$B_o = \binom{c-1}{k-1} \quad [30]$$

3. Si los casos perdidos se pueden fusionar con cualquier número de variables, los contrastes que se efectúan atienden a la siguiente expresión

$$B_{om} = \binom{c-1}{k-1} \frac{k-1+k(c-k)}{c-1} \quad [31]$$

Las ventajas del algoritmo CHAID se presentan a continuación:

- ✓ El método identifica aquellas clases o perfiles de las variables explicativas que no difieren desde el punto de vista estadístico respecto de la variable dependiente uniéndolas en el mismo nodo.
- ✓ El resultado no tiene que ser dicotómico dado que el algoritmo mantiene todas las categorías que son heterogéneas.
- ✓ El algoritmo posibilita la supresión de variables no significativas de forma segura.
- ✓ Permite conocer las variables que mantienen una fuerte interacción entre ellas.
- ✓ Cuando hay una fuerte correlación entre un grupo de variables predictoras, si se selecciona una de ellas las otras no serán consideradas. Esto supone la unión de variables desde el punto de vista de su impacto explicativo.

2.5. ÁRBOL CRT (CLASSIFICATION AND REGRESSION TREES)

El algoritmo CART es el acrónimo de Classification And Regression Trees (Árboles de Clasificación y de Regresión) fue diseñado por Breiman *et al.* (1984). Con este algoritmo, se generan árboles de decisión binarios, lo que quiere decir que cada nodo se divide en exactamente dos ramas.

Este modelo admite variables de entrada y de salida nominales, ordinales y continuas, por lo que se pueden resolver tanto problemas de clasificación como de regresión.

El algoritmo utiliza el índice de Gini para calcular la medida de impureza definido en la ecuación [50].

Para casos dicotómicos, se trata de ver cuán diferentes son las probabilidades de los valores de la variable dependiente en cada uno de los grupos generados por el procedimiento clasificador. Esta medida en Breiman *et al.* (1984) llamada **Índice Binario** toma la siguiente expresión:

$$\phi(s,t) = \frac{p_L p_R}{4} \left[\sum_{j=1}^J |p(j|t_L) - p(j|t_R)| \right]^2 \quad [32]$$

Si j solo toma dos valores, podemos contemplarlo como un promedio al cuadrado de las diferencias absolutas de los porcentajes que presentan los dos segmentos candidatos para dividirse, multiplicados por el producto de las proporciones de casos que se encuentran en cada uno de los segmentos. La expresión es la siguiente:

$$\phi(s,t) = \left[\sum_{j=1}^2 \frac{|p(j|t_L) - p(j|t_R)|}{2} \right]^2 p_L p_R \quad [33]$$

Si la variable es cuantitativa, lo que implica que estamos trabajando con árboles de regresión, se emplean las fórmulas propias del Análisis de la Varianza, similares a las utilizadas en el algoritmo AID.

Este cálculo de la varianza del nodo parental puede explicarse de la siguiente manera:

$$S^2(t) = \frac{\sum_{i=1}^{n(t)} (y_i - \bar{y}(t))^2}{n(t)} \quad [34]$$

Lo que interesa en este análisis, es estudiar si el predictor mejora la homogeneidad de los grupos que se forman tras la partición de la muestra en dos, y no el cálculo de la heterogeneidad en sí misma, para lo cual se resta a la varianza del nodo parental $S^2(t)$ las de los grupos filiales formados ($S^2(t_L)$ y $S^2(t_R)$), multiplicados por la proporción de casos que existen en las particiones (p_L y p_R):

$$\Phi(S, T) = S^2(T) - p_L S^2(t_L) - p_R S^2(t_R) \quad [35]$$

También se pueden emplear las desviaciones medias, cuyas fórmulas son las siguientes:

$$Dm(t) = \frac{\sum_{i=1}^{n(t)} |y_i - \bar{y}(t)|}{n(t)} \quad [36]$$

$$\Phi(S, T) = Dm(t) - p_L Dm(t_L) - p_R Dm(t_R) \quad [37]$$

2.6. ÁRBOL QUEST (QUICK, UNBIASED, EFFICIENT STATISTICAL TREE)

Este procedimiento denominado QUEST es el acrónimo de Quick, Unbiased, Efficient Statistical Tree (Árbol Estadístico Eficiente, Insesgado y Rápido). Este método fue propuesto por Loh y Shih (1997) que retoma las ideas previas contenidas en el trabajo de Loh y Vanichsetakul (1988) y le añaden diversas mejoras.

Este algoritmo trata de corregir y de restringir la exhaustiva búsqueda de particiones significativas que se generan tanto en los algoritmos AID y CHAID como en el CART.

Este método selecciona de forma previa la variable que segmenta mejor los datos, y después realiza la división óptima de ella.

Sintetizando el procedimiento, primero se elige la mejor variable predictora cuyo objetivo es que el número de categorías que poseen las variables no afecte a la elección de la mejor variable, para realizar después la mejor segmentación de la variable que ha seleccionado.

Este método QUEST solo puede ser utilizado si la variable de salida es categórica nominal.

Además de empezar el proceso de segmentación con la selección de variables, en vez de con la fusión de categorías, se procede después a la mejor división de los valores de la variable elegida. Otros cambios propuestos en este algoritmo es la eliminación de la poda, la transformación de las variables cualitativas en cuantitativas a través del procedimiento CRIMCOORD y un cambio en los valores perdidos de los clasificadores en los distintos nodos. Además, el algoritmo contiene la posibilidad de construir particiones no binarias como CHAID y, a semejanza del método CART, el rechazo a la validación cruzada propuesta por Breiman *et al.* (1984). Respecto a estos algoritmos, la diferencia está en la forma de particionar los nodos.

Los autores propusieron una clasificación arbórea basada en el análisis discriminante, a la que llamaron FACT (Fast Algorithm for Classification Trees). Así, una vez que se ha seleccionado la variable, se procede a ver cuál es la mejor partición binaria del nodo, donde nos podemos encontrar en alguno de los casos siguientes:

- ✓ Si la variable dependiente tiene J categorías y necesitamos reducirla a 2, se realiza a través de un procedimiento de conglomerados K-means de Hartigan y Wong (1979). Como centros de los conglomerados se escogen las medias muestrales de los pronosticadores más extremos y para cada media adicional se calcula la distancia cuadrática a los centros anteriormente elegidos y se agrupan al más cercano. Se vuelven a recalcular los centros y se vuelve a asignar un grupo dependiendo de la proximidad a los nuevos centros recalculados.
- ✓ Si la variable elegida es nominal hay que convertirla en un vector de variables ficticias empleando el análisis discriminante, que convierte cada valor discreto en otro continuo con valores entre -1 y +1. El valor asignado es la puntuación discriminante, calculada como se explica a continuación.

Suponemos que X es una variable categórica que toma los siguientes valores $\{c_1, c_2, \dots, c_M\}$. Como se ha indicado cada valor de X es transformado primero en una variable M-dimensional 0-1 que es un vector columna $v = (v_1, v_2, \dots, v_M)'$ donde todos los valores son ceros excepto el componente l -ésimo, el cual es igual a 1, donde l es definido implícitamente a través de $X = cl$.

$V_{i(j)}$ especifica la i -ésima observación de v en la j -ésima clase y define el M -dimensional vector columna:

$$\bar{v}^{(j)} = N_j^{-1} \sum_{i=1}^{N_j} v_i^{(j)}, \bar{v} = N^{-1} \sum_{i=1}^N v_i^{(j)} \quad [38]$$

Definimos las siguientes matrices de orden M x M:

$$B = \sum_{j=1}^J N_j (\bar{v}^{(j)} - \bar{v})(\bar{v}^{(j)} - \bar{v})' \quad [39]$$

$$W = \sum_{j=1}^J \sum_{i=1}^{N_j} (v_i^{(j)} - \bar{v}^{(j)})(v_i^{(j)} - \bar{v}^{(j)})' \quad [40]$$

$$T = \sum_{j=1}^J \sum_{i=1}^{N_j} (v_i^{(j)} - \bar{v})(v_i^{(j)} - \bar{v})' \quad [41]$$

Donde $T = B + W$

Se trata de hallar la proyección $a'v$ que maximice la suma de cuadrados de la razón *entre clases/intra clases*:

$$\frac{a' Ba}{a' Wa} \quad [42]$$

El valor a se corresponde con el autovector que está asociado con el mayor autovalor de la matriz $W'B$, siempre que la matriz inversa de W exista, Mardia *et al.* (1979)

Una vez que se ha dicotomizado la variable dependiente, si es el caso, y se han calculado las puntuaciones discriminantes se aplica ahora un análisis discriminante cuadrático para producir una división de la muestra por encima o por debajo de un valor calculado d . Tal y como se describe en Loh y Shih (1997) los pasos a seguir son los siguientes:

- ✓ Definimos a \bar{x}_A y S_A^2 como la media y la varianza de los elementos del grupo A y similarmente \bar{x}_B y S_B^2 representan la media y la varianza de la otra clase B.
- ✓ Sabemos que

$$p(A|t) = \sum_{j \in A} p(j|t) \text{ y } p(B|t) = 1 - p(A|t) \quad [43]$$

- ✓ Tomando logaritmos a ambos lados de la ecuación obtenemos:

$$p(A|t)S_A^{-1}\phi\{(x - \bar{x}_A)/S_A\} = p(B|t)S_B^{-1}\phi\{(x - \bar{x}_B)/S_B\} \quad [44]$$

Para encontrar la solución, el punto d que nos separe los grupos, es necesario resolver la ecuación de segundo grado $ax^2 + bx + c = 0$ donde los coeficientes toman las siguientes expresiones:

$$a = S_A^2 - S_B^2 \quad [45]$$

$$b = 2(\bar{x}_A S_A^2 - \bar{x}_B S_B^2) \quad [46]$$

$$c = (\bar{x}_B S_A)^2 - (\bar{x}_A S_B)^2 + 2S_A^2 S_B^2 \log\{p(A|t)S_B\}/\{p(B|t)S_A\} \quad [47]$$

Los diferentes casos que se pueden presentar son:

- a. Si $a = 0$ y $\bar{x}_A \neq \bar{x}_B$ solo existe una raíz dada por la siguiente expresión:

$$d = (\bar{x}_A + \bar{x}_B)/2 - (\bar{x}_A - \bar{x}_B)^{-1} S_A^2 \log\{p(A|t)\}/\{p(B|t)\} \quad [48]$$

- b. La ecuación de segundo grado no tiene solución si $a = 0$ y $\bar{x}_A = \bar{x}_B$

- c. Si $a \neq 0$ Entonces nos encontramos con dos posibilidades:

- Si el discriminante $b^2 - 4ac$ de la fórmula de resolución de la ecuación
$$d = \frac{-b \pm \sqrt{b^2 - 4ac}}{2}$$
es menor que cero $d = (\bar{x}_A + \bar{x}_B)/2$
- Si $b^2 - 4ac > 0$, que se verifica siempre que se cumple la igualdad $p(A|t) = p(B|t)$, obtenemos dos soluciones diferentes en la ecuación y escogemos aquella que esté más próxima a \bar{x}_A .

2.7. ÁRBOL C5.0

El algoritmo C5.0 y, sobre todo, su versión no comercial, C4.5, es uno de los algoritmos más utilizados en el ámbito de los árboles de clasificación.

La forma de inferir árboles de decisión a través de este algoritmo es el resultado de la evolución del algoritmo C4.5 (Quinlan, 1993) diseñado por el mismo autor y que a su vez es el núcleo del programa perteneciente a la versión ID3 (Quinlan, 1986).

Este algoritmo crea modelos de árbol de clasificación, permitiendo solo variables de salida categórica. Las variables de entrada pueden ser de naturaleza continua o categórica.

El algoritmo básico ID3 construye el árbol de decisión de manera descendente y empieza preguntándose, ¿qué atributo es el que debería ser colocado en la raíz del árbol? Para resolver esta cuestión cada atributo es evaluado a través de un test estadístico que determina cómo clasifica él solo los ejemplos de entrenamiento. Cuando se selecciona el mejor atributo éste es colocado en la raíz del árbol. Entonces una rama y su nodo se crean para cada valor posible del atributo en cuestión. Los ejemplos de entrenamiento son repartidos en los nodos descendentes de acuerdo al valor que tengan para el atributo de la raíz.

El proceso se repite con los ejemplos, para seleccionar un atributo que será ahora colocado en cada uno de los nodos generados. Generalmente el algoritmo se detiene cuando los ejemplos de entrenamiento comparten el mismo valor para el atributo que está siendo probado. Sin embargo, es posible utilizar otros criterios para finalizar la búsqueda:

- ✓ Cobertura mínima de tal forma que el número de ejemplos por cada nodo está por debajo de cierto umbral.
- ✓ Pruebas estadísticas para probar si las distribuciones de las clases en los subárboles difieren significativamente.

Una de las maneras de cuantificar la bondad de un atributo consiste en considerar la cantidad e información que proveerá ese atributo tal y como está definido en la teoría de la información. Por tanto, este algoritmo está basado en el concepto de “ganancia de información”. El C4.5 modifica el criterio de selección del atributo empleando en lugar de la ganancia, la razón de ganancia. Para definir este concepto necesitamos definir el concepto de entropía.

Si el conjunto de los registros de la base de datos T se agrupan en función de las categorías de la variable de salida S, obteniéndose una proporción p_k para cada grupo asociado a un posible resultado o categoría, la función de entropía, en el caso de dos atributos de salida, con probabilidades p, y su complementaria, 1-p, y de acuerdo a la ecuación [3] toma la siguiente expresión:

$$INFO(T) = p * \log_2(p) + (1 - p) \log_2(1 - p) \quad [49]$$

Ahora, se puede expresar la ganancia de información teniendo en cuenta una variable de entrada, según ecuación [48]

$$GANANCIA(X, T) = INFO(T) - INFO(X, T) \quad [50]$$

donde

$$INFO(X, T) = \sum_{i=1}^k \frac{T_i}{T} * INFO(T_i) \quad [51]$$

$INFO(X, T)$ nos proporciona la información aportada por la variable de salida S cuando se tiene en cuenta una variable de entrada X.

$INFO(X, T_i)$ es la entropía de la variable de salida S en cada subconjunto T_i determinado por las k categorías de la variable de entrada X. T_i es el número de registros asociados a una categoría i de la variable X.

El concepto de ganancia representa la diferencia necesitada para identificar la categoría destino asociada a un elemento T y la información necesitada para identificar dicha categoría cuando se conoce el valor de una variable de entrada para ese mismo elemento. Lo que esto significa es que esa variable mostrará menor incertidumbre a la hora de la clasificación que el resto de variables de entrada.

En el ejemplo, la variable NACIONALIDAD es la que menor incertidumbre presenta o la que tiene mayor ganancia de información, por lo que será la variable que constituirá el nodo raíz.

La ganancia de información posee el inconveniente de que favorece a los atributos o variables con muchos valores, por lo que este algoritmo calcula la medida siguiente:

$$GAINRATIO(X, T) = \frac{GANANCIA(X, T)}{SPLITINFO(X, T)} \quad [52]$$

donde,

$$SPLITINFO(X, T) = -\sum \frac{T_i}{T} * \log_2 \left(\frac{T_i}{T} \right) \quad [53]$$

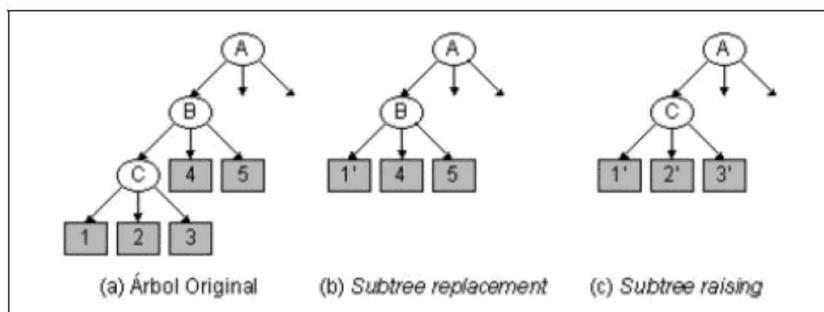
$SPLITINFO(X, T)$ es la información aportada por la división (split) del conjunto de registros T a partir de los valores de la variable de entrada.

Este proceso se iterá para cada una de las ramas descendientes ciñéndonos únicamente al total de registros asociados a cada rama y con las variables de entrada distintas a las utilizadas en el nodo raíz. El proceso para una rama concreta termina cuando todos los registros de esa rama quedan perfectamente clasificados en una de las categorías de la variable de salida.

Una de las incorporaciones más novedosas de este algoritmo es la inclusión de la técnica "boosting" para la generación y combinación de múltiples modelos de clasificación. Otras interesantes aportaciones realizadas en el algoritmo C5 son que permite aplicar diferentes costes a los errores de clasificación, que se admiten ahora formatos nuevos de datos, por ejemplo, fechas, horas, etc. y que se ha añadido la posibilidad de suprimir ciertos atributos marginales antes de construir el árbol para así poder reducir la dimensionalidad de la base de datos.

Los tres algoritmos más empleados por parte de la comunidad científica son el C4.5, CART y CHAID.

FIGURA 3. TIPOS DE OPERACIONES DE PODA EN C.4.5.



Fuente: Molina y García (2006)

2.8. OTROS ALGORITMOS DE CLASIFICACIÓN

2.8.1. Algoritmo de construcción de árboles consolidados

El algoritmo CTC (Construcción de árboles consolidados), Pérez (2006), para construir el árbol consolidado, se basa en técnicas de remuestreo.

Primero el algoritmo genera un conjunto de muestras, posteriormente lo que hace es que en cada nodo va construyendo un árbol C4.5 asociado a cada muestra. Por medio de un consenso entre una serie de submuestras, se elige la variable más prometedora por la que hay que dividir ese nodo, la variable consolidada.

Es decir, a cada muestra se le va a realizar un proceso por el cual decidirá cuál es la variable por la que esa muestra quiere dividir. Ese proceso está basado en un árbol de clasificación estándar como es el C4.5 (J48 en WEKA). Posteriormente, teniendo todas las variables por las que las submuestras quieren dividir, se realizará una votación entre todas las variables “candidatas” y se elegirá la variable más votada, variable consolidada. Tras elegirla, todas las submuestras se dividirán obligatoriamente por esa variable consolidada, y tras este paso, diremos que el nodo ha sido consolidado.

El proceso acabará cuando todos los nodos del árbol hayan sido procesados.

2.8.2. Random Forest

La clasificación Random Forest emplea el algoritmo descrito por Breiman (2001). Este algoritmo está basado en la combinación de árboles de decisión independientes generados a partir de un vector de muestreo aleatorio que usa la misma distribución para todos los árboles de estudio. El término Random Forest se toma de la primera propuesta realizada por Ho (1995)

Este algoritmo está considerado como un clasificador bastante preciso. Trabaja bien, aunque haya datos perdidos, y ofrece un método para la interacción de las variables.

2.8.3. Decision Stump

Es un algoritmo muy sencillo que genera un árbol de decisión de un único nivel, con una única división (con una rama adicional para valores no definidos). Utiliza solo un atributo

para construir el árbol de decisión. Para la selección de este atributo el algoritmo se basa en el criterio de la ganancia de información.

Funciona de forma aceptable en problemas de dos clases. No obstante, para problemas de más de dos clases las tasas de error son muy elevadas. Aunque su rendimiento en general es más bajo que otros árboles es utilizado, por la propia concepción de la metodología, en los procedimientos de Gradient Boosting.

Admite tanto atributos numéricos como categóricos, y deben tenerse en cuenta diversas posibilidades cuando se calcula la ganancia de información, Molina y García (2006).

Finalmente, en la Tabla 2 se ofrece una comparación entre los principales algoritmos clásicos de los árboles de clasificación, mostrando algunas de sus características más relevantes:

TABLA 2. CARACTERÍSTICAS DE LOS PRINCIPALES ALGORITMOS DE ÁRBOLES DE DECISIÓN						
Algoritmo	Variables predictoras	Tipo de división	Criterio de División	Casos missing	Método de Poda	Implementación
CART (1984)	Continuas/ Discretas	Binaria	Impureza (<i>Gini index</i>)	SI	Post-	Libre Comercial
ID3 (1979)	Discretas	<i>n</i> -aria	Ganancia de información (Entropía)	NO	NO	Comercial
C4.5 (1993)	Continuas/ Discretas	Binaria/ <i>n</i> -aria	<i>Gain ratio</i> (Entropía)	SI	Pre-/Post-	Libre Comercial
J4.8	Continuas/ Discretas	Binaria/ <i>n</i> -aria	<i>Gain ratio</i> (Entropía)	SI	Pre-/Post-	Libre (Weka)
C5.0	Continuas/ Discretas	Binaria/ <i>n</i> -aria	<i>Gain ratio</i> (Entropía)	SI	Pre-/Post-	Comercial
CHAID (1975)	Discretas	<i>n</i> -aria	χ^2	SI	Pre- (nivel de significancia)	Comercial

Fuente: Pérez (2006)

2.9. ÁRBOLES DE DECISIÓN CON R

En la programación en R de este epígrafe se han utilizado diferentes librerías originales que cubren diferentes modelos de árboles de regresión y clasificación. Podemos sacar los resultados y los gráficos con las librerías originales o bien a través de la librería caret y Rweka. Caret ofrece muchas ventajas entre las que se encuentra la forma fácil con la que se pueden realizar simulaciones con los valores diferentes que se emplean en muchos algoritmos. Por su parte RWeka nos permite incluir en R los diferentes algoritmos del programa WEKA y visualizar los resultados de la misma manera que se ofrecen en el programa original.

A continuación, se presenta el código en R y los resultados más significativos.

```
# Instalación librería CHAID
install.packages("CHAID", repos="http://R-Forge.R-project.org")

# Librerías que se emplean en este tema
library(party)
library(rpart)
library(randomForest)
library(C50)
```

```

library(CHAID)
library(caret)
library(RWeka)

# Preparamos las muestras de entrenamiento y de test.
set.seed(123)
train_sample <- sample(1000,900)
train <- German[train_sample,]
test <- German[-train_sample,]

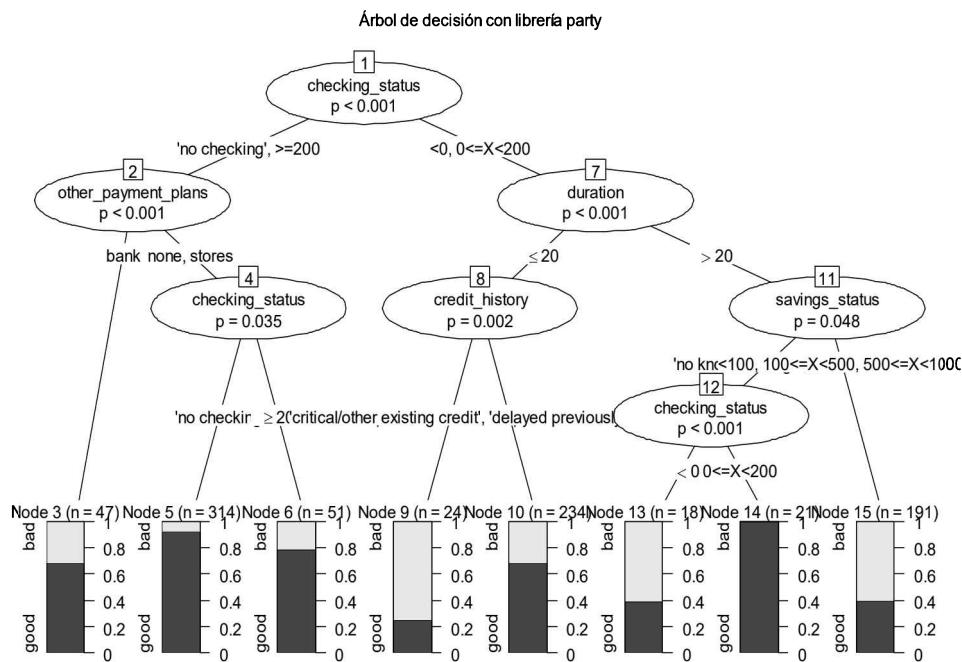
```

2.9.1. Conditional Inference Tree

```

library(party)
german_formula <- class ~ .
german_ctree <- ctree(german_formula, data=train)
plot(german_ctree, main= 'Árbol de decisión con librería party')

```



```

#Predicciones con test
pred1 <- predict(german_ctree, newdata = test)

# Se puede sacar la tabla de confusión de diferentes maneras
table(pred1, test$class)
  pred1  bad  good
    bad   13     7
    good   20    60

library(gmodels)
CrossTable(pred1, test$class, prop.chisq=FALSE, prop.c=FALSE, prop.r=TRUE)

```

Cell Contents	
N	
N / Row Total	
N / Table Total	

Total Observations in Table: 100

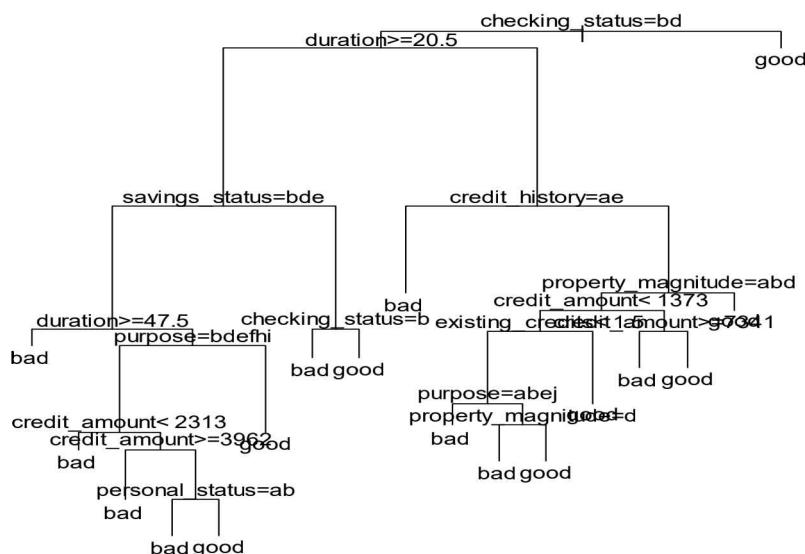
| test\$class

pred1	bad	good	Row Total
bad	13 0.650 0.130	7 0.350 0.070	20 0.200
good	20 0.250 0.200	60 0.750 0.600	80 0.800
Column Total	33	67	100

2.9.2. Recursive Partitioning and Regression Trees

```
library(rpart)
fit1 <- rpart(class~, data=train)
plot(fit1, main= 'Árbol de decisión con librería rpart' );text(fit1);
```

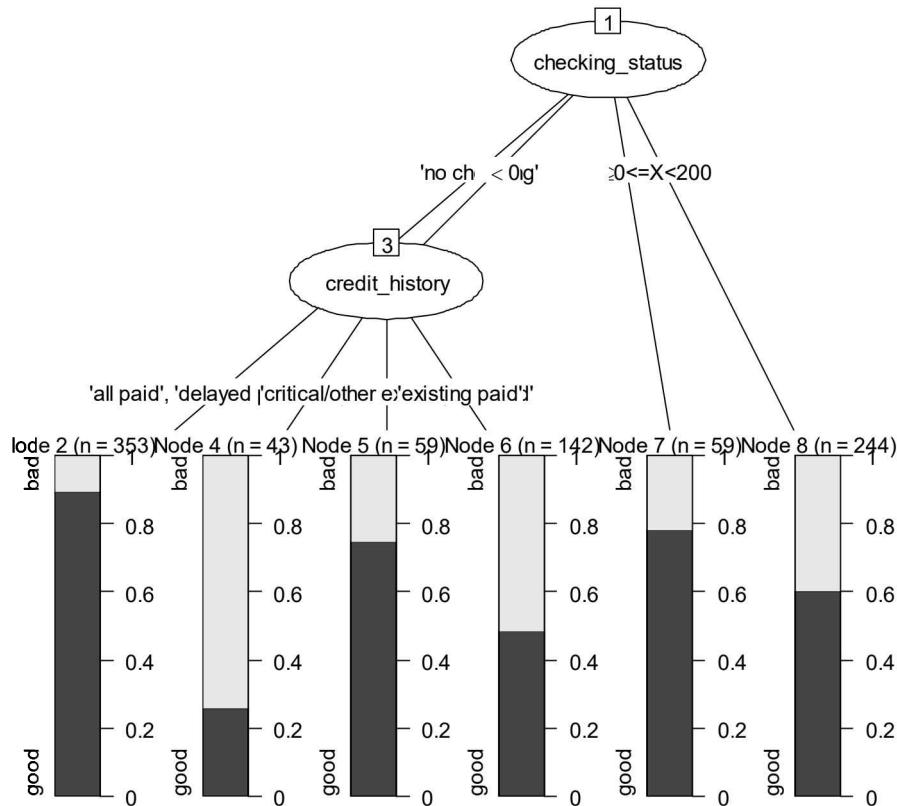
Árbol de decisión con librería rpart



```
# Clase y probabilidad predicha
test$t <- predict(fit1,type='class',test)
pred_prob <- predict(fit1,type='prob',test)
```

2.9.3. CHAID (Chi-square Automatic Interaction Detection)

```
library(CHAID)
chaid <- chaid(class ~ checking_status + credit_history + job, control =
  chaid_control(minprob = 0.001, minsplit = 50, minbucket = 20), data=train)
plot(chaid, uniform = T, compress = T, margin = 0.2, branch = 0.3)
```



2.9.4. Árbol C5.0 de Quinlan

```

library(C50)
german_model <- C5.0(x = train[,-21], y = train$class)
summary(german_model)

Call:
C5.0.default(x = train[, -21], y = train$class)

C5.0 [Release 2.07 GPL Edition]      Mon Mar 05 22:04:21 2018
-----
Class specified by attribute `outcome'

Read 900 cases (21 attributes) from undefined.data

Decision tree:

checking_status in {'no checking',>=200}: good (412/50)
checking_status in {<0,0<=X<200}:
...other_parties = guarantor:
  ...duration > 36: bad (4/1)
  : duration <= 36:
    : ...other_payment_plans in {none,stores}: good (24)
    :   other_payment_plans = bank:
      :     ....purpose = 'new car': bad (3)
      :       purpose in {'domestic appliance','used car',business,education,
      :                     furniture/equipment,other,radio/tv,repairs,
      :                     retraining}: good (7/1)
    other_parties in {'co applicant',none}:
      ....credit_history = 'critical/other existing credit': good (102/30)
      credit_history = 'no credits/all paid': bad (27/6)
      credit_history = 'all paid':
        ....other_parties = 'co applicant': good (2)
        :   other_parties = none: bad (26/8)
  
```

```

credit_history in {'delayed previously','existing paid'}:
:....savings_status in {>=1000,500<=X<1000}: good (19/3)
    savings_status = 100<=X<500:
        ....other_parties = 'co applicant': bad (3)
        : other_parties = none:
            ....personal_status in {'male div/sep',
            : 'male mar/wid'}: bad (6/1)
            personal_status = 'female div/dep/mar':
            ....installment_commitment <= 3: good (4/1)
            : installment_commitment > 3: bad (4)
            personal_status = 'male single':
            ....age <= 41: good (15/2)
            : age > 41: bad (2)
        savings_status = 'no known savings':
        ....credit_history = 'delayed previously': good (8)
        credit_history = 'existing paid':
        ....foreign_worker = no: good (2)
        foreign_worker = yes:
        ....checking_status = <0:
            ....own_telephone = none: bad (11/2)
            : own_telephone = yes:
            ....credit_amount <= 5045: good (5/1)
            : credit_amount > 5045: bad (2)
            checking_status = 0<=X<200:
            ....residence_since > 3: good (9)
            : residence_since <= 3: [S1]
        savings_status = <100:
        ....duration > 39:
            ....residence_since <= 1: good (2)
            : residence_since > 1: bad (19/1)
        duration <= 39:
            ....purpose in {'domestic appliance',other}: good (3)
            purpose in {'new car',retraining}: bad (47/16)
            purpose = 'used car':
            ....credit_amount <= 8086: good (9/1)
            : credit_amount > 8086: bad (5)
            purpose = education:
            ....checking_status = <0: bad (5)
            : checking_status = 0<=X<200: good (2)
            purpose = repairs:
            ....residence_since <= 3: bad (4/1)
            : residence_since > 3: good (3)
            purpose = business:
            ....credit_history = 'delayed previously': bad (2)
            credit_history = 'existing paid':
            ....age <= 34: good (5)
            : age > 34: bad (2)
            purpose = radio/tv:
            ....employment in {<1,unemployed}: bad (14/5)
            employment = 4<=X<7: good (3)
            employment = >=7:
            ....credit_amount <= 932: bad (2)
            : credit_amount > 932: good (7)
            employment = 1<=X<4:
            ....duration <= 15: good (6)
            : duration > 15:
            ....credit_amount <= 3275: bad (7)
            : credit_amount > 3275: good (2)
            purpose = furniture/equipment:
            ....residence_since <= 1: good (8/1)
            residence_since > 1:
            ....other_payment_plans = bank: bad (2/1)
            other_payment_plans = stores: good (1)
            other_payment_plans = none:
            ....own_telephone = yes: bad (7/1)
            own_telephone = none:

```

```
:...duration > 27: bad (3)
duration <= 27: [S2]
```

SubTree [S1]

```
property_magnitude in {'life insurance','no known property'}: bad (4)
property_magnitude = car: good (6)
property_magnitude = 'real estate':
:...job = 'unemp/unskilled non res': good (2)
    job in {'high qualif/self emp/mgmt','unskilled resident',skilled}: bad (2)
```

SubTree [S2]

```
checking_status = 0<=X<200: bad (5/2)
checking_status = <0:
:...property_magnitude in {'life insurance','no known property',
:                           'real estate'}: good (8)
property_magnitude = car:
:...installment_commitment <= 1: good (2)
    installment_commitment > 1: bad (4)
```

Evaluation on training data (900 cases):

Decision Tree		
Size	Errors	
55	135(15.0%)	<<
(a)	(b)	<-classified as
177	90	(a): class bad
45	588	(b): class good

Attribute usage:

```
100.00% checking_status
54.22% other_parties
50.00% credit_history
32.56% savings_status
25.22% duration
19.78% purpose
10.11% residence_since
7.33% other_payment_plans
5.22% own_telephone
4.78% foreign_worker
4.56% employment
4.33% credit_amount
3.44% personal_status
3.11% property_magnitude
2.67% age
1.56% installment_commitment
0.44% job
```

Time: 0.0 secs

```
# Importancia de las variables
C5imp(german_model)
          Overall
checking_status      100.00
other_parties        54.22
credit_history        50.00
savings_status        32.56
duration              25.22
purpose                19.78
```

```

residence_since      10.11
other_payment_plans 7.33
own_telephone        5.22
foreign_worker        4.78
employment           4.56
credit_amount         4.33
personal_status       3.44
property_magnitude   3.11
age                   2.67
installment_commitment 1.56
job                  0.44
housing               0.00
existing_credits     0.00
num_dependents       0.00

C5imp(german_model, metric = "splits")
    Overall
  checking_status      10.256410
  credit_amount         10.256410
  duration              10.256410
  residence_since       10.256410
  credit_history         7.692308
  other_parties          7.692308
  age                   5.128205
  installment_commitment 5.128205
  other_payment_plans   5.128205
  own_telephone          5.128205
  property_magnitude    5.128205
  purpose                5.128205
  employment             2.564103
  foreign_worker          2.564103
  job                   2.564103
  personal_status         2.564103
  savings_status          2.564103
  housing                 0.000000
  existing_credits        0.000000
  num_dependents          0.000000

```

```

# Predicciones y tabla de clasificación
german_predicciones <- predict(german_model, test, type = "class")
CrossTable(test$class, german_predicciones, prop.chisq=FALSE, prop.c=FALSE,
prop.r=TRUE)

```

Cell Contents		

N		

N / Row Total		
N / Table Total		

Total Observations in Table: 100

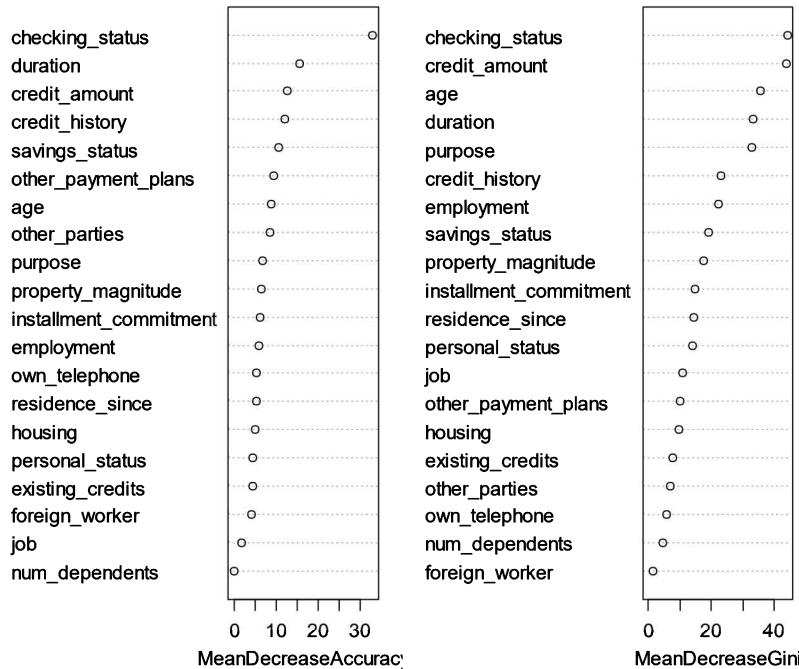
test\$class	german_predicciones		
	bad	good	Row Total
bad	14	19	33
	0.424	0.576	0.330
	0.140	0.190	
good	8	59	67
	0.119	0.881	0.670
	0.080	0.590	
Column Total	22	78	100

2.9.5. Random Forest

```
library(randomForest)
arf <- randomForest(class~., data=train, importance=TRUE, proximity=TRUE, ntree=500,
keep.forest=TRUE)

# Gráfico de la importancia de las variables
varImpPlot(arf, main="Importancia de las variables con Random Forest")
```

Importancia de las variables con Random Forest



2.9.6. Árboles con caret (ejemplo de Random Forest)

```
# Muestras de entrenamiento y test
set.seed(107)
inTrain <- createDataPartition(y = German$class, p = .8, list = FALSE)
train <- German[inTrain,]
test <- German[-inTrain,]

# Árbol
rf1 <- train(class ~ ., data = train, method = "rf", trControl = trainControl(method
= "repeatedcv", repeats = 5))
rf1
Random Forest

800 samples
20 predictor
2 classes: 'bad', 'good'

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 5 times)
Summary of sample sizes: 720, 720, 720, 720, 720, 720, ...
Resampling results across tuning parameters:

  mtry  Accuracy   Kappa
  2     0.71975   0.1051813
  25    0.73950   0.3006740
  48    0.73175   0.2909812
```

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was mtry = 25.

2.9.7. Árboles con Rweka

```
library(RWeka)
WOW(J48) # Parámetros de la función J48
-U      Use unpruned tree.
-O      Do not collapse tree.
-C <pruning confidence>
      Set confidence threshold for pruning. (default 0.25)
Number of arguments: 1.
-M <minimum number of instances>
      Set minimum number of instances per leaf. (default 2)
Number of arguments: 1.
-R      Use reduced error pruning.
-N <number of folds>
      Set number of folds for reduced error pruning. One fold is
      used as pruning set. (default 3)
Number of arguments: 1.
-B      Use binary splits only.
-S      Do not perform subtree raising.
-L      Do not clean up after the tree has been built.
-A      Laplace smoothing for predicted probabilities.
-J      Do not use MDL correction for info gain on numeric
      attributes.
-Q <seed>
      Seed for random data shuffling (default 1).
Number of arguments: 1.
-doNotMakeSplitPointActualValue
      Do not make split point actual value.
-output-debug-info
      If set, classifier is run in debug mode and may output
      additional info to the console
-do-not-check-capabilities
      If set, classifier capabilities are not checked before
      classifier is built (use with caution).
-num-decimal-places
      The number of decimal places for the output of numbers in
      the model (default 2).
Number of arguments: 1.
-batch-size
      The desired batch size for batch prediction (default 100).
Number of arguments: 1.

# resume los datos del modelo
modelo <- J48(class ~., data = German)
e <- evaluate_Weka_classifier(modelo, cost = matrix(c(0,5,1,0), ncol = 2), numFolds =
10, complexity = FALSE, seed = 123, class = TRUE)
e
==== 10 Fold Cross Validation ===

==== Summary ===

  Correctly Classified Instances       707        70.7    %
  Incorrectly Classified Instances     293        29.3    %
  Kappa statistic                      0.2409
  Total Cost                           709
  Average Cost                         0.709
  Mean absolute error                  0.3521
  Root mean squared error              0.4892
  Relative absolute error              83.7945 %
  Root relative squared error         106.7585 %
  Total Number of Instances            1000
```

```
==== Detailed Accuracy By Class ====
```

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0,370	0,149	0,516	0,370	0,431	0,247	0,605	0,415	bad
0,851	0,630	0,759	0,851	0,803	0,247	0,605	0,724	good
0,707	0,486	0,686	0,707	0,691	0,247	0,605	0,631	W.Avg.

```
==== Cost Matrix ====
```

```
0 1  
5 0
```

```
==== Confusion Matrix ====
```

a	b	<-- classified as
111	189	a = bad
104	596	b = good

```
# Objetos de la evaluación del modelo
```

```
objects(e)
[1] "confusionMatrix" "details"           "detailsClass"      "detailsCost"
[5] "string"
```

```
# Resumen sumario del modelo
```

```
summary(e)
   Length Class Mode
string       1   -none- character
details      8   -none- numeric
detailsCost   1   -none- numeric
detailsClass  12  -none- numeric
confusionMatrix 4   -none- numeric
```

```
e$details
  pctCorrect          pctIncorrect          pctUnclassified
  70.7000000          29.3000000          0.0000000
  kappa               meanAbsoluteError      rootMeanSquaredError
  0.2409326          0.3520856          0.4892290
  relativeAbsoluteError rootRelativeSquaredError
  83.7945248          106.7584751
```

```
e$detailsClass
  falsePositiveRate falseNegativeRate precision    recall   fMeasure
bad      0.1485714      0.6300000 0.5162791 0.3700000 0.4310680
good     0.6300000      0.1485714 0.7592357 0.8514286 0.8026936
  areaUnderROC
bad      0.6049929
good     0.6049929
```

```
# El modelo nos ofrece las reglas
```

```
modelo
J48 pruned tree
-----
```

```
checking_status = 'no checking': good (394.0/46.0)
checking_status = <0
|   foreign_worker = no: good (15.0/2.0)
|   foreign_worker = yes
|   duration <= 11
|   |   existing_credits <= 1
|   |   |   property_magnitude = 'life insurance'
|   |   |   |   own_telephone = none: bad (2.0)
|   |   |   |   own_telephone = yes: good (4.0)
|   |   |   property_magnitude = 'no known property': bad (3.0)
|   |   |   property_magnitude = 'real estate': good (8.0/1.0)
|   |   |   property_magnitude = 'car': bad (2.0/1.0)
|   |   |   existing_credits > 1: good (14.0)
```

```

duration > 11
|   job = 'high qualif/self emp/mgmt': good (30.0/8.0)
|   job = 'unemp/unskilled non res': bad (5.0/1.0)
|   job = 'unskilled resident'
|       purpose = 'domestic appliance': bad (1.0)
|       purpose = 'new car'
|           own_telephone = none: bad (10.0/2.0)
|           own_telephone = yes: good (2.0)
|       purpose = 'used car': bad (1.0)
|       purpose = business: good (3.0)
|       purpose = education: bad (1.0)
|       purpose = furniture/equipment
|           employment = <1: bad (3.0)
|           employment = >=7: good (2.0)
|           employment = 1<=X<4: good (4.0)
|           employment = 4<=X<7: good (1.0)
|           employment = unemployed: good (0.0)
|       purpose = other: good (1.0)
|       purpose = radio/tv
|           existing_credits <= 1: bad (10.0/3.0)
|           existing_credits > 1: good (2.0)
|       purpose = repairs: bad (1.0)
|       purpose = retraining: good (1.0)
job = skilled
|   other_parties = 'co applicant': bad (7.0/1.0)
|   other_parties = guarantor: good (12.0/3.0)
|   other_parties = none
|       duration <= 30
|           savings_status = 'no known savings'
|               existing_credits <= 1
|                   own_telephone = none: bad (9.0/1.0)
|                   own_telephone = yes: good (4.0/1.0)
|               existing_credits > 1: good (2.0)
|           savings_status = <100
|               credit_history = 'all paid': bad (6.0)
|               credit_history = 'critical/other existing credit': good
(14.0/4.0)

(7.0/2.0)
|   |   |   |   |   credit_history = 'delayed previously': bad (4.0)
|   |   |   |   |   credit_history = 'existing paid'
|   |   |   |   |       own_telephone = none
|   |   |   |   |           existing_credits <= 1
|   |   |   |   |               property_magnitude = 'life insurance': bad
good (2.0)
|   |   |   |   |       property_magnitude = 'no known property':
|   |   |   |   |           property_magnitude = 'real estate'
|   |   |   |   |               age <= 26: bad (5.0)
|   |   |   |   |               age > 26: good (2.0)
|   |   |   |   |       property_magnitude = car
|   |   |   |   |           credit_amount <= 1386: bad (3.0)
|   |   |   |   |               credit_amount > 1386: good (11.0/1.0)
|   |   |   |   |       existing_credits > 1: bad (3.0)
|   |   |   |   |       own_telephone = yes: bad (5.0)
|   |   |   |   |       credit_history = 'no credits/all paid': bad (8.0/1.0)
|   |   |   |   |       savings_status = >=1000: good (4.0)
|   |   |   |   |       savings_status = 100<=X<500
|   |   |   |   |           credit_history = 'all paid': good (1.0)
|   |   |   |   |           credit_history = 'critical/other existing credit': good
(2.0)
|   |   |   |   |       credit_history = 'delayed previously': bad (0.0)
|   |   |   |   |       credit_history = 'existing paid': bad (3.0)
|   |   |   |   |       credit_history = 'no credits/all paid': bad (0.0)
|   |   |   |   |       savings_status = 500<=X<1000: good (4.0/1.0)
|   |   |   |   |       duration > 30: bad (30.0/3.0)
|   |   |   |   |       checking_status = >=200: good (63.0/14.0)
|   |   |   |   |       checking_status = 0<=X<200

```

```

credit_amount <= 9857
savings_status = 'no known savings': good (41.0/5.0)
savings_status = <100
    other_parties = 'co applicant': good (2.0)
    other_parties = guarantor
        purpose = 'domestic appliance': good (0.0)
        purpose = 'new car': bad (2.0)
        purpose = 'used car': good (0.0)
        purpose = business: good (0.0)
        purpose = education: good (0.0)
        purpose = furniture/equipment: good (0.0)
        purpose = other: good (0.0)
        purpose = radio/tv: good (18.0/1.0)
        purpose = repairs: good (0.0)
        purpose = retraining: good (0.0)
    other_parties = none
    duration <= 42
        personal_status = 'female div/dep/mar'
            purpose = 'domestic appliance': good (0.0)
            purpose = 'new car': bad (5.0/1.0)
            purpose = 'used car': bad (1.0)
            purpose = business
                | residence_since <= 2: good (3.0)
                | residence_since > 2: bad (2.0)
            purpose = education: bad (4.0/2.0)
            purpose = furniture/equipment
                | duration <= 10: bad (3.0)
                | duration > 10
                    | duration <= 21: good (6.0/1.0)
                    | duration > 21: bad (2.0)
            purpose = other: good (0.0)
            purpose = radio/tv: good (8.0/2.0)
            purpose = repairs: good (1.0)
            purpose = retraining: good (0.0)
            personal_status = 'male div/sep': bad (8.0/2.0)
            personal_status = 'male mar/wid'
                duration <= 10: good (6.0)
                duration > 10: bad (10.0/3.0)
            personal_status = 'male single': good (52.0/15.0)
        duration > 42: bad (7.0)
savings_status = >=1000: good (13.0/3.0)
savings_status = 100<=X<500
    purpose = 'domestic appliance': bad (0.0)
    purpose = 'new car': bad (15.0/5.0)
    purpose = 'used car': good (3.0)
    purpose = business
        housing = 'for free': bad (1.0)
        housing = own: good (6.0)
        housing = rent
            | existing_credits <= 1: good (2.0)
            | existing_credits > 1: bad (2.0)
    purpose = education: bad (0.0)
    purpose = furniture/equipment: bad (4.0/1.0)
    purpose = other: good (1.0)
    purpose = radio/tv: bad (8.0/2.0)
    purpose = repairs: good (2.0)
    purpose = retraining: bad (0.0)
    savings_status = 500<=X<1000: good (11.0/3.0)
credit_amount > 9857: bad (20.0/3.0)

```

Number of Leaves : 98

Size of the tree : 135

3. REDES NEURONALES ARTIFICIALES

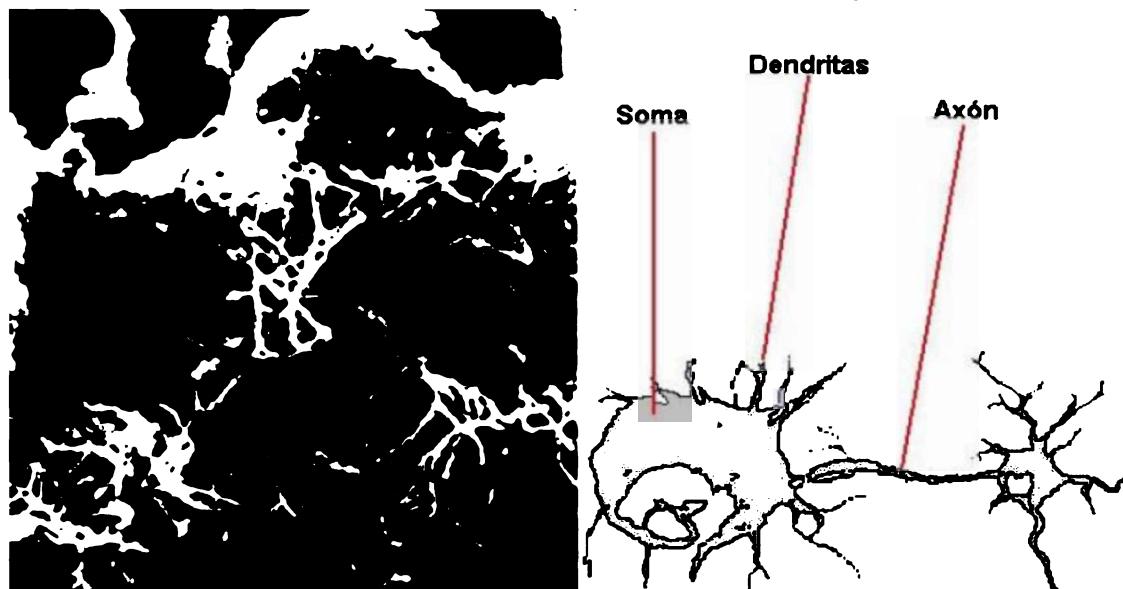
3.1. INTRODUCCIÓN

En la actualidad, las redes neuronales artificiales (RNA) constituyen un campo muy activo, fecundo y multidisciplinar. Existen numerosas aplicaciones desarrolladas con este método comparando su efectividad con otros métodos de clasificación.

En los últimos 25 años las redes neuronales artificiales han irrumpido como una potente herramienta estadística, tanto para problemas de clasificación como de regresión, o de agrupamiento. Su habilidad para procesar bases de datos con ruido o incompletas y su tolerancia a fallos, permiten a estas redes operar en tiempo real por su operatividad en paralelo.

La principal virtud de una red neuronal del tipo Perceptron Multicapa (Multilayer Perceptron), que explica su amplia utilización como técnica en el análisis de datos, es que es un aproximador universal de funciones. La base matemática de esta afirmación se debe a Kolmogorov (1957). Un Perceptrón contenido al menos una capa oculta con suficientes unidades no lineales, tiene la capacidad de aprender virtualmente cualquier tipo de relación, siempre que pueda ser aproximada en términos de una función continua (Cybenko, 1989; Funahashi, 1989; Hornik *et al.*, 1989).

FIGURA 4. MICROGRAFÍA AMPLIADA DE UN CÚMULO DE NEURONAS Y ESQUEMA DE LA MISMA.



Fuente: Brain Research Institute. UCLA en SAGAN 1980

En la parte izquierda de la figura 4 se puede observar un cúmulo de neuronas en el cerebro humano. Micrografía ampliada en 15.000 aumentos.

Las redes neuronales tratan de emular el comportamiento cerebral y están inspiradas en la estructura y funcionamiento de las redes neuronales biológicas. Las diferencias que separan a ambas redes neuronales quedan reflejadas en el siguiente cuadro:

TABLA 3. COMPARACIÓN DEL CEREBRO CON UN ORDENADOR CONVENCIONAL.

	Cerebro	Ordenador
Velocidad de procesamiento	10-3 s	10-9 s
Modo de procesamiento	Paralelo	Serie
Número de procesadores	1011	Pocos
Tipo de control del proceso	Democrático	Dictatorial
Conexiones	10000 por procesador	Pocas
Almacenamiento del conocimiento	Distribuido	En posiciones precisas
Tolerancia a fallos	Amplia	Poca o nula

Fuente: Nelson y Illingworth (1991).

Una red neuronal puede describirse mediante cuatro conceptos:

1. el tipo de modelo de red neuronal;
2. las unidades de procesamiento, que recogen información, la procesan, y arrojan un valor;
3. la organización del sistema de nodos para transmitir las señales desde los nodos de entrada a los nodos de salida y, por último,
4. la función de aprendizaje a través de la cual el sistema se retroalimenta.

3.2. TIPOS DE MODELOS DE REDES NEURONALES

Existen actualmente más de 40 paradigmas de redes neuronales artificiales. Se estima que tan solo cuatro arquitecturas:

1. el modelo perceptrón multicapa (MLP),
2. los mapas autoorganizados de Kohonen, (SOFM),
3. el vector de cuantificación (LVQ) y
4. las redes de base radial (RBF)

cubren, aproximadamente, el 90% de las aplicaciones prácticas de redes neuronales.

El modelo más utilizado es el perceptrón multicapa, que abarca el 70% de las aplicaciones, dado que se ha demostrado que este modelo es un aproximador universal de funciones (Funahashi, 1989).

El primer investigador que estudió el cerebro como una forma de ver el mundo de la computación fue Alan Turing, pero los primeros teóricos que concibieron los fundamentos de la computación neuronal fueron el neuro fisiólogo Warren McCulloch y el matemático Walter Pitts.

En 1949, Donald Hebb publica un importante libro titulado “La organización del comportamiento” en el que establece una clara conexión entre psicología y fisiología, y desarrolla una regla de cómo ocurría el aprendizaje.

Estos antecedentes sirven para que, en 1958, Frank Rosenblatt desarrolle el perceptrón, que es la red neuronal más antigua, utilizándose hoy en día en aplicaciones como reconocedor de patrones. Empezando con este autor, a continuación, se representa una relación de las principales redes neuronales con información sobre el tipo de aprendizaje utilizado, el año de creación y sus autores:

TABLA 4. CLASIFICACIÓN DE LAS RNA MÁS CONOCIDAS.

1. Supervisado**1. Con conexiones feedforward**

- Lineales
- Perceptrón (Rosenblatt, 1958)
- Adaline (Widrow y Hoff, 1960)
- Perceptrón multicapa (Multilayer perceptron) (MLP)
- Backpropagation (Rumelhart, Hinton y Williams, 1986)
- Correlación en cascada (Cascade correlation) (Fahlman y Lebiere, 1990)
- Quickpropagation (Quickprop) (Fahlman, 1988)
- Delta-bar-delta (Jacobs, 1988)
- Resilient Propagation (RPROP) (Riedmiller y Braun, 1993)
- Gradiente conjugado (Battiti, 1992)
- Radial Basis Function (RBF) (Broomhead y Lowe, 1988; Moody y Darken, 1989)
- Orthogonal Least Squares (OLS) (Chen, Cowan y Grant, 1991)
- Cerebellar Articulation Controller (CMAC) (Albus, 1975)
- Solo clasificación:
 - Learning Vector Quantization (LVQ) (Kohonen, 1988)
 - Red Neuronal Probabilística (PNN) (Probabilistic Neural Network) (Specht, 1990)
- Solo regresión:
 - General Regression Neural Network (GRNN) (Specht, 1991)

2. Con conexiones feedback

- Bidirectional Associative Memory (BAM) (Kosko, 1992)
- Máquina de Boltzman (Ackley, Hinton y Sejnowski, 1985)
- Series temporales recurrentes
- Backpropagation through time (Werbos, 1990)
- Elman (Elman, 1990)
- Finite Impulse Response (FIR) (Wan, 1990)
- Jordan (Jordan, 1986)
- Real-time recurrent network (Williams y Zipser, 1989)
- Recurrent backpropagation (Pineda, 1989)
- Time Delay NN (TDNN) (Lang, Waibel y Hinton, 1990)

3. Competitivo

- ARTMAP (Carpenter, Grossberg y Reynolds, 1991)
- Fuzzy ARTMAP (Carpenter, Grossberg, Markuzon, Reynolds y Rosen, 1992)
- Gaussian ARTMAP (Williamson, 1995)
- Counterpropagation (Hecht-Nielsen, 1987, 1988, 1990)
- Neocognitrón (Fukushima, Miyake e Ito, 1983; Fukushima, 1988)

2. No supervisado

1. Competitivo

- Vector Quantization
- Grossberg (Grossberg, 1976)
- Kohonen (Kohonen, 1984)
- Conscience (Desieno, 1988)
- Mapa Auto-Organizado (Self-Organizing Map) (Kohonen, 1982; 1995)
- Teoría de la Resonancia Adaptativa (Adaptive Resonance Theory, ART)
- ART 1 (Carpenter y Grossberg, 1987a)
- ART 2 (Carpenter y Grossberg, 1987b)
- ART 2-A (Carpenter, Grossberg y Rosen, 1991a)
- ART 3 (Carpenter y Grossberg, 1990)
- Fuzzy ART (Carpenter, Grossberg y Rosen (1991b))
- Differential Competitive Learning (DCL) (Kosko, 1992)

2. Reducción de dimensionalidad

- Regla de Oja (Oja, 1989)
- Sanger (Sanger, 1989)
- Differential hebbian (Kosko, 1992)

3. Autoasociación

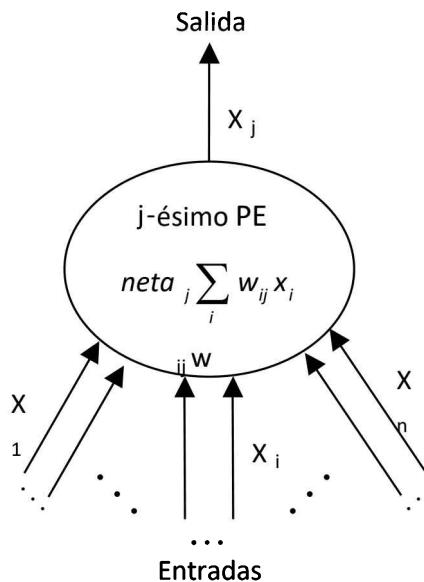
- Autoasociador lineal (Anderson, Silverstein, Ritz y Jones, 1977)
- Brain-State-in-a-Box (BSB) (Anderson, Silverstein, Ritz y Jones, 1977)
- Red de Hopfield (1982)

Fuente: Montaño (2005)

3.3. UNIDADES DE PROCESAMIENTO DE LA INFORMACIÓN

El elemento básico de una red neuronal es un nodo. Es la unidad de procesamiento que actúa en paralelo con otros nodos de la red. Es similar a las neuronas del cerebro humano: acepta input y genera output. Su forma de actuar se muestra en la figura 5; los nodos aceptan input de otros nodos. La primera tarea del nodo es procesar los datos de entrada creando un valor resumen que es la suma de todas las entradas multiplicadas por sus ponderaciones. Este valor resumen se procesa a continuación mediante una función de activación para generar una salida que se envía al siguiente nodo del sistema.

FIGURA 5. UNIDAD BÁSICA DE UNA RED NEURONAL



Fuente: Elaboración propia

Las cuatro funciones de activación más utilizadas son:

a) La función escalón

Considerando que la activación debe llegar a un determinado nivel U (umbral de activación), esta función adopta la forma:

$$x_j = f_j(neta_j) = \begin{cases} 1 & \forall neta_j \geq U \\ 0 & \forall neta_j < U \end{cases}$$

Para evitar la discontinuidad se utilizan frecuentemente las tres funciones siguientes, (con mayor frecuencia las dos primeras), sin que por ello se agoten ni mucho menos todas las posibilidades.

b) Función identidad

Es la función más sencilla.

c) Función sigmoide o $x_j = \text{neta}_j$ logística.

$$x_j = \frac{1}{1 + e^{-\text{neta}_j}} \quad [54]$$

Esta función que toma valores entre 0 y 1, es conocida como la sigmoide asimétrica.

d) Tangente hiperbólica

$$x_j = 1 - \frac{1}{e^{2\text{neta}_j} + 1} \quad [55]$$

Esta función varía entre -1 y 1. Es simétrica y en la literatura se la denomina frecuentemente sigmoide simétrica. Estas funciones ofrecen mayores ventajas, fundamentalmente una convergencia más rápida en el aprendizaje de la red.

FIGURA 6. FUNCIONES ACTIVACIÓN MÁS UTILIZADAS EN REDES NEURONALES.

	Función	Rango	Gráfica
Identidad	$y = x$	$[-\infty, +\infty]$	
Escalón	$y = \text{signo}(x)$ $y = H(x)$	$\{-1, +1\}$ $\{0, +1\}$	
Lineal a tramos	$y = \begin{cases} -1, & \text{si } x < -l \\ x, & \text{si } -l \leq x \leq +l \\ +1, & \text{si } x > +l \end{cases}$	$[-1, +1]$	
Sigmoidea	$y = \frac{1}{1 + e^{-x}}$ $y = \text{tgh}(x)$	$[0, +1]$ $[-1, +1]$	
Gaussiana	$y = A.e^{-Bx^2}$	$[0, +1]$	
Sinusoidal	$y = A.\text{sen}(\omega x + \varphi)$	$[-1, +1]$	

Fuente: Martín del Brío y Sanz (2006)

3.4. PROPIEDADES DE LOS SISTEMAS NEURONALES

Se considera una red neuronal la ordenación secuencial de tres tipos básicos de nodos o capas: nodos de entrada, nodos de salida y nodos intermedios (capa oculta o escondida).

Los nodos de entrada se encargan de recibir los valores iniciales de los datos de cada caso para transmitirlos a la red. Los nodos de salida reciben entradas y calculan el valor de salida (no van a otro nodo). En casi todas las redes existe una tercera capa denominada oculta.

Este conjunto de nodos utilizados por la red neuronal, junto con la función de activación, posibilita a las redes neuronales representar fácilmente las relaciones no lineales, que son muy problemáticas para las técnicas multivariantes.

Se puede decir que una red neuronal tiene tres ventajas que le hacen muy atractiva en el tratamiento de los datos: aprendizaje, robustez y paralelismo masivo:

- ✓ Aprendizaje adaptativo a través de ejemplos

Una de las características más sobresalientes de las redes neuronales y que la aleja del resto de las técnicas multivariantes es su capacidad de aprender o de corregirse a sí misma

basándose en los errores. Se puede considerar que el conocimiento se encuentra representado en los pesos de las conexiones entre las neuronas y en sus umbrales. El proceso de aprendizaje implica cierto número de cambios en estos valores de tal forma que se puede decir que “se aprende modificando los valores de los pesos y umbrales de las neuronas de la red”.

Un criterio para clasificar las redes neuronales es respecto a las reglas de aprendizaje. Si el aprendizaje se basa en la existencia de un agente externo decimos que la red neuronal es supervisada, mientras que cuando no interviene el analista estamos frente a una red neuronal no supervisada. Un ejemplo de una red supervisada es el perceptrón multicapa que se describe a continuación, mientras que una red neuronal no supervisada es la red de Kohonen, que también se describe en este módulo.

- ✓ Tolerancia a fallos

Algunas de las capacidades de la red, se pueden retener aún si ésta sufre daños. Las redes neuronales artificiales son muy robustas en el tratamiento de la información redundante e imprecisa.

- ✓ Paralelismo masivo

Lo que significa que las operaciones se realizan en tiempo real. Los cómputos de la red pueden realizarse en paralelo para lo cual se pueden fabricar máquinas con hardware especial.

3.5. PERCEPTRÓN MULTICAPA

El método más utilizado en las aplicaciones prácticas de redes neuronales es el perceptrón multicapa, que fue popularizado por Rumelhart *et al.* (1986). Este modelo de red es conocido también como backpropagation error (propagación del error hacia atrás), también denominado método del gradiente decreciente.

La verdadera razón de su tremenda utilidad radica en su capacidad de organizar una representación interna del conocimiento en las capas ocultas de neuronas a fin de aprender la relación entre un conjunto de datos de entradas y salidas. El perceptrón multicapa es un aproximador universal de funciones. La red backpropagation, conteniendo al menos una capa oculta, es capaz de aprender cualquier tipo de función o relación continua. Esta propiedad convierte a esta red en una herramienta de propósito general.

3.5.1. Etapa de funcionamiento

El desarrollo de la red consta de una fase de funcionamiento y otra de aprendizaje.

En la primera etapa, cuando se presenta un patrón de entrada $X_p: x_{p1}, \dots, x_{pb}, \dots, x_{pN}$, se transmite a la red a través de los pesos w_{ji} desde la capa de entrada a la capa oculta. Las neuronas de esta capa transforman las señales a través de la función de activación proporcionando un valor de salida. Este valor se transmite a su vez a través de los pesos v_{kj} a la capa de salida, donde aplicando de nuevo la función de activación obtenemos un valor de salida.

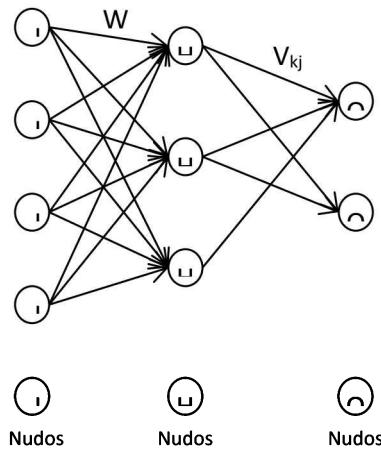
Vamos a suponer que la entrada total o neta de una neurona oculta j la expresamos como net_{pj} , entonces matemáticamente la podemos expresar de la siguiente manera:

$$net_{pj} = \sum_{i=1}^N w_{ji} x_{pi} + \theta_j \quad [56]$$

θ es el umbral de la neurona, que se considera como un peso asociado a una neurona ficticia con valor de salida igual a 1. El valor de salida de la neurona oculta j, b_{pj} lo obtenemos aplicando la función de activación $f(\cdot)$ sobre la entrada neta.

$$b_{pj} = f(net_{pj}) \quad [57]$$

FIGURA 7. RESPUESTA LOCALIZADA DE LAS NEURONAS OCULTAS EN LA RBF.



Fuente: Elaboración propia

La entrada neta que recibe una neurona de salida k la podemos expresar como:

$$net_{pk} = \sum_{j=1}^L v_{kj} b_{pj} + \theta_k \quad [58]$$

El valor de salida de la neurona k, y_{pk} es el siguiente:

$$y_{pk} = f(net_{pk}) \quad [59]$$

3.5.2. Etapa de aprendizaje

En esta segunda etapa el objetivo que se persigue es hacer mínima la discrepancia o error entre la salida de la red y el valor real que presenta el usuario. La función que se pretende minimizar para cada patrón p viene dada por la siguiente expresión:

$$E_p = \frac{1}{2} \sum_{k=1}^M (d_{pk} - y_{pk})^2 \quad [60]$$

Donde d_{pk} es la salida presentada por la red de la neurona k ante la presentación del patrón p. La medida general del error es la suma de todos los errores para todos los patrones.

$$E = \sum_{p=1}^P E_p \quad [61]$$

El objetivo fundamental del análisis con las RNA es minimizar el cuadrado de los errores entre el valor real y el de la variable salida, dado por la ecuación 61. Esta función depende fundamentalmente de dos parámetros: el conjunto de variables de entrada, las variables explicativas de nuestro modelo, y el conjunto de pesos sinápticos.

Disponemos de un buen número de métodos o algoritmos que permiten estimar los parámetros del modelo de RNA minimizando E_p . Esta minimización de la función debe ser realizada por métodos de optimización no lineales. Existen diferentes enfoques a la hora de aplicar estos métodos y deben ser adaptados dependiendo de la dimensión y de la complejidad del problema. En los métodos no lineales la función E_p es una función continua y diferenciable, así que podemos aplicar los métodos del gradiente. En estos métodos, el proceso de búsqueda de la solución óptima usado puede ser descrito como:

$$\vartheta_{t+1} = \vartheta_t + \lambda_t \Delta_t \quad [62]$$

Donde ϑ_t es la solución que es viable, λ_t es el tamaño del peso y $\Delta_t = M_t g_t$ es el vector de dirección. Para el vector de dirección se tiene que M_t es una matriz definida positiva y $g_t = g(\vartheta_t)$ es el vector gradiente.

En RNA la técnica más utilizada es la del gradiente decreciente, Rumelhart *et al.* (1986), denominada algoritmo backpropagation debido a la forma de modificación de los pesos. Este gradiente toma la dirección que determina el incremento más rápido en el error. Así que, el error puede reducirse ajustando cada peso en la siguiente dirección:

$$-\sum_{p=1}^P \frac{\partial E_p}{\partial w_{ji}} \quad [63]$$

Entre los métodos gradientes se encuentra el método de Newton-Rapshon. Este algoritmo utiliza la inversa de la matriz hessiana como matriz M. En este procedimiento puede ocurrir que el valor del punto inicial ϑ_0 no esté próximo al punto óptimo, lo que dificultaría el cálculo de la matriz hessiana, además de que puede suceder que en algunas aplicaciones esta matriz sea difícil de calcular, por lo que diferentes investigadores han desarrollado procedimientos de estimación que se conocen como métodos Quasi-Newton. Estos métodos usan una aproximación iterativa de la matriz hessiana:

$$M_{t+1} = M_t + N_t \quad [64]$$

Donde N_t es una matriz definida positiva, lo que garantiza que en cada paso del proceso iterativo la aproximación sea también definida positiva, al ser suma de dos matrices definidas positivas. En esta implementación tenemos que seleccionar el punto inicial ϑ_0 y una matriz M_0 que deberá ser definida positiva.

Uno de los algoritmos empleados para la estimación de la estructura de la RNA es el BFGS (Broyden-Fletcher-Goldfarb-Shanno) descrito en Fletcher (1987) que utiliza la siguiente expresión iterativa:

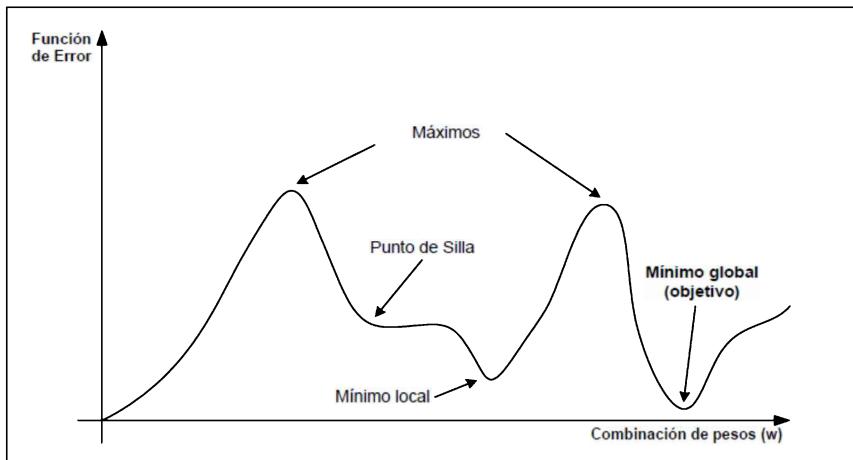
$$M_{t+1} = M_t \frac{\delta_t \delta_t'}{\delta_t' v_t} + \frac{M_t v_t v_t' M_t}{v_t' M_t \delta_t} - v_t' M_t v_t \left(\frac{\delta_t}{\delta_t' v_t} - \frac{M_t v_t}{v_t' M_t v_t} \right) \left(\frac{\delta_t}{\delta_t' v_t} - \frac{M_t v_t}{v_t' M_t v_t} \right)$$

[65]

donde $\delta_t = \theta_{t+1} - \theta_t$ y $v_t = g(\theta_{t+1}) - g(\theta_t)$

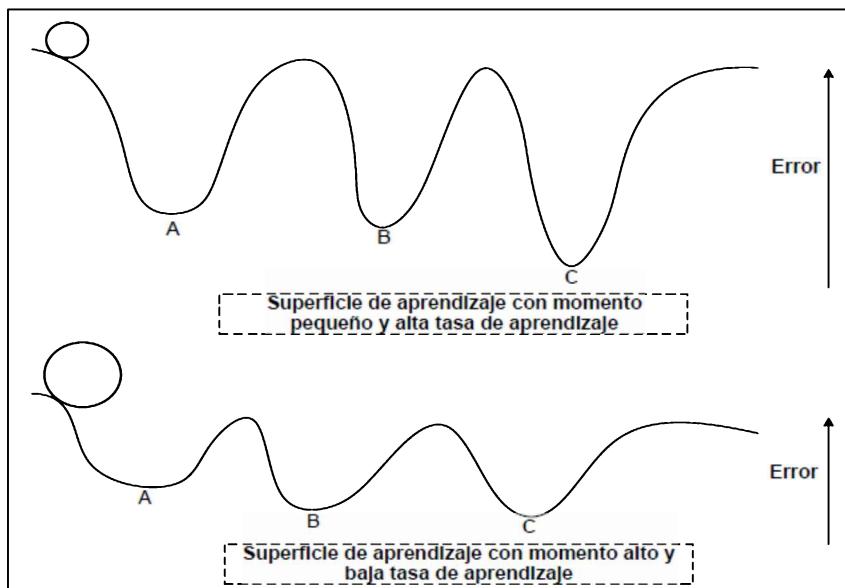
Un problema que se puede presentar al utilizar este método del gradiente conjugado, es que caiga en mínimos locales. Sin embargo, como ya han señalado diversos autores, se da en contadas ocasiones cuando se trabaja con datos reales.

FIGURA 8. COMPLEJIDAD EN LA BÚSQUEDA DEL MÍNIMO GLOBAL



Fuente: Urquiza y Mendivil (2011).

FIGURA 9. DINÁMICA EN LA BÚSQUEDA DEL MÍNIMO GLOBAL



Fuente: Urquiza y Mendivil (2011).

Según Nisbet *et al.* (2009), en las redes neuronales artificiales configuradas manualmente, una tasa de aprendizaje del orden del 0,9 obtiene los mejores resultados, junto a momentos que oscilen entre 0,1 y 0,3.

El hecho de que se pueda definir un momento con valores entre 0 y 1, tal y como señalan algunos autores, hace prácticamente imposible realizar una búsqueda exhaustiva para

encontrar la mejor combinación, junto con la tasa de aprendizaje, implicando el diseño de RNA supervisadas una tarea que implica necesariamente cierto ensayo y error.

Otro parámetro que es necesario controlar adecuadamente es la tasa de disminución de aprendizaje, puesto que comúnmente la tasa de aprendizaje va decayendo a medida que se realiza el entrenamiento. Esto tiene el efecto importante de aplanar la superficie de búsqueda y de esta manera permitir encontrar los mínimos globales de una forma más fácil.

En la práctica, la forma de modificar los pesos de forma iterativa se realiza aplicando la regla de la cadena a la expresión del gradiente, y añadiendo una tasa de aprendizaje que se denomina η . Para una neurona de salida, obtenemos la siguiente expresión:

$$\Delta v_{kj}(n+1) = \eta \sum_{p=1}^P \delta_{pk} b_{pj} \quad [66]$$

Donde δ_{pk} tiene la siguiente expresión: $\delta_{pk} = (d_{pk} - y_{pk})f'(net_{pk})$ y η nos indica la iteración.

Para el peso de una neurona oculta:

$$\Delta w_{ji}(n+1) = \eta \sum_{p=1}^P \delta_{pj} x_{pi} \quad [67]$$

donde δ_{pj} es igual a:

$$\delta_{pj} = f'(net_{pj}) \sum_{k=1}^M \delta_{pk} v_{kj}$$

Para acelerar el proceso de convergencia de los pesos se sugiere añadir a la expresión un factor momento, α , el cual tiene en cuenta la dirección del incremento tomado en la iteración anterior. Por ejemplo, para el peso de una neurona de salida, la expresión es la siguiente:

$$\Delta w_{ji}(n+1) = \alpha(n)(x_{pi} - w_{ji}(n)) \quad [68]$$

Y para el peso de una neurona oculta:

$$\Delta v_{kj}(n+1) = \eta \left(\sum_{p=1}^P \delta_{pk} b_{pj} \right) + \alpha \Delta v_{kj}(n) \quad [69]$$

3.5.3. Metodología de aplicación de un perceptrón multicapa

A continuación, se ofrecen una serie de pasos para la ejecución de este modelo de red, que es el modelo más extendido, en las aplicaciones prácticas:

- ✓ **Elección adecuada de las variables**

Para obtener los mejores resultados se deberán de escoger cuidadosamente las variables. La introducción de variables irrelevantes en el modelo o que estén muy correlacionadas puede generar un sobreajuste, lo que puede provocar una disminución sensible de su capacidad de generalización.

✓ **Creación de los conjuntos de aprendizaje, validación y test**

La muestra de los datos se divide generalmente en tres conjuntos: entrenamiento, validación y test. Para evitar el sobreajuste es aconsejable utilizar un segundo grupo que nos permita controlar el proceso de aprendizaje de la red neuronal. El conjunto de datos que forman el grupo de test es aconsejable porque debemos de disponer de un grupo independiente del proceso de entrenamiento de la red para probar su eficacia y que nos proporcionará una estimación insesgada del error de generalización.

✓ **Entrenamiento de la red neuronal**

Para que la red efectúe el entrenamiento hay que proporcionar una serie de elementos determinados mediante ensayo y error. El grupo de validación nos permitirá optimizar este conjunto de parámetros: elección de los pesos iniciales, arquitectura de la red, tasa de aprendizaje y factor momento, y las funciones de activación de las neuronas ocultas y de salida.

Para que la red empiece su etapa de entrenamiento se han de designar los pesos de las conexiones y los valores umbrales. Una forma sencilla y muy utilizada es asignar pesos pequeños elegidos de forma aleatoria en un rango de valores entre -0,5 y 0,5.

En cuanto a la arquitectura de la red es sabido que para la resolución de la mayor parte de los problemas es suficiente con utilizar una sola capa oculta. El número de neuronas de la capa de entrada está determinado por las variables predictoras. Las neuronas de la capa de salida están en función de si el problema que queremos resolver es de clasificación o estimación. Las neuronas de la capa oculta, que nos determinan la capacidad de aprendizaje de la red, se deben de elegir de forma que la red rinda de forma adecuada con el menor número de neuronas en esta capa. No hay teoría que nos lo indique, pero disponemos del conjunto de validación para probar diferentes arquitecturas.

La tasa de aprendizaje controla el tamaño del cambio en el proceso de entrenamiento de la red. Si el ritmo de aprendizaje es muy pequeño la velocidad de convergencia disminuye y se puede caer en mínimos locales. Un valor muy alto puede ocasionar inestabilidades y evitar la convergencia. Algunos autores recomiendan probar con valores comprendidos entre 0,005 y 0,5.

El factor momento (α) permite filtrar las oscilaciones en la superficie del error que provoca la tasa de aprendizaje. El valor que suele tomar es próximo a 1.

El algoritmo backpropagation exige que las funciones de activación sean derivables para poder obtener el valor del error de las neuronas de la capa oculta y de salida. Si queremos aprovechar la capacidad que tienen las redes neuronales artificiales de aprender relaciones complejas y no lineales es imprescindible la utilización de funciones no lineales, al menos, en la capa oculta. En general se utilizan la función logística o la tangente hiperbólica.

3.5.4. Evaluación del rendimiento del modelo

El último paso y el más importante es evaluar la capacidad de generalización que ha conseguido la red. Para este fin se ha reservado el tercer conjunto de datos de test. Cuando el problema es de estimación, normalmente se utiliza la media cuadrática del error. Si el problema es de clasificación, podemos construir una tabla de confusión y sobre los datos calcular diferentes índices de asociación. Si estamos interesados en discriminar entre dos categorías, se puede hacer uso de los índices de sensibilidad, especificidad y eficacia y del

análisis de curvas ROC (Receiver Operating Characteristic) (Palmer, Montaño y Calafat, 2000).

La expresión de la media cuadrática del error se calcula a través de la siguiente fórmula:

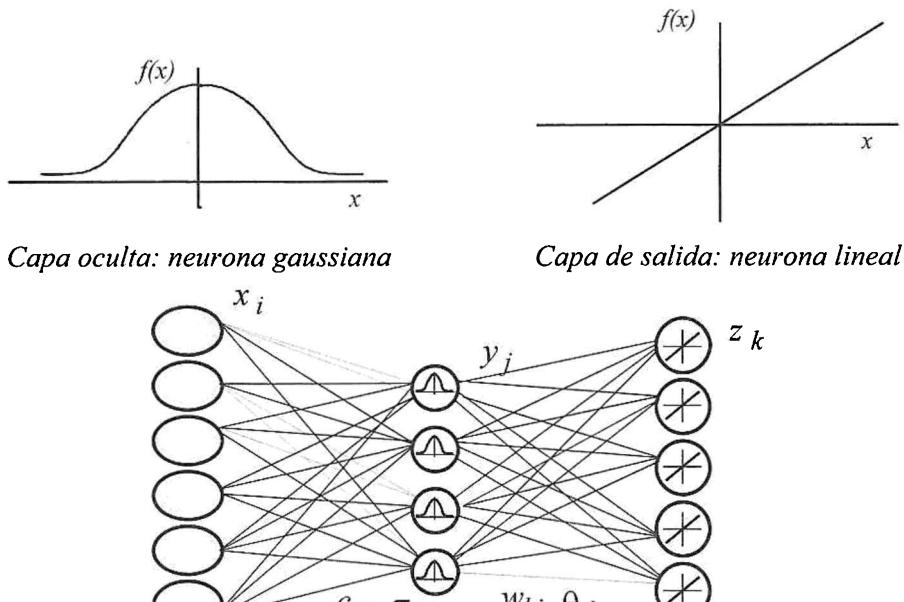
$$MC_{Error} = \frac{\sum_{p=1}^P \sum_{k=1}^M (d_{pk} - y_{pk})^2}{P \cdot M} \quad [70]$$

3.6. FUNCIONES DE BASE RADIAL

Las funciones de base radial (Radial Basis Functions o RBS) (Moody y Darken 1989, Poggio 1990) cuentan cada vez con más aplicaciones prácticas debido, sobre todo, a su simplicidad. Una función de base radial es una función cuya característica principal es que su respuesta disminuye (o aumenta) monótonamente con la distancia a un punto fijo llamado centro (o centroide).

Este modelo se puede considerar como un modelo híbrido de red neuronal al incorporar aprendizaje supervisado y no supervisado.

FIGURA 10. ARQUITECTURA DE UNA RED NEURONAL RBF



Fuente: Martín del Brío y Sanz (2006)

Al igual que en el perceptrón multicapa, se pueden modelar fácilmente sistemas no lineales en menor tiempo que el modelo clásico. Al igual que en el perceptrón multicapa, estas arquitecturas constituyen aproximadores universales de funciones.

Este modelo de red cuenta con tres capas de neuronas: de entrada, oculta y de salida. Las neuronas de entrada envían información a la capa oculta como en el perceptrón multicapa. La diferencia fundamental se halla en como procesan la información de las neuronas de la capa oculta. En esta capa se opera en base a la distancia que separa el vector de entrada respecto del vector sináptico que cada neurona de la capa oculta almacena. A estos vectores

de la capa oculta se les llama centroides. A esta cantidad se le aplica una función radial con forma guassiana.

Debido a esto, en esta capa las neuronas tienen una respuesta localizada respondiendo con una intensidad apreciable cuando el vector de entradas y el centroide de la neurona pertenecen a una zona próxima en el espacio de las entradas. De forma matemática el modelo se expresa así:

$$r_j^2 = \|x - c_j\|^2 = \sum_i (x_i - c_{ji})^2 \quad [71]$$

Donde x_i son las entradas de la red y c_{ji} representa al centroide.

La salida de la neurona y_j se calcula a partir de la función de activación denominada función radial $\phi(r)$, que suele ser la función gaussiana.

$$\phi(r) = e^{-r^2/\sigma^2} \quad [72]$$

En esta función, cuanto mayor sea σ mayor será la región que la neurona domina en torno al centroide.

La función de salida y_j es la siguiente:

$$y_j = e^{-r_j^2/\sigma_j^2} = e^{-\sum_i (x_i - c_{ji})^2 / \sigma_j^2} \quad [73]$$

Cuando el vector de entrada se aproxima al centroide de una neurona, ésta se activa, lo que significa que reconoce al patrón de entrada. Sin embargo, si el patrón de entrada es muy diferente al centroide la respuesta tiende a cero.

Las salidas de las neuronas ocultas son las entradas de las neuronas de la capa de salida z_k cuya expresión es la siguiente:

$$z_k = \sum_j w_{kj} y_j + \phi_k = \sum_j w_{kj} \phi(r_j) + \phi_k \quad [74]$$

Donde w_{kj} es el peso que conecta la neurona oculta j con la salida k , y ϕ_k un parámetro adicional de la neurona k que es el umbral.

En el aprendizaje de las redes de base radial hay que tomar en cuenta, en primer lugar, el número de neuronas en la capa oculta, también llamados nodos ocultos radiales. Al margen del método de prueba y error, se pueden emplear algunos procedimientos más o menos automáticos como el modelo auto organizado jerárquico (Hierarchically Self organizing Learning Algorithm) de Lee y Kil (1991) para su determinación.

Una vez determinado el número de nodos radiales, suele emplearse un aprendizaje por etapas, donde primero se realiza el entrenamiento de las neuronas ocultas y después se procede al entrenamiento de las neuronas de salida.

Existen tres enfoques para el aprendizaje de las redes:

1. Selección fija de centros, pesos y varianzas.
2. Aprendizaje auto-organizado de los centros.
3. Aprendizaje supervisado.

Los dos primeros suponen funciones gaussianas normalizadas:

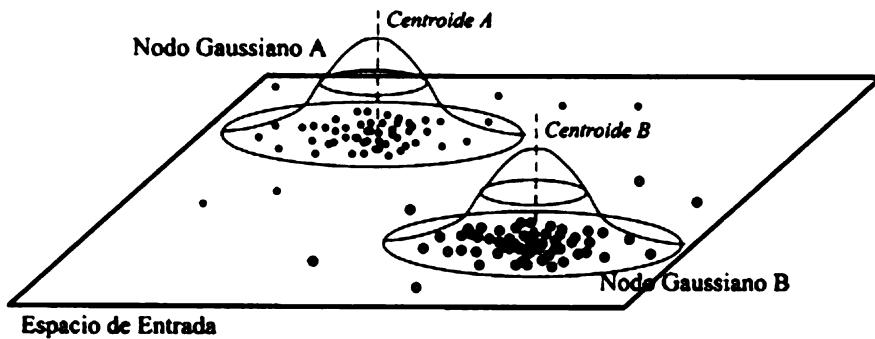
$$h_c(x) = \exp\left(-\frac{1}{\sigma^2} \|x - c\|^2\right) \quad [75]$$

donde c es el centro, σ es la varianza (lo que regula la anchura).

El tercer caso, para más versatilidad, considera el caso general de las gaussianas.

Las funciones de base radial se denominan así precisamente por la simetría radial de estas funciones, lo que aquí significa es que el nodo da una salida idéntica para aquellos patrones que distan lo mismo del centroide. El parámetro de normalización, o factor de escala σ , nos mide la anchura de la función gaussiana y equivale al radio de influencia de la neurona en el espacio de las entradas. Cuanto mayor es el valor de σ la región que la neurona domina en torno al centroide es más amplia, como puede observarse en el siguiente gráfico. Los puntos en el gráfico representan patrones en el espacio de las entradas y, como se puede observar, se agrupan en su mayoría en torno a dos centros.

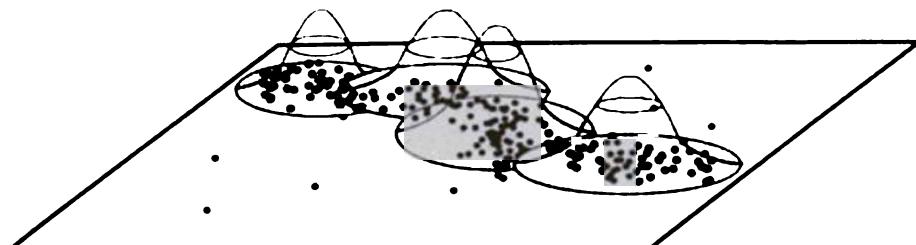
FIGURA 11. RESPUESTA LOCALIZADA DE LAS NEURONAS OCULTAS EN LA RBF.



Fuente: Martín del Brío y Sanz (2006).

En las redes de base radial cada nodo se ocupa de una zona del espacio y el conjunto de ellos debe de cubrir la zona de interés. Este proceso de cubrimiento tiene que llevarse de la manera más suave posible controlando el número de nodos de la capa oculta y con la amplitud de σ . En el grafico siguiente se observa cómo cuatro nodos gaussianos cubren el espacio de trabajo.

FIGURA 12. NODOS GAUSIANOS RECUBRIENDO EL ESPACIO DE TRABAJO.



Fuente: Martín del Brío y Sanz (2006).

A la familia de las redes RBF pertenecen otras arquitecturas de redes, entre otras, la Red Neuronal Probabilística (PNN, Probabilistic Neural Network) (Specht, 1990), la General Regression Neural Network (GRNN) (Specht, 1991) y la Counterpropagation (Hecht-Nielsen, 1987, 1988, 1990).

3.7. COMPARACIÓN ENTRE LAS FUNCIONES DE BASE RADIAL Y EL PERCEPTRÓN MULTICAPA

Una RBF tiene solo una capa escondida, mientras que MLP puede tener varias.

La capa oculta de una RBF es no lineal, mientras que la capa de salida es lineal. Mientras que en un MLP como clasificador las capas escondida y de salida usualmente son no lineales; y cuando MLP se usa para síntesis funcional la capa de salida se elige como lineal.

El argumento de activación de la función de activación de una neurona escondida de una RBF es una distancia euclídea entre el vector entrante y el centro de esa unidad. En MLP el argumento de la función de activación de cada neurona escondida es un producto interno del vector de entrada con el de los pesos.

MLPs construyen aproximaciones globales, pero las RBF construyen aproximaciones locales.

3.8. ANÁLISIS DE SENSIBILIDAD E INTERPRETACIÓN DE LOS PESOS DE LA RED

En las investigaciones de las redes neuronales se ha prestado mucha atención al desarrollo de las arquitecturas y desarrollo de algoritmos y existen pocas investigaciones relacionadas con los procedimientos que nos ayuden a comprender la verdadera naturaleza de las representaciones internas generadas por la red, Montaño y Palmer (2003).

Una de las principales ventajas que se han señalado de las Redes Neuronales es que se han visto como una suerte de “cajas negra”, y este hecho, tal y como afirma Shachmurove (2002), señala una indeterminación observacional que surge de la naturaleza auto poética o de auto organización de las Redes neuronales artificiales, lo que hace muy complejo entender como las relaciones en las capas ocultas son estimadas.

Tanto en Montaño y Palmer (2003) como en Nisbet *et al.* (2009) se observa cómo estudios recientes están abriendo en gran medida esa caja negra, como algunos autores la han denominado, a través del análisis de sensibilidad.

Para abrir esta caja negra de las redes neuronales existen dos metodologías que nos ayudan a interpretar qué ha aprendido el perceptrón multicapa a partir del valor de los pesos y de los valores de activación de las neuronas. Con el conocimiento de estos valores, lo que se pretende es conocer la importancia de cada variable independiente sobre la variable de salida.

Las dos metodologías que se comentan brevemente a continuación están basadas, la primera, en un análisis sobre los pesos de las neuronas y la segunda recibe el nombre de análisis de sensibilidad donde se estudia el efecto que produce en una variable de salida el cambio originado en una variable de entrada, o bien en el error cometido.

3.8.1. Análisis basado en la magnitud de los pesos de la red

Estos procedimientos se basan solo en los valores de la matriz estática de pesos, y tienen como propósito determinar la influencia relativa que tiene cada variable de entrada sobre la de salida.

No es correcto suponer, tal como nos lo recuerda Masters (1993), que las entradas con pesos con valor absoluto mayor tengan que ser las más importantes o las de peso menor sean las menos significativas. En la literatura sobre redes neuronales existen diferentes propuestas de ecuaciones basadas en la magnitud de los pesos si bien todas ellas se caracterizan por calcular el producto de los pesos w_{ij} y v_{jk} para cada una de las neuronas ocultas: Yoon *et al.* (1989); Garson (1991a); Garson (1991b); Yoon *et al.* (1993); Milne (1995); Gedeon (1997); Tsaih (1999).

Una de las expresiones más utilizadas es la propuesta por Garson, (1991a, 1991b) y por Modai *et al.* (1995):

$$Q_{ik} = \frac{\sum_{j=1}^L \left(\frac{w_{ij}}{\sum_{r=1}^N w_{rj}} v_{jk} \right)}{\sum_{i=1}^N \sum_{j=1}^L \left(\frac{w_{ij}}{\sum_{r=1}^N w_{rj}} v_{jk} \right)} \quad [76]$$

donde $\sum_{r=1}^N w_{rj}$ es la suma de los pesos de conexión entre las i neuronas de entrada y la neurona oculta j .

Este índice Q_{ik} es el porcentaje de influencias de la variable de entrada i sobre la salida k y la suma de este índice para todas las variables debe de valer el 100%.

3.8.2. Análisis de sensibilidad

El análisis de sensibilidad según Yeung *et al.* (2010) se refiere a observar cómo la salida (el output) es afectada por las entradas o inputs y/o en los pesos sinápticos tal y como hemos visto anteriormente. Sobre este importante tema podemos encontrar tres desarrollos que se comentan brevemente a continuación.

Análisis de sensibilidad basado en el error

La función de error que se utiliza tiene la siguiente fórmula:

$$RMC_{\text{error}} = \sqrt{\frac{\sum_{p=1}^P \sum_{k=1}^M (d_{pk} - y_{pk})^2}{P \cdot M}} \quad [77]$$

donde d_{pk} es la salida deseada para el patrón p en la neurona de salida k .

El método consiste, siguiendo a Frost y Karry (1999), en aplicar pequeños incrementos a las variables de entrada mientras se mantienen los valores originales del resto de variables de entrada de la red. Una vez realizado este paso se procede a entrenar la red calculando el valor RMC_{error} . Si seguimos este procedimiento para todas las variables, podemos determinar una ordenación de las mismas: la variable de entrada que consiga el mayor RMC será considerada como la variable más influyente sobre la variable de salida.

Otras variantes de este procedimiento las podemos encontrar en Masters (1993).

Análisis de sensibilidad basado en la salida

Citando al estudio de Engelbrecht *et al.* (1999), el análisis de sensibilidad puede ser empleado para una gran variedad de tareas como son: optimización, robustez y análisis de la relación inputs/outputs, inferencia y causalidad, aprendizaje selectivo o reducción del conjunto de inputs, entre otros.

Esta forma de estudiar la sensibilidad se basa en los efectos que se producen en las salidas debido a los cambios de las variables de entrada. Sobre la red entrenada se fija el valor de todas las entradas a su valor medio. Variando el valor de una de las variables o introduciendo ruido podemos registrar los cambios producidos en la red.

Esta forma de proceder ha sido aplicada a muchos campos del conocimiento en tareas de predicción. Algunos de ellos son los siguientes: predicción del comportamiento de la bolsa (Bilge *et al.*, 1993), predicción de las auto expectativas en niños, Reid *et al.* (1994) y Kashani *et al.* (1996), en el análisis de supervivencia por De Laurentiis y Ravdin, (1994), en estudios de tratamiento psiquiátrico, Modai *et al.* (1995) o en la predicción farmacológica, Opara *et al.* (1999).

Otra forma de proceder es a través de la matriz Jacobiana (SIK) dado que los elementos de la misma nos proporcionan de forma analítica una medida de sensibilidad de las salidas por cambios efectuados en las variables de entrada.

Si tomamos en cuenta que una red neuronal artificial opera en un mapa no lineal y diferenciable $\Gamma: R^I \rightarrow R^K$ desde un vector de entradas $X = (X_1, X_2, \dots, X_I)$ hacia un vector de salidas $Y = (Y_1, Y_2, \dots, Y_K)$. La matriz Jacobiana toma la siguiente expresión:

$$S_{IK} = \begin{pmatrix} \frac{\partial Y_1}{\partial X_1} & \frac{\partial Y_1}{\partial X_2} & \cdots & \frac{\partial Y_1}{\partial X_I} \\ \frac{\partial Y_2}{\partial X_1} & \frac{\partial Y_2}{\partial X_2} & \cdots & \frac{\partial Y_2}{\partial X_I} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial Y_K}{\partial X_1} & \frac{\partial Y_K}{\partial X_2} & \cdots & \frac{\partial Y_K}{\partial X_I} \end{pmatrix}$$

En esta matriz Jacobiana de orden I x K cada fila constituye una entrada de la red y las columnas representan las salidas de la red:

$$S_{IK} = \frac{\partial y_K}{\partial x_I} = f'(net_k) \sum_{j=1}^L v_{JK} f'(net_j) w_{Ij} \quad [78]$$

Aplicando esta fórmula a una red con función de activación sigmoidal logística, la sensibilidad de la salida K con respecto a la entrada tomaría la siguiente expresión:

$$S_{IK} = y_K(1 - y_K) \sum_{j=1}^L v_{jk} b_j(1 - b_j) w_j \quad [79]$$

Los valores de la matriz Jacobiana dependen de la información aprendida por la red que se encuentra distribuida en las diferentes conexiones y de las funciones de transferencia entre las diferentes capas.

Para un patrón X_p se puede calcular la sensibilidad $S_{ik}(p)$ a partir de la esperanza matemática y de su desviación estándar, calculadas a través de las siguientes expresiones:

$$E(S_{IK}(p)) = \frac{\sum_{p=1}^P S_{IK}(p)}{P} \quad [80]$$

$$SD(S_{ik}(p)) = \sqrt{\frac{\sum_{p=1}^P (S_{ik}(p) - E(S_{ik}(p)))^2}{P-1}} \quad [81]$$

Se pueden citar numerosos autores que han utilizado este análisis de sensibilidad. Algunos de los más representativos son los siguientes: Hwang *et al.* (1991); Hashem (1992); Fu y Chen (1993); Zurada *et al.* (1994); Bishop (1995); Rzempoluck (1998) y ha sido aplicado en campos tan variados como el reconocimiento de imágenes (Takenaga *et al.*, 1991), la ingeniería (Guo y Uhrig, 1992; Bahbah y Grgis, 1999), la meteorología, Castellanos *et al.* (1994) y la medicina, Harrison *et al.* (1991); Engelbrecht *et al.* (1995); Rambhia *et al.* (1999).

En Gedeon (1997) se puede encontrar una comparación de los métodos basados en la magnitud de los pesos y el análisis de sensibilidad a través del cálculo de la matriz Jacobiana. El estudio refleja que los análisis basados en propiedades dinámicas son más fiables que los análisis estáticos.

Análisis de sensibilidad numérico

Este análisis supera algunas de las limitaciones que surgen de las restricciones encontradas en los métodos descritos en los apartados anteriores. En este sentido, el análisis basado en la magnitud de los pesos no ha demostrado ser sensible a la hora de ordenar las variables de entrada en función de su importancia sobre la salida (Garson, 1991a; Sarle, 2000). También hay que tener en cuenta que una gran variedad de estudios, sobre todo los realizados en las ciencias sociales utilizan variables discretas y/o nominales y no es muy correcto utilizar la técnica de los incrementos a las variables de entrada (Hunter, Kennedy, Henry y Ferguson, 2000). También hay que añadir que el método de la matriz Jacobiana se basa en que las variables de entrada son continuas Sarle (2000).

Todas estas condiciones restrictivas están superadas en el nuevo método llamado sensibilidad numérica (NSA, Numeric Sensitivity Analysis), descrito a continuación brevemente según las notaciones y fórmulas empleadas en Montaño y Palmer (2003).

Este método está basado en el cálculo de los pendientes que se forman entre las entradas y salidas sin necesidad de realizar ningún supuesto sobre la naturaleza de las variables y siempre respetando la estructura primitiva de los datos.

El índice NSA está basado en el cálculo numérico de la pendiente formada entre cada par de grupos consecutivos, g_r y g_{r+1} , de x_i sobre y_k mediante la siguiente expresión:

$$NSA_{ik}(g_r) \equiv \frac{\bar{y}_k(g_{r+1}) - \bar{y}_k(g_r)}{\bar{x}_i(g_{r+1}) - \bar{x}_i(g_r)} \quad [82]$$

donde

$\bar{x}_i(g_r)$ y $\bar{x}_i(g_{r+1})$ son las medias de la variable x_i correspondientes a los grupos g_r y g_{r+1} , respectivamente. $\bar{y}_k(g_r)$ e $\bar{y}_k(g_{r+1})$ son las medias de la variable y_k correspondientes a los grupos g_r y g_{r+1} , respectivamente.

El valor de la esperanza matemática del índice NSA o pendiente entre la variable de entrada i y la variable de salida k se calcula mediante:

$$E(NSA_{ik}(g_r)) = \sum_{r=1}^{G-1} NSA_{ik}(g_r) \cdot f(NSA_{ik}(g_r)) = \frac{\bar{y}_k(g_G) - \bar{y}_k(g_1)}{\bar{x}_i(g_G) - \bar{x}_i(g_1)} \quad [83]$$

donde

$$f(NSA_{ik}(g_r)) \equiv \frac{\bar{x}_i(g_{r+1}) - \bar{x}_i(g_r)}{\bar{x}_i(g_G) - \bar{x}_i(g_1)} \text{ representa la función de probabilidad del índice NSA.}$$

$\bar{x}_i(g_G)$ y $\bar{x}_i(g_1)$ son los valores promedio de la variable x_i para el último grupo g_G y el primer grupo g_1 , respectivamente, $\bar{y}_k(g_G)$ e $\bar{y}_k(g_1)$ son los valores promedio de la variable y_k para el grupo g_G y el grupo g_1 , respectivamente.

El valor de la esperanza matemática del índice NSA representa el efecto promedio que tiene un incremento de x_i sobre y_k . Al igual que en el caso de la matriz Jacobiana, cuanto mayor sea el valor absoluto de $E(NSA_{ik}(g_r))$, más importante es x_i con relación a y_k . El signo de $E(NSA_{ik}(g_r))$ indica si el cambio observado en y_k va en la misma dirección o no que el cambio provocado en x_i .

El cálculo de la desviación estándar del índice NSA, cuando las variables implicadas son de naturaleza continua, se puede realizar mediante:

$$SD(NSA_{ik}(g_r)) = \sqrt{E(NSA_{ik}^2(g_r)) - (E(NSA_{ik}(g_r)))^2} \quad [84]$$

El valor de la desviación estándar se debe interpretar como el grado de oscilaciones que ha sufrido la pendiente que se establece entre x_i e y_k , de manera que, a mayor valor de la desviación estándar, mayor comportamiento caótico o aleatorio tiene la función entre las dos variables implicadas.

Las conclusiones de los autores: Montaño y Palmer (2003) después de experimentar con cuatro matrices de datos y con el programa informático creado por ellos "Sensitivity Neural Network 1.0" son, en primer lugar y respecto al grado de generalidad, es que el método NSA describe mejor el efecto o la importancia de las variables, pero si éstas son cuantitativas los resultados son muy similares a los proporcionados por el método de la matriz Jacobiana. Si

las variables implicadas son discretas, los valores que proporciona son muy similares a los que aporta el índice de asociación Phi en el caso de variables binarias y el índice de asociación V en el caso de variables politómicas.

En el Método NSA las interpretaciones de los efectos de las variables son más sencillas porque el índice que proporciona está acotado en el intervalo [-1;1]. También proporciona un método gráfico que representa la función aprendida por la red entre una variable de entrada y la de salida lo que permite complementar el análisis numérico.

3.9. REDES NEURONALES Y MODELOS ESTADÍSTICOS CLÁSICOS

Es muy interesante observar cómo se relacionan los modelos de redes neuronales con los métodos estadísticos clásicos, dado que esta comparación ofrecerá una visión más completa de la importancia de las redes neuronales como excelentes clasificadores al mismo tiempo que nos motivará a utilizarlas en próximos estudios.

Una posible idea falsa acerca de las redes neuronales, que provoca diferencia entre ambos modelos, es que parece que la terminología utilizada no está en consonancia con la que se utiliza en la estadística clásica debido, fundamentalmente, a que las redes neuronales proceden del campo de la Inteligencia Artificial con aportaciones de una gran variedad de disciplinas. Sarle (1994) y Vicino (1998) en los estudios llevados a cabo desmienten estas diferencias y establecen muchas de las semejanzas que hay entre los modelos estadísticos clásicos y las diversas arquitecturas de las redes neuronales.

TABLA 5. EQUIVALENCIA EN LA TERMINOLOGÍA ESTADÍSTICA Y DE REDES NEURONALES

Terminología estadística	Terminología de redes neuronales
Observación	Patrón
Muestra	Datos de entrenamiento
Muestra de validación	Datos de validación, test
Variables explicativas	Variables de entrada
Variable de respuesta	Variable de salida
Modelo	Arquitectura
Residual	Error
Error aleatorio	Ruido
Estimación	Entrenamiento, aprendizaje
Interpolación	Generalización
Interacción	Conexión funcional
Coeficientes	Pesos de conexión
Constante	Peso umbral
Regresión y análisis discriminante	Aprendizaje supervisado o heteroasociación
Reducción de datos	Aprendizaje no supervisado o autoasociación
Análisis de clúster	Aprendizaje competitivo

Fuente: Sarle (1994) y Vicino (1998)

TABLA 6. EQUIVALENCIA ENTRE MODELOS ESTADÍSTICOS Y MODELOS DE RED NEURONAL

Modelo estadístico	Modelo de red neuronal
Regresión lineal múltiple	Perceptrón simple con función lineal
Regresión logística	Perceptrón simple con función logística
Función discriminante lineal	Perceptrón simple con función umbral
Regresión no lineal múltiple	Perceptrón multicapa con función lineal en la salida
Función discriminante no lineal	Perceptrón multicapa con función logística en la salida
Análisis de componentes principales	Regla de Oja
	Perceptrón multicapa autoasociativo
Análisis de clúster	Mapas autoorganizados de Kohonen
K vecinos más cercanos	Learning Vector Quantization (LVQ)
Regresión kernel	Funciones de Base Radial (RBF)

Fuente: Sarle (1994)

A la vista de estos dos cuadros se observa que la mayoría de redes neuronales aplicadas al análisis de datos son similares y, en algunos casos equivalentes, a modelos estadísticos muy conocidos y utilizados en la resolución de problemas de clasificación, regresión y de análisis de conglomerados.

Sí que podemos señalar una importante diferencia entre las redes neuronales y los modelos estadísticos en sus aspectos explicativos de las variables independientes sobre la variable dependiente, y es que, a pesar del análisis y los esfuerzos llevados a cabo para encontrar el efecto de la importancia de las variables del modelo en las redes neuronales, como se ha explicado en el epígrafe anterior, no parece que sea tan evidente como lo son en los modelos clásicos.

Podemos encontrar diversos estudios en la literatura de redes neuronales donde se manifiestan las equivalencias entre ambas perspectivas. Algunos de estos trabajos, los principales, se mencionan a continuación.

Sarle (2002) señala que un Perceptrón simple puede considerarse como un Modelo Lineal Generalizado. Según Biganzoli *et al.* (1998) el concepto de discrepancia en un MLG y el concepto de función de error en un Perceptrón también son equivalentes. La función que en general se intenta minimizar, en el caso del Perceptrón es la suma del error cuadrático:

$$E = \sum_{p=1}^P \frac{1}{2} \sum_{k=1}^M (d_{pk} - y_{pk})^2 \quad [85]$$

donde P hace referencia al número de patrones, M hace referencia al número de neuronas de salida, d_{pk} es la salida deseada para la neurona de salida k para el patrón p e y_{pk} es la salida obtenida por la red para la neurona de salida k para el patrón p .

Normalmente el método del Perceptrón estima los parámetros a través del criterio de los mínimos cuadrados, intentando minimizar la función E y el modelo MLG estima los parámetros por el método de máxima verosimilitud. Este método también se puede aplicar a un Perceptrón en tareas de clasificación si asumimos un error con distribución de Bernoulli: Hinton (1989), Spackman (1992), Van Ooyen y Nienhuis, (1992); Ohno-Machado y Musen (1997b); Biganzoli *et al.* (1998). En este caso, la función de error que se intenta

minimizar se denomina cross entropy (Bishop, 1995), cuya fórmula viene dada por la siguiente expresión:

$$E = - \sum_{p=1}^P \sum_{k=1}^M [d_{pk} \log y_{pk} + (1 - d_{pk}) \log(1 - y_{pk})] \quad [86]$$

Cuando se utiliza esta función de error se consigue que las salidas puedan ser interpretadas como probabilidades a posteriori, Bishop (1994).

Un modelo de regresión logística es similar a un Perceptrón simple con función de activación logística en la neurona de salida, Sarle (1994). La función logística puede ser vista como una generalización no lineal de los MLG, (Biganzoli *et al.*, 1998).

La Función Discriminante Lineal de Fisher es semejante a un Perceptrón simple con función de activación umbral en la neurona de salida, Kemp *et al.* (1997).

Una red MLP compuesta por tres capas, cuya capa oculta de neuronas utiliza una función de activación no lineal, en general la función logística, puede ser vista como una generalización no lineal de los MLG, Biganzoli *et al.* (1998).

Según qué tipo de función de activación se utilice en la capa de salida, el MLP se puede orientar a la predicción o a la clasificación. En el caso de utilizar la función identidad en la capa de salida, estaríamos ante un modelo de regresión no lineal Cheng y Titterington, (1994), Ripley (1994) y Flexer (1995). Si la función de activación en la capa de salida es la logística puede ser utilizada como una Función Discriminante no lineal Biganzoli *et al.* (1998).

3.10. OTRAS ARQUITECTURAS DE REDES NEURONALES

Otros modelos de Redes Neuronales, a partir de los cuales también se puede establecer una clara analogía con modelos estadísticos clásicos conocidos, son aquellas arquitecturas de redes entrenadas mediante la regla de Oja (1982 y 1989), las cuales permiten realizar Análisis de Componentes Principales (PCA). La red backpropagation autosupervisada o MLP autoasociativo es otro modelo de red que también ha sido aplicado al PCA y a la reducción de la dimensionalidad. Esta red fue utilizada inicialmente por Cottrell *et al.* (1989)

Las RNA también han sido utilizadas en el análisis de series temporales. El modelado de una serie temporal univariante se realiza tradicionalmente mediante una red perceptrón multicapa usando un número determinado de términos atrasados como entradas y las previsiones como salidas, Bishop (1995). Más recientemente se están utilizando las redes recurrentes que resultan de gran utilidad en la previsión de series temporales debido a que son capaces de aprender las relaciones temporales que se establecen entre patrones de entrada y salida, Elman (1990) y Montaño *et al.* (2011).

Existen otros estudios relacionados con la regresión de Cox y redes SOM que se pueden estudiar en Montaño (2005).

Desde el año 2006 ha habido una gran atención y desarrollo en las redes neuronales artificiales con varias capas ocultas y con otras arquitecturas diferentes a la del perceptrón multicapa, que dan soporte a los métodos conocidos como Deep Learning y que se estudian

en el módulo siete: Autoencoder, la máquina de Boltzmann, las redes de creencia profunda, las redes convolucionales y algunos prototipos de redes recurrentes.

3.11. LIBRERÍA R WEKA CON REDES NEURONALES

Aunque existen librerías en R como nnet, amore y neuralnet, en este epígrafe se ofrece el código de la librería RWeka porque permite que las variables explicativas de los modelos, tanto para clasificación como para regresión, sean tanto cuantitativas como cualitativas.

```
library(RWeka)
library(rJava)
library(caret)

# Para cargar otros algoritmos en RWeka descargarlos desde:
# https://sourceforge.net/projects/weka/files/weka-packages/
# El comando WOW nos presenta los parámetros con los que podemos configurar la red
# neuronal. Los principales son L (learning rate) y M (Momentum)
WOW("weka/classifiers/functions/MultilayerPerceptron")
-L <learning rate>
    Learning Rate for the backpropagation algorithm. (Value
    should be between 0 - 1, Default = 0.3).
Number of arguments: 1.
-M <momentum>
    Momentum Rate for the backpropagation algorithm. (Value
    should be between 0 - 1, Default = 0.2).
Number of arguments: 1.
-N <number of epochs>
    Number of epochs to train through. (Default = 500).
Number of arguments: 1.
-V <percentage size of validation set>
    Percentage size of validation set to use to terminate
    training (if this is non zero it can pre-empt num of
    epochs. (Value should be between 0 - 100, Default = 0).
Number of arguments: 1.
-S <seed>
    The value used to seed the random number generator (Value
    should be >= 0 and and a long, Default = 0).
Number of arguments: 1.
-E <threshold for number of consecutive errors>
    The consecutive number of errors allowed for validation
    testing before the network terminates. (Value should be >
    0, Default = 20).
Number of arguments: 1.
-G GUI will be opened. (Use this to bring up a GUI).
-A Autocreation of the network connections will NOT be done.
    (This will be ignored if -G is NOT set)
-B A NominalToBinary filter will NOT automatically be used.
    (Set this to not use a NominalToBinary filter).
-H <comma separated numbers for nodes on each layer>
    The hidden layers to be created for the network. (Value
    should be a list of comma separated Natural numbers or the
    letters 'a' = (attribs + classes) / 2, 'i' = attribs, 'o'
    = classes, 't' = attribs .+ classes) for wildcard values,
    Default = a).
Number of arguments: 1.
-C Normalizing a numeric class will NOT be done. (Set this
    to not normalize the class if it's numeric).
-I Normalizing the attributes will NOT be done. (Set this to
    not normalize the attributes).
-R Resetting the network will NOT be allowed. (Set this to
    not allow the network to reset).
-D Learning rate decay will occur. (Set this to cause the
    learning rate to decay).
```

```

-output-debug-info
    If set, classifier is run in debug mode and may output
    additional info to the console
-do-not-check-capabilities
    If set, classifier capabilities are not checked before
    classifier is built (use with caution).
-num-decimal-places
    The number of decimal places for the output of numbers in
    the model (default 2).
    Number of arguments: 1.
-batch-size
    The desired batch size for batch prediction (default 100).
    Number of arguments: 1.

# Instalación de paquetes y parámetros
WPM("install-package","C:/Trabajo/Big Data/Master/Modulo
5/Material//RBFNetwork1.0.8.zip")
WOW("weka/classifiers/functions/RBFNetwork")
-B <number>
    Set the number of clusters (basis functions) to generate.
    (default = 2).
    Number of arguments: 1.
-S <seed>
    Set the random seed to be used by K-means. (default = 1).
    Number of arguments: 1.
-R <ridge>
    Set the ridge value for the logistic or linear regression.
    Number of arguments: 1.
-M <number>
    Set the maximum number of iterations for the logistic
    regression. (default -1, until convergence).
    Number of arguments: 1.
-W <number>
    Set the minimum standard deviation for the clusters.
    (default 0.1).
    Number of arguments: 1.
-output-debug-info
    If set, classifier is run in debug mode and may output
    additional info to the console
-do-not-check-capabilities
    If set, classifier capabilities are not checked before
    classifier is built (use with caution).
-num-decimal-places
    The number of decimal places for the output of numbers in
    the model (default 2).
    Number of arguments: 1.
-batch-size
    The desired batch size for batch prediction (default 100).
    Number of arguments: 1.

WPM("install-package","C:/Trabajo/Big Data/Master/Modulo
5/Material/multiLayerPerceptrons1.0.10.zip")
WOW("weka/classifiers/functions/MLPClassifier")
-N <int>
    Number of hidden units (default is 2).
    Number of arguments: 1.
-R <double>
    Ridge factor for quadratic penalty on weights (default is
    0.01).
    Number of arguments: 1.
-O <double>
    Tolerance parameter for delta values (default is 1.0e-6).
    Number of arguments: 1.
-G      Use conjugate gradient descent (recommended for many
        attributes).
-P <int>

```

```

The size of the thread pool, for example, the number of
cores in the CPU. (default 1)
Number of arguments: 1.
-E <int>
    The number of threads to use, which should be >= size of
    thread pool. (default 1)
Number of arguments: 1.
-L <classname and parameters>
    The loss function to use. (default:
        weka.classifiers.functions.loss.SquaredError)
Number of arguments: 1.
-A <classname and parameters>
    The activation function to use. (default:
        weka.classifiers.functions.activation.ApproximateSigmoid)
Number of arguments: 1.
-S <num>
    Random number seed. (default 1)
Number of arguments: 1.
-output-debug-info
    If set, classifier is run in debug mode and may output
    additional info to the console
-do-not-check-capabilities
    If set, classifier capabilities are not checked before
    classifier is built (use with caution).
-num-decimal-places
    The number of decimal places for the output of numbers in
    the model (default 2).
Number of arguments: 1.
-batch-size
    The desired batch size for batch prediction (default 100).
Number of arguments: 1.

```

3.11.1. Análisis con base de datos German Credit

```

# Muestras de entrenamiento y test
set.seed(107)
inTrain <- createDataPartition(y = German$class, p = .8, list = FALSE)
train <- German[inTrain,]
test <- German[-inTrain,]

```

Perceptrón multicapa

```

mlp <- make_Weka_classifier("weka/classifiers/functions/MultilayerPerceptron")
m <- mlp(class~., data=train, control=Weka_control(L=0.1, M=0.2, N=500, H="3", V=0, S=0,
E=20))
modelo <- evaluate_Weka_classifier(m, newdata=test, numFolds=10, class=T)
print(modelo)
    === 10 Fold Cross Validation ===

    === Summary ===

    Correctly Classified Instances      158           79           %
    Incorrectly Classified Instances     42            21           %
    Kappa statistic                   0.4802
    Mean absolute error               0.2284
    Root mean squared error          0.4278
    Relative absolute error          54.2696 %
    Root relative squared error      93.3602 %
    Total Number of Instances         200


```

== Detailed Accuracy By Class ==

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0,583	0,121	0,673	0,583	0,625	0,483	0,806	0,606	bad
0,879	0,417	0,831	0,879	0,854	0,483	0,806	0,891	good
0,790	0,328	0,784	0,790	0,785	0,483	0,806	0,806	W. Avg

```
==== Confusion Matrix ====
```

a	b	<-- classified as
35	25	a = bad
17	123	b = good

3.11.2. Análisis con base de datos Boston Housing

```
# Recodificamos variable como factor
Boston$chas <- as.factor(Boston$chas)

# Muestras de entrenamiento y test
set.seed(107)
inTrain <- createDataPartition(y = Boston$medv, p = .8, list = FALSE)
train_b <- Boston[ inTrain,]
test_b <- Boston[-inTrain,]
```

Perceptrón multicapa

```
mlp <- make_Weka_classifier("weka/classifiers/functions/MultilayerPerceptron")
m <- mlp(medv~., data=train_b, control=Weka_control(L=0.1, M=0.2, N=500, H="3", V=0,
S=0, E=20))
modelo <- evaluate_Weka_classifier(m, newdata=test_b, numFolds=10, class=T)
print(modelo)
```

```
==== 10 Fold Cross Validation ====
```

```
==== Summary ====
```

Correlation coefficient	0.7713
Mean absolute error	3.421
Root mean squared error	5.6512
Relative absolute error	57.5205 %
Root relative squared error	67.1005 %
Total Number of Instances	99

Función de base radial

```
mlp <- make_Weka_classifier("weka/classifiers/functions/RBFNetwork")
m <- mlp(medv~., data=train_b, control=Weka_control())
modelo <- evaluate_Weka_classifier(m, newdata=test_b, numFolds=10, class=T)
print(modelo)
```

```
==== 10 Fold Cross Validation ====
```

```
==== Summary ====
```

Correlation coefficient	0.4378
Mean absolute error	5.118
Root mean squared error	7.5164
Relative absolute error	85.1946 %
Root relative squared error	89.2547 %
Total Number of Instances	99

TEMA 7: EXPLORACIÓN Y PREPROCESADO DE LOS DATOS

ÍNDICE

- 1. Introducción: Fases Metodológicas de un Proceso de Data Science**
- 2. Imputación de Datos Ausentes**
- 3. Filtrado y Eliminación de Valores Extremos u Outlier**
- 4. Transformación de la Base de Datos**
 - 4.1. Discretización de variables
- 5. Balanceo de las Clases**
- 6. Reducción de Variables o de la Dimensionalidad**
 - 6.1. Aproximación indirecta o filter
 - 6.2. Aproximación directa o wrapper (envoltura)
 - 6.3. Selección de variables con la librería caret de R
 - 6.4. Selección de variables con el programa WEKA

OBJETIVOS	PALABRAS CLAVE
<ul style="list-style-type: none"> - Definir las etapas a llevar a cabo en un proceso de Data Science. - Definir las operaciones más importantes de la exploración y el preprocesado de los datos. · Exponer una relación de métodos de imputación de datos ausentes. · Señalar algunas transformaciones útiles de la base de datos. · Recalcar la importancia de la reducción de la dimensionalidad y algunas de las técnicas más utilizadas. 	<ul style="list-style-type: none"> · Integración, limpieza y transformación de la información. - Tipos de datos faltantes: MAR, MCAR, MNAR - Valor extremo u outlier · Balanceo de muestras · Selección de variables: métodos Filter y Wrapper

1. INTRODUCCIÓN: FASES METODOLÓGICAS DE UN PROCESO DE DATA SCIENCE

Es bastante obvio que para obtener buenos resultados en los análisis estadísticos y de minería de datos se debe de partir de una base de datos con información consistente, completa, comprensible y limpia para que los análisis que se lleven a cabo puedan resultar útiles. Es necesario, por tanto, que los datos sean analizados con conciencia.

Las distintas tareas que conlleva la exploración y el preprocessado de los datos en un procedimiento de minería de datos, según varios autores, abarca un tiempo considerable, estimado entre el 70% y el 90%, del tiempo total destinado a un proyecto de minería de datos.

Las metodologías más importantes de minería de datos a la hora de llevar a cabo investigaciones son el proyecto CRISP-DM (Cross Industry Standard Process for Data Mining) enfocado en procesos industriales, y la metodología SEMMA (Sampling, Exploration, Modification, Modelization, Assessment). Si bien dichos proyectos cubren todas las fases que implica una correcta metodología, su uso es de carácter general en los problemas de data mining.

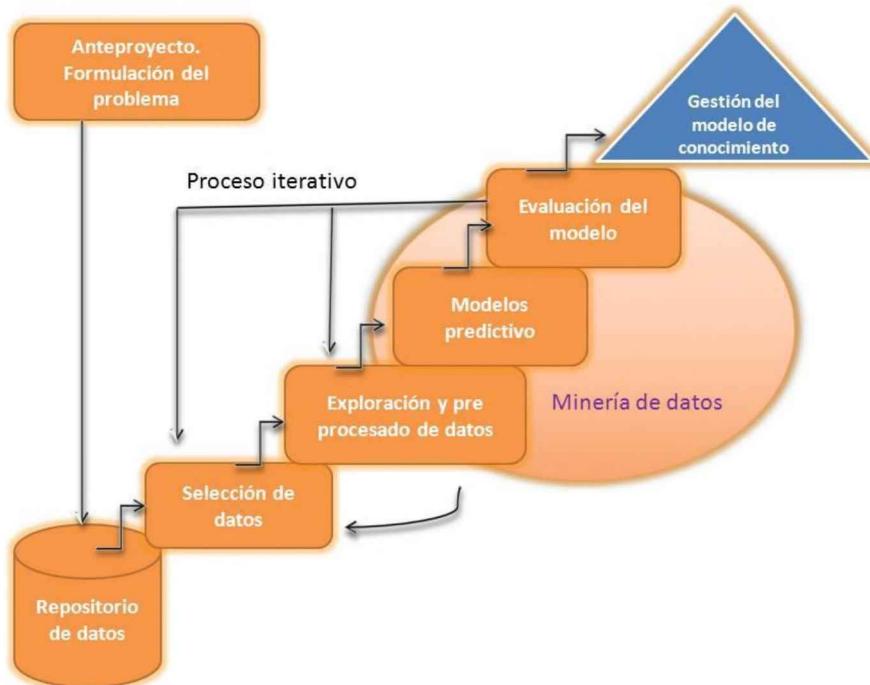
Por tanto, se entiende que las metodologías de resolución de problemas estadísticos o de minería de datos siguen un procedimiento común, pero cada caso en el que se trabaja dispone de una problemática especial que requiere, en algunas etapas del proceso, que se ponga especial atención en asuntos de vital importancia para la óptima resolución del problema planteado. En los datos de muchos trabajos de investigación se presentan algunas cuestiones claves que demandan de soluciones específicas y que se estudian en los epígrafes siguientes.

La fase inicial de cualquier proyecto de investigación estará basada normalmente en un anteproyecto donde se han especificado todas las fases del trabajo de investigación para conseguir los objetivos que se pretenden alcanzar. Una vez que se ha superado esta etapa inicial lo normal es ir allá donde se encuentran los datos para continuar con la investigación. Lo más normal es que la información, los datos, se encuentren en un repositorio de datos.

Los pasos metodológicos que integran el proceso de extracción de conocimiento útil, que se aplican en general en cualquier proceso de investigación, se pueden observar en la Figura 1. Estas cinco grandes fases se concretan en las siguientes:

- ✓ Formulación del problema. Integración de la información.
- ✓ Selección de datos, limpieza y transformación.
- ✓ Exploración y preprocessado de los datos.
- ✓ Análisis de los modelos predictivos.
- ✓ Gestión del modelo de conocimiento.

FIGURA 1 FASES METODOLÓGICAS DE UN PROCESO DE DATA SCIENCE



Fuente: Elaboración propia

A partir del momento en que se entrega el fichero inicial se empiezan a analizar las posibles variables que constituirán el fichero con el que se abordará siguiente fase.

La tarea del preprocesado de los datos tiene como objetivo obtener una vista minable, es decir, trata de quedarse con aquel conjunto de datos lo suficientemente libre de errores, ya filtrados, sin datos anómalos y cuyas variables sean lo más adecuadas al proceso de clasificación o de regresión que se está manejando.

Este primer paso inicial es de vital importancia, ya que implica realizar labores de limpieza de los datos, imputación, transformación, selección de variables y otro conjunto de tareas sin las cuales es imposible optimizar los métodos estadísticos.

Las tareas de limpieza y transformación relacionadas con la base de datos, en cualquier campo de investigación, hasta conseguir una base de datos minable son numerosas, pero las principales se pueden resumir en las siguientes:

- ✓ Imputación de datos ausentes.
- ✓ Filtrado y eliminación de valores anómalos o outlier.
- ✓ Transformación de la Base de datos.
- ✓ Balanceo de la base de datos.
- ✓ Reducción de variables o de la dimensionalidad.
- ✓ Discretización de variables.

2. IMPUTACIÓN DE DATOS AUSENTES

En la base de datos pueden existir, por varias razones, un conjunto significativo de valores ausentes, perdidos o faltantes que pueden ser reemplazados. Tanto en la fase de detección como en la de tratamiento de la información, es muy importante averiguar los motivos de los datos faltantes.

La aparición de datos faltantes (missing data) o la falta de respuesta es común en encuestas y, en general, en cualquier estudio o base de datos en cualquier campo del conocimiento humano.

Las principales razones para tratar los valores faltantes son:

- que el método de minería de datos no funcione bien con estos valores ausentes, o
- que se vayan a utilizar agregaciones de datos o variables y que estos datos ausentes no nos permitan realizarlas, o bien
- que el método utilizado nos elimine todo el ejemplo o instancia por no encontrar la existencia del dato.

En la literatura existente se considera que existen tres tipos de datos faltantes según sea la distribución de la aleatoriedad de los valores faltantes:

- **MAR (Missing At Random):** Esta situación se da cuando los datos no disponibles en una de las variables dependen de los valores observados en otra u otras variables del conjunto de datos. Por tanto, es una situación “manejable” estadísticamente. Por ejemplo, supongamos que el nivel de respuestas faltantes en una encuesta está relacionado con el nivel socio-económico del encuestado. Estimar a priori posibles pérdidas por este mecanismo debiera formar parte del protocolo en el cálculo del tamaño de la muestra en el caso de trabajar con encuestas.
- **MCAR (Missing Completely At random):** Cuando los datos faltantes son completamente aleatorios, es decir son independientes tanto de las variables observadas como de las no observadas. Estamos en el mejor escenario posible, siempre y cuando no existan demasiados datos faltantes. Diferentes investigadores señalan un umbral máximo del 5% del total para grandes conjuntos de datos.
- **MNAR (Missing Not At Random):** En este caso, los datos perdidos dependen o están relacionados con datos no observados. Por ejemplo, supóngase falta de respuesta en un cuestionario o pérdida durante el seguimiento. La falta de datos no aleatorios es un problema muy serio y sería aconsejable revisar el proceso de recopilación de datos e intentar comprender por qué falta la información.

Con tipos de datos faltantes MAR Y MCAR pueden obtenerse resultados no sesgados, mientras que con valores perdidos del tipo MNAR el sesgo o los sesgos que pueden introducirse son evidentes e invalidan en mayor o menor medida los resultados, con lo que sería adecuado tener cuidado y proceder a estudiarlos detenidamente con los expertos.

Las técnicas imputación de datos faltantes pueden clasificarse en dos grupos: las que utilizan técnicas de imputación simple y los métodos de imputación múltiple.

Respecto a las técnicas de imputación simple, se puede afirmar que son fáciles de implementar y que, en general, se ha demostrado que no experimentan importantes pérdidas de eficiencia en comparación con las técnicas de imputación múltiple.

En la imputación múltiple se realizan varias asignaciones de las observaciones faltantes y después se analizan los conjuntos de datos completados, los cuales se combinan para obtener una estimación final. Se consideran tres etapas en este tipo de imputación: fase de imputación, fase de análisis y fase de puesta en común.

Una vez que se han establecido las causas de los valores ausentes, podemos proceder a realizar alguna de las siguientes acciones

- ✓ Ignorarlos, es decir, no realizar ningún tratamiento, dado que la técnica que vamos a utilizar es consistente con valores ausentes, como por ejemplo los árboles de decisión o las redes bayesianas.
- ✓ Eliminar el atributo si la proporción de datos ausentes es elevada. En este caso se elimina toda la columna de la base de datos.
- ✓ Eliminar los ejemplos o instancias donde se encuentran los valores faltantes.
- ✓ Reemplazar el valor. Si esta es la acción elegida, existen diferentes procedimientos para sustituir los valores ausentes: imputación automática de casos perdidos a través de técnicas de predicción o clasificación, sustitución del valor por otro dato preservando la media o la varianza, por la moda si son valores nominales, etcétera. También podemos utilizar el algoritmo EM (Expectation Maximization) que se utiliza de forma tradicional para realizar esta operación de sustitución de valores.

En cuanto al tratamiento de los datos ausentes, podemos utilizar varias estrategias dependiendo de las necesidades del estudio:

- ✓ Utilizar la media, moda, mediana o cualquier otro estimador robusto, o un valor que preserve la desviación estándar de la distribución de la variable, etcétera.
- ✓ Cuando el número de registros es muy numeroso simplemente se pueden eliminar aquellos que tienen datos ausentes.
- ✓ Si son series temporales podemos calcular la media del valor anterior o posterior o realizar una predicción del valor a través de otros métodos (medias móviles, modelos autorregresivos y de medias móviles (ARIMA), etcétera)
- ✓ También se pueden aplicar otras técnicas avanzadas de estadística o técnicas heurísticas.

Existen muchas utilidades y lenguajes de programación que nos ayudan a la identificación y al tratamiento de los datos ausentes. En el software estadístico R existen un conjunto amplio de librerías que nos permiten llevar a cabo la imputación de datos. Recomendamos que se estudien las diferentes opciones que nos ofrecen las librerías incluidas en las task views del proyecto cran relacionadas con los datos perdidos: <https://cran.r-project.org/web/views/MissingData.html>.

TABLA 1. CRAN TASK VIEW

CRAN Task View: Missing Data	
Maintainer:	Julie Josse, Nicholas Tierney and Nathalie Vialaneix (r-miss-tastic team)
Contact:	r-miss-tastic at clementine.wf
Version:	2018-10-30
URL:	https://CRAN.R-project.org/view=MissingData
<p>Missing data are very frequently found in datasets. Base R provides a few options to handle them us package stats also contains the generic function <code>na.action</code> that extracts information of the NA action</p> <p>These basic options are complemented by many packages on CRAN, which we structure into main</p> <ul style="list-style-type: none"> • Exploration of missing data • Likelihood based approaches • Single imputation • Multiple imputation • Weighting methods • Specific types of data • Specific application fields 	
<p>If you think that we missed some important packages in this list, please contact the maintainer.</p>	
<p>Exploration of missing data</p> <ul style="list-style-type: none"> • <i>Manipulation of missing data</i> is implemented in the packages <code>sjmisc</code> and <code>sjlabelled</code>, <code>memisc</code> and <code>haven</code>. • <i>Missing data patterns</i> can be identified and explored using the packages <code>mi</code>, <code>dlookr</code>, <code>wrangle</code>, <code>missMDA</code> and <code>missForest</code>. • <i>Graphics that describe distributions and patterns of missing data</i> are implemented in <code>VIM</code> (<code>visNetwork</code>). • <i>Tests of the MAR assumption (versus the MCAR assumption)</i> are implemented in the function <code>testForMCAR</code> of <code>mi</code>. • <i>Evaluation with simulations</i> can be performed using the function <code>ampute</code> of <code>mice</code>. 	

Fuente: R

Entre las librerías más utilizadas en Data Science se encuentran las siguientes:

MissingDataGUI

Este paquete proporciona resúmenes numéricos y gráficos para los datos faltantes de variables categóricas y variables cuantitativas. Dispone de una buena variedad de métodos de imputación, incluyendo imputaciones univariantes como valores fijos o aleatorios, imputaciones multivariantes como los vecinos más cercanos e imputación múltiple a través de otros procedimientos, así como de imputaciones condicionadas a una variable categórica.

Amelia II

Esta librería imputa de forma múltiple datos perdidos tanto en estudios de sección cruzada como de series temporales. Amelia II implementa un algoritmo basado en bootstrap, lo que le dota de una rapidez de ejecución considerable. Esta utilidad también incluye diagnósticos útiles de ajuste de modelos de imputación múltiple. Esta librería dispone de una GUI atractiva.

VIM (Visualization and Imputation of Missing Values)

Contiene excelentes herramientas para la visualización de valores faltantes y/o imputados, que pueden ser utilizados para explorar los datos y la estructura de los valores faltantes y/o de los imputados. La calidad de la imputación se puede realizar de forma visual utilizando varios métodos gráficos univariantes, bivariantes y multivariantes.

MI. (Missing Data Imputation and Model Checking)

Este paquete proporciona funciones para la manipulación de datos, imputando valores perdidos dentro de un marco bayesiano. Esta librería está capacitada para realizar una imputación múltiple de los datos con valores faltantes. El algoritmo de forma iterativa extrae valores imputados de la distribución condicional para cada variable dados los valores observados y los valores imputados de las demás variables.

MICE (Multiple Imputation by Chained Equations)

Es la librería de R más utilizada en la imputación de datos. Realiza esta tarea utilizando Fully Conditionally Specification (FCS), implementado por el algoritmo MICE. En esta técnica cada variable puede tener su propio modelo de imputación. Dispone de métodos de imputación para datos continuos (predictive mean matching), para datos binarios (regresión logística), datos categóricos no ordenados (regresión logística multinomial) y datos categóricos ordenados (odds proporcional). Esta librería dispone de varios gráficos de diagnóstico que nos permiten examinar la calidad de las imputaciones.

TABLA 2. ALGUNAS TÉCNICAS DE IMPUTACIÓN INCORPORADAS EN MICE

Método	Descripción	Tipo de escala
pmm	Predictive mean matching	Numérico
norm	Regresión lineal bayesiana	Numérico
norm.nob	Regresión lineal no bayesiana	Numérico
norm.predict	Regresión lineal	Numérico
mean	Imputación por media incondicional	Numérico
logreg	Regresión logística	Factor, 2 niveles
polyreg	Modelo logístico multinomial	Factor, > 2 niveles
polr	Modelo logístico ordenado	Ordenado
lda	Análisis lineal discriminante	Factor
cart	Árboles de clasificación y regresión	Cualquiera
sample	Muestra aleatoria de los datos observados	Cualquiera

Fuente: R

Existen bastantes paquetes muy interesantes en R que nos facilitan la tarea de imputación y diagnóstico de los datos ausentes, al margen de los comentados anteriormente. Algunas de estas librerías son las siguientes: outliers, baboon, missForest, tsoutliers, Hmisc, récipes, etcétera (véase la página: <https://cran.r-project.org/>).

En las páginas que siguen se muestra el código en el lenguaje R, así como algunas de las salidas, para el descubrimiento y la imputación de valores faltantes.

Creamos un data frame con cuatro variables

```
a <- c(190, NA, 156, NA, 170, 159, 168, 162, 180, 164)
b <- c(75, 68, NA, 56, 67, 78, 65, 89, 64, 82)
c <- c("V", "V", "M", "V", "V", "M", NA, "V", "V", "M")
d <- c(90, NA, 78, 49, 62, 89, 78, NA, 83, 75)

datos <- as.data.frame(list(a,b,c,d))
colnames(datos) <- c("Altura", "Peso", "Sexo", "Edad")
```

	Altura	Peso	Sexo	Edad
1	190	75	V	90
2	NA	68	V	NA
3	156	NA	M	78
4	NA	56	V	49
5	170	67	V	62
6	159	78	M	89
7	168	65	NA	78
8	162	89	V	NA
9	180	64	V	83
10	164	82	M	75

```
summary(datos)
```

Altura	Peso	Sexo	Edad
Min. :156.0	Min. :56.00	M :3	Min. :49.00
1st Qu.:161.2	1st Qu.:65.00	V :6	1st Qu.:71.75
Median :166.0	Median :68.00	NA's:1	Median :78.00
Mean :168.6	Mean :71.56		Mean :75.50
3rd Qu.:172.5	3rd Qu.:78.00		3rd Qu.:84.50
Max. :190.0	Max. :89.00		Max. :90.00
NA's :2	NA's :1		NA's :2

Con la función `sapply` podemos sumar todos los valores faltantes para todas las variables de la base de datos.

```
sapply(datos, function(x) sum(is.na(x)))
Altura    Peso    Sexo   Edad
      2       1       1       2
```

Con los comandos de más abajo se pueden borrar todos los registros de la base de datos que contengan valores faltantes, aquí se presentan dos formas:

```
datos_sin <- na.omit(datos)
datos_sin <- datos[complete.cases(datos),]
```

	Altura	Peso	Sexo	Edad
1	190	75	V	90
5	170	67	V	62
6	159	78	M	89
9	180	64	V	83
10	164	82	M	75

Si a los valores faltantes los queremos reemplazar por el valor de la media o de la mediana se pueden utilizar los siguientes comandos:

```
datos[is.na(datos$Altura), "Altura"] <- round(mean(datos$Altura,na.rm=T),0)
datos[is.na(datos$Peso), "Peso"] <- round(median(datos$Peso,na.rm=T),0)
datos[is.na(datos$Edad), "Edad"] <- round(mean(datos$Edad,na.rm=T),0)
datos
```

	Altura	Peso	Sexo	Edad
1	190	75	V	90
2	169	68	V	76
3	156	68	M	78
4	169	56	V	49
5	170	67	V	62
6	159	78	M	89
7	168	65	<NA>	78
8	162	89	V	76
9	180	64	V	83
10	164	82	M	75

Para las variables no cuantitativas podemos asignar, por ejemplo, la moda a través de una función.

```
moda <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}
datos[is.na(datos$Sexo), "Sexo"] <- moda(datos$Sexo)
```

Imputación de valores faltantes a través de Random Forest

Utilizamos la base de datos iris disponible en el módulo base.

```
library(randomForest)
iris.na <- iris
apply(iris[,1:4],2,mean)
Sepal.Length Sepal.Width Petal.Length Petal.Width
5.843333    3.057333    3.758000    1.199333
```

Introducimos valores ausentes (NA)

```
for (i in 1:4) iris.na[sample(150,sample(50)),i] <- NA
iris.imputed <- rfImpute(Species ~ ., iris.na)
# Ponemos solo un decimal
iris.imputed <- round(iris.imputed[,2:5],1)
apply(iris.imputed[,1:4],2,mean)
Sepal.Length Sepal.Width Petal.Length Petal.Width
5.828667    3.018000    3.769333    1.187333
```

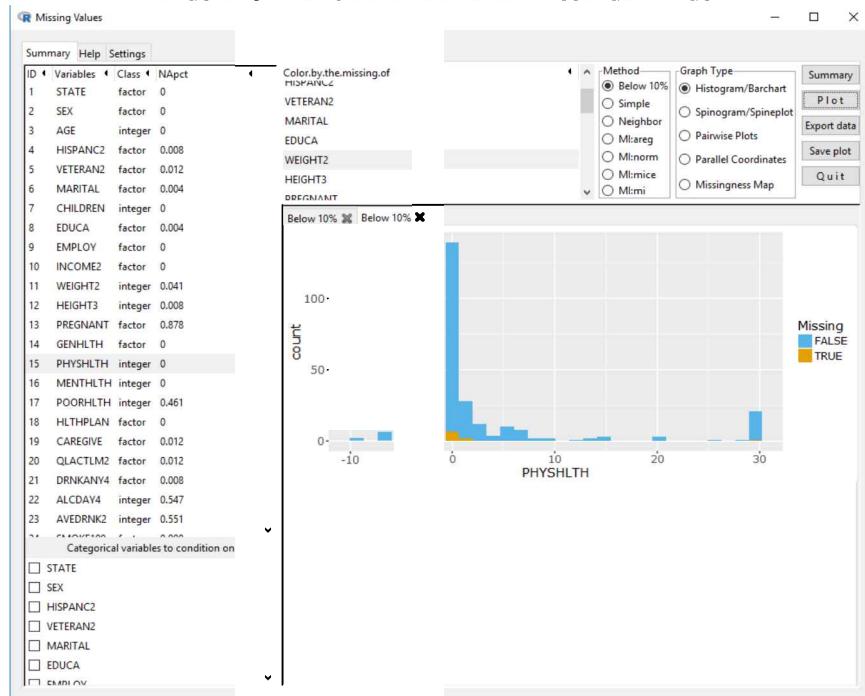
Si observamos las medias de las variables antes y después de realizar la imputación podemos concluir que esta forma de imputar es muy precisa.

Librería MissingDataGUI

```
install.packages("MissingDataGUI", dependencies=TRUE)
library(MissingDataGUI)
if (interactive()) {
  MissingDataGUI(brfss)
}
data(brfss)
str(brfss)
```

A través de la interface de usuario podemos obtener diferentes tablas informativas y una variedad de tipos de gráficos, como se muestra en la siguiente imagen, donde se ha marcado la opción de histograma para la variable PHYSHLTH del fichero cargado denominado brfss que está disponible en la librería.

FIGURA 5. MENÚ DE LA LIBRERÍA MISSINGDATAGUI



Fuente: R

A través de la opción del Menú Summary obtenemos un resumen de los valores perdidos para cada una de las variables.

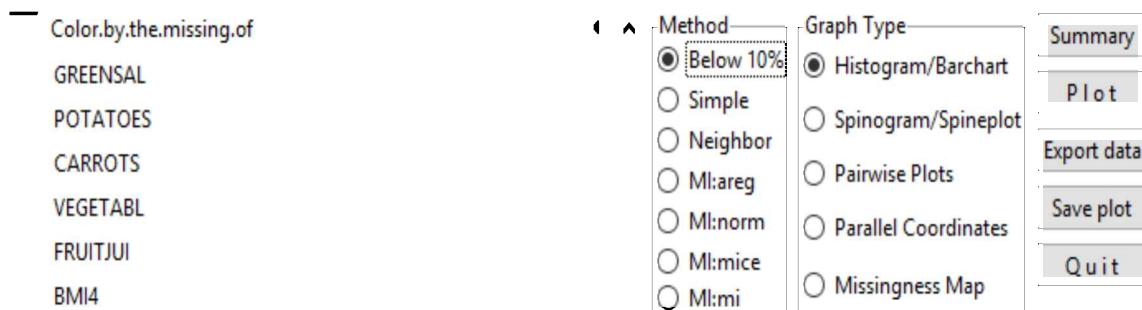
FIGURA 6. RESUMEN DE VALORES FALTANTES. LIBRERÍA MISSINGDATAGUI

Missing:		
14.24% of the numbers		
70.59% of variables		
100% of samples		
No_of_miss_by_case	No_of_Case	Percent
0	0	0
1	4	1.6
2	31	12.7
3	30	12.2
4	49	20
5	45	18.4
6	36	14.7
7	39	15.9
8	3	1.2
9	1	0.4
10	1	0.4
11	1	0.4
12	3	1.2
13	0	0
14	0	0
15	2	0.8
16	0	0
17	0	0
18	0	0
19	0	0
20	0	0
21	0	0
22	0	0
23	0	0
24	0	0
25	0	0
26	0	0
27	0	0
28	0	0

Fuente: R

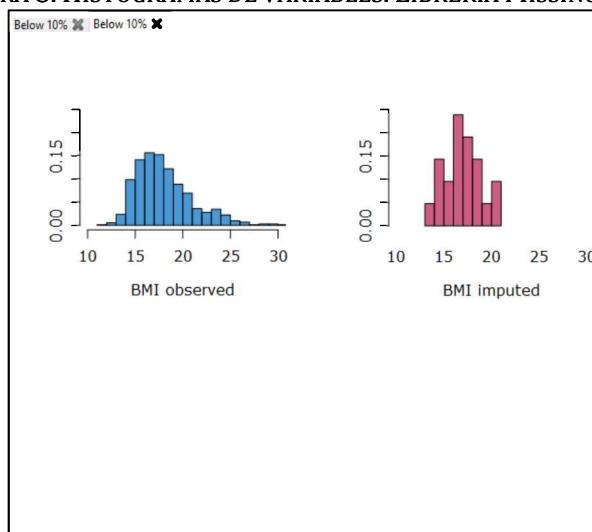
En cuanto a las técnicas de imputación múltiple de datos perdidos, esta librería las toma de otros paquetes de R: norm, mi y mice.

FIGURA 7. MÉTODOS DE IMPUTACIÓN. LIBRERÍA MISSINGDATAGUI



Fuente: R

FIGURA 8. HISTOGRAMAS DE VARIABLES. LIBRERÍA MISSINGDATAGUI



Fuente: R

Librería MICE

```
library(mice)
datos <- airquality
str(datos)

sapply(datos, function(x) sum(is.na(x)))
sapply(datos, function(x) sum(is.na(x)/length(x)*100))

# Si queremos chequear por filas y columnas utilizamos apply con una función
por <- function(x){
  sum(is.na(x))/length(x)*100
}
apply(datos,2,por)
apply(datos,1,por)

# Añadimos algunos datos missing más a algunas variables
datos[sample(1:nrow(datos),10),"Ozone"] <- NA
datos[sample(1:nrow(datos),15),"Solar.R"] <- NA
datos[sample(1:nrow(datos),10),"Wind"] <- NA
datos[sample(1:nrow(datos),20),"Temp"] <- NA

sapply(datos,function(x) sum(is.na(x)))
```

```
Ozone Solar.R     Wind    Temp   Month    Day
43      21      10      20      0       0
> md.pattern(datos)

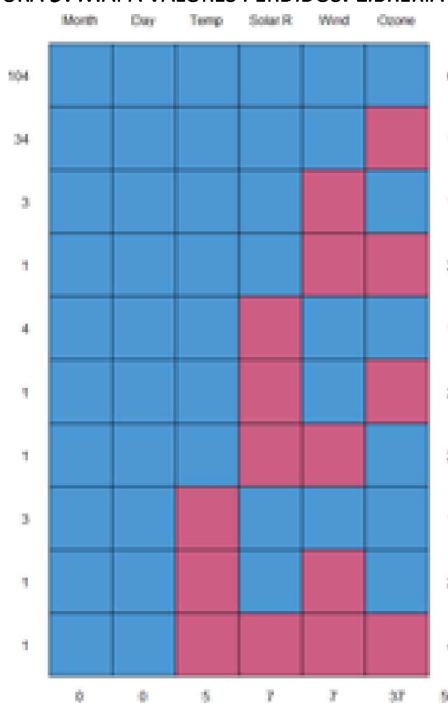
          Month Day Wind Temp Solar.R Ozone
81      1   1   1   1      1   1   0
29      1   1   1   1      1   0   1
10      1   1   1   1      0   1   1
6       1   1   1   1      0   0   2
8       1   1   1   0      1   1   1
6       1   1   1   0      1   0   2
3       1   1   1   0      0   1   2
5       1   1   0   1      1   1   1
1       1   1   0   1      1   0   2
1       1   1   0   1      0   1   2
2       1   1   0   0      1   1   2
1       1   1   0   0      0   0   4
0       0   10  20  21      43  94
```

La interpretación de la salida del comando `md.pattern` es sencilla. Existen 81 registros que no tienen ningún valor perdido en ninguna de sus variables. Hay 29 muestras que contiene un valor perdido en la variable Ozono. De igual forma se interpretan el resto de las filas del cuadro.

La librería VIM nos aporta un conjunto de gráficos que ayudan a la localización de forma visual a la interpretación de los datos perdidos.

```
library(VIM)
aggr_plot <- aggr(datos, col=c('navyblue','red'), numbers=TRUE, sortVars=TRUE,
                    labels=names(data), cex.axis=.7, gap=3, ylab=c("Histograma de valores
perdidos","Estructura"))
```

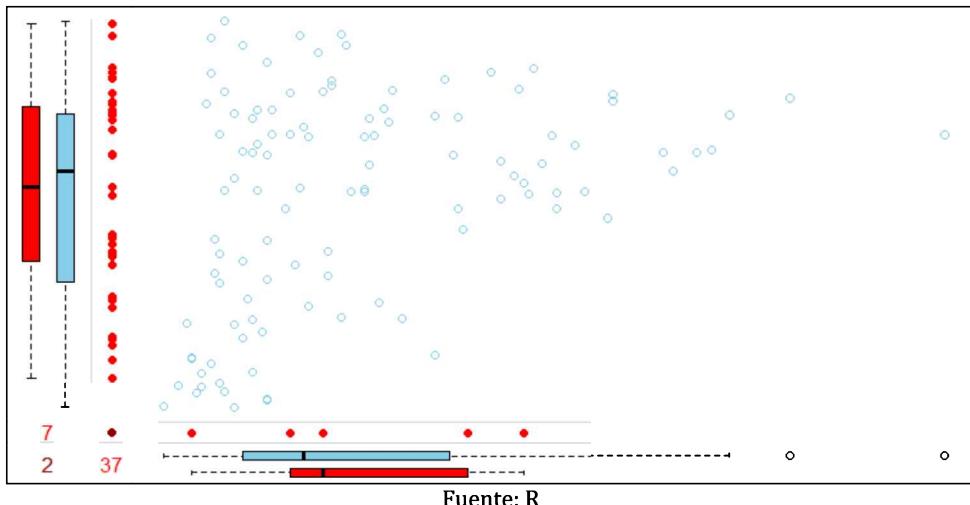
FIGURA 9. MAPA VALORES PERDIDOS. LIBRERÍA MICE



Fuente: R

```
marginplot(datos[c(1,2)])
```

FIGURA 10. GRÁFICO BOXPLOT VARIABLES 1 Y 2. LIBRERÍA MICE



```
tempData <- mice(datos, m=5, maxit=50, meth='pmm', seed='500')
```

El parámetro $m = 5$ se refiere al número de conjuntos de datos imputados. Cinco es el valor predeterminado.

`meth = 'pmm'` se refiere al método de imputación. En este caso, estamos utilizando la media predictiva como método de imputación. Se pueden usar otros métodos de imputación como se ha señalado anteriormente.

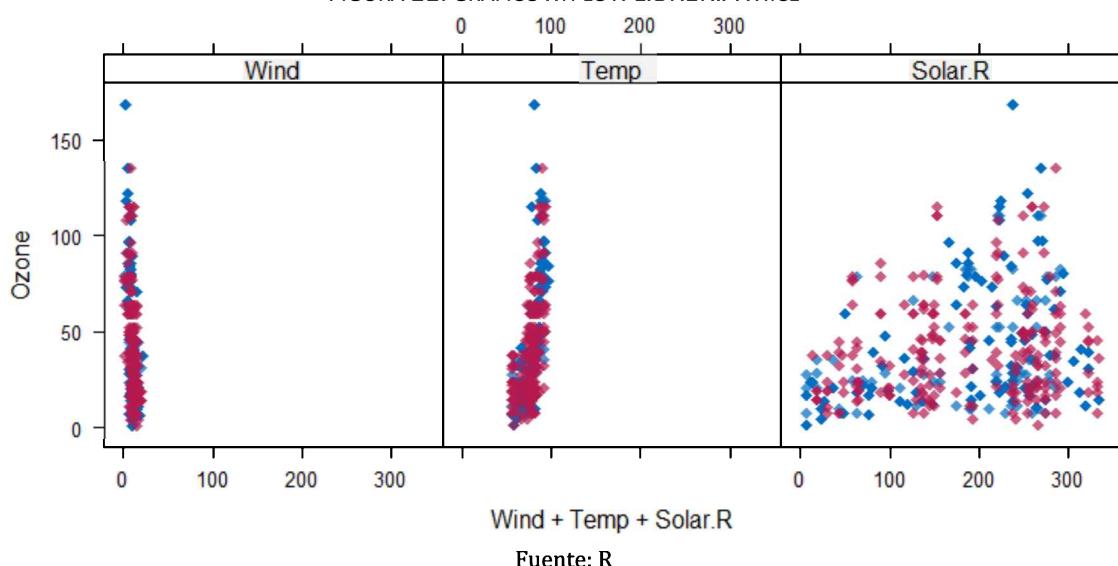
```

tempData$imp$Ozone
tempData$meth
completeData <- complete(tempData,1)
sapply(completeData,function(x) sum(is.na(x)))
Ozone Solar.R      Wind      Temp      Month      Day

          0         0         0         0         0         0
xyplot(tempData,Ozone ~ Wind +Temp+Solar.R, pch=18, cex=1)

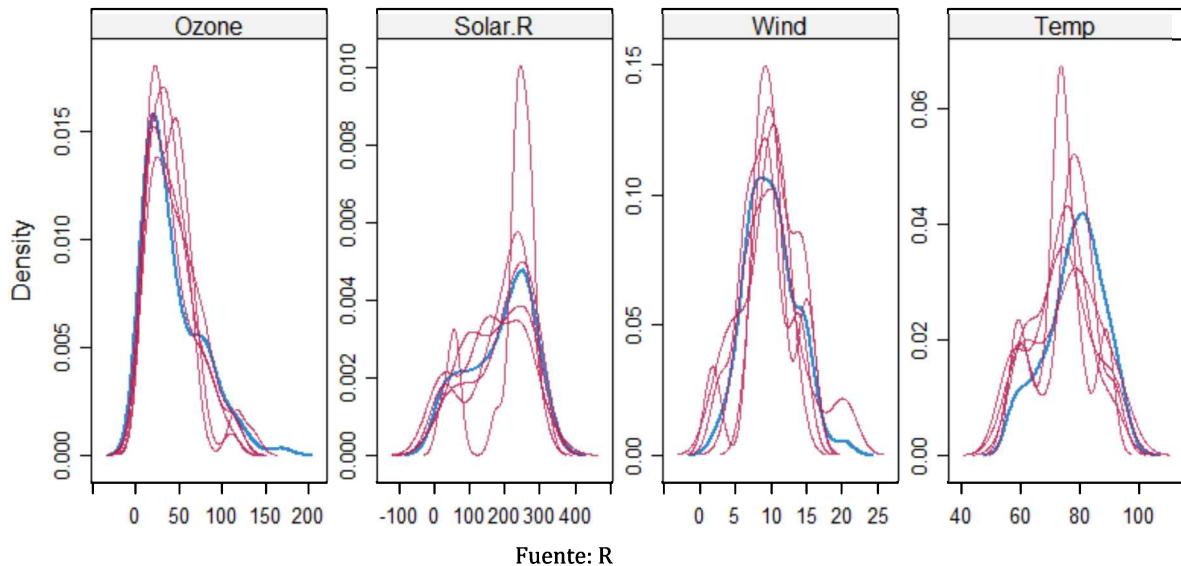
```

FIGURA 11. GRÁFICO XYPLOT. LIBRERÍA MICE



```
densityplot(tempData)
```

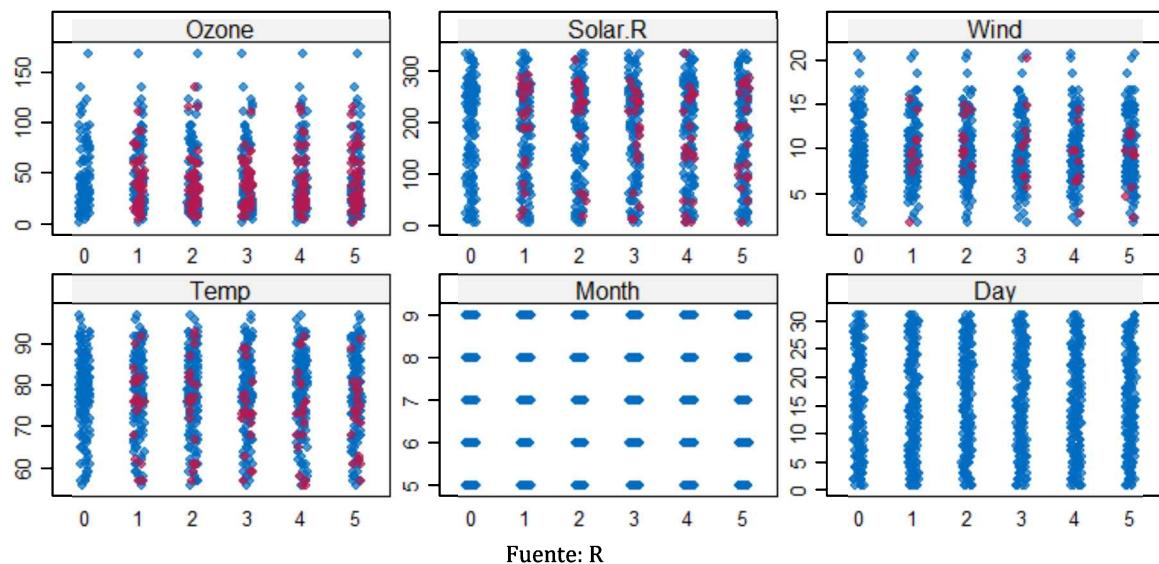
FIGURA 12. GRÁFICOS DE DENSIDAD. LIBRERÍA MICE



Fuente: R

```
stripplot(tempData, pch=20, cex=1.2)
```

FIGURA 13. STRIPLOT. LIBRERÍA MICE



Fuente: R

También podríamos plantearnos entre las diferentes alternativas del comando mice la imputación a través de un análisis de regresión:

```
modelFit1 <- with(tempData, lm(Temp~Ozone+Solar.R+Wind))
summary(pool(modelFit1))

tempData2 <- mice(datos, m=50, maxit=50, meth='pmm', seed=245435)
modelFit2 <- with(tempData2, lm(Temp~Ozone+Solar.R+Wind))
summary(pool(modelFit2))
```

3. FILTRADO Y ELIMINACIÓN DE VALORES EXTREMOS U OUTLIER

Los valores extremos, no usuales o erróneos son aquellos cuya disposición especial es extraña respecto al comportamiento general del conjunto. Estos datos se denominan de forma general como espurios o anómalos y su tratamiento resulta esencial, dado que afectan normalmente a las conclusiones y resultados finales de las investigaciones.

Las causas de encontrar valores extremos pueden provenir de:

- Sucesos anormales, que incluso pueden aportarnos información muy valiosa sobre el fenómeno de estudio.
- Errores de medición por aparatos mal calibrados.
- Datos mal introducidos (ya sea por defectos en la base de datos, errores de transmisión, o fallos de lectura, conversión o transformación de la información).

Si los datos proceden de encuestas, frecuentemente muchos valores anómalos son debidos a formularios incorrectos, a datos mal apuntados o a valores que no se han rellenado.

Son muchas las técnicas que se han propuesto para la detección de datos anómalos. Si las muestras de datos están generadas por poblaciones con distribuciones normales multivariantes las siguientes técnicas pueden resultar muy válidas:

- ✓ Técnicas de regresión para estimar el modelo que define los datos para determinar la desviación de los puntos frente al mismo.
- ✓ Utilización de histogramas o gráficos boxplots para detectarlos gráficamente.
- ✓ Mediante el uso de los autovalores de la muestra.
- ✓ Mediante el cálculo de la distancia de Mahalanobis.
- ✓ A través del Análisis de Componentes Principales, Proyección Pursuit.
- ✓ Utilizando análisis clúster.
- ✓ etc.

Si los datos proceden de poblaciones que siguen distribuciones de probabilidad no normales, las siguientes técnicas que se han propuesto son apropiadas:

- ✓ Proyección Sammon.
- ✓ Redes de mapas autoorganizados (SOM).
- ✓ Proyectores PCA no Lineales.
- ✓ Generative Topographic Maps (GTM).
- ✓ Otros proyectores No Lineales basados en redes neuronales.
- ✓ Otras técnicas de Visualización Multivariante.
- ✓ Coordenadas paralelas, dendogramas, curvas Andrews, iconos, Radviz, etc.
- ✓ Otros métodos.

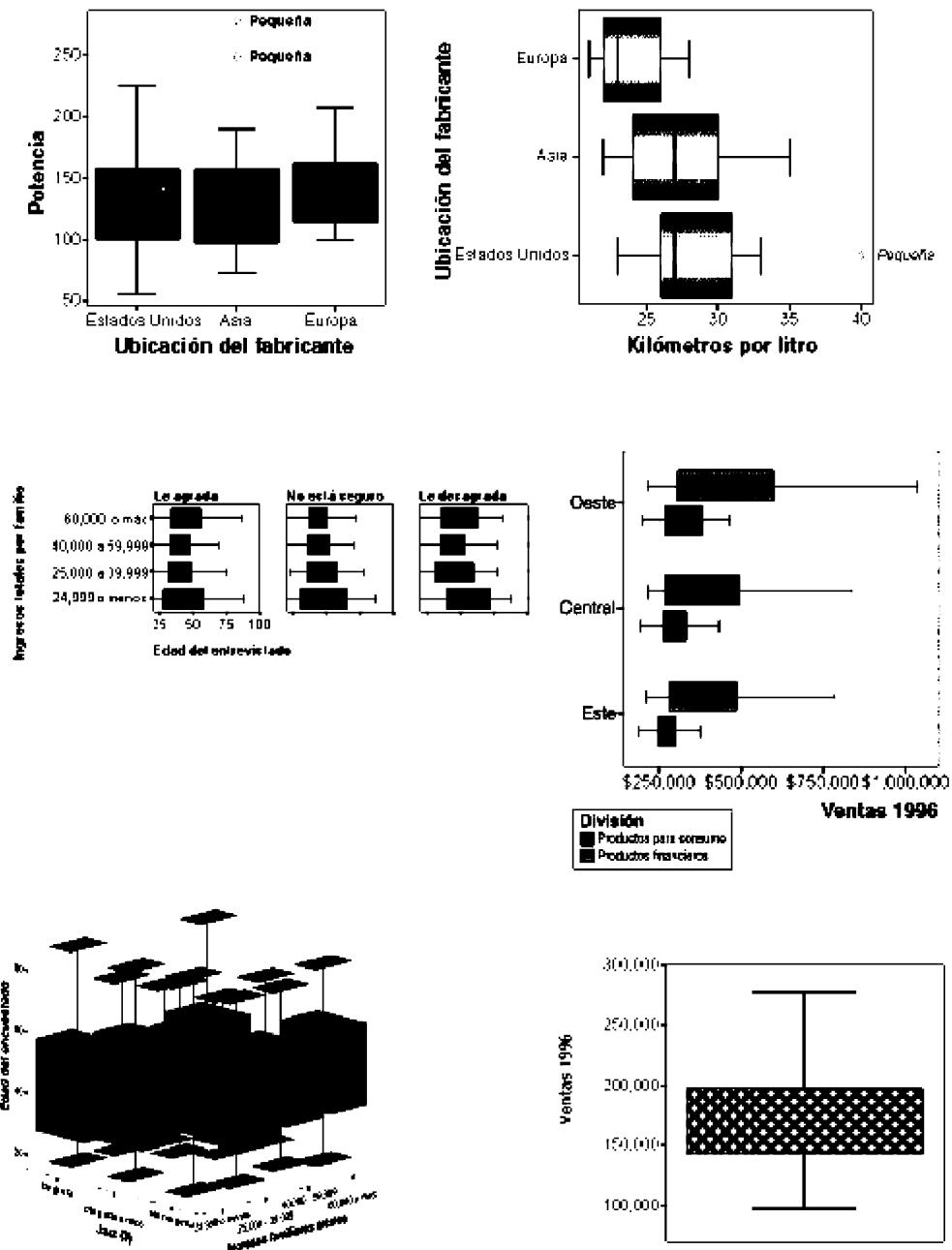
El apoyo gráfico resulta fundamental para la detección de outliers. En este sentido, podemos destacar los gráficos denominados diagramas de caja.

Los diagramas de caja son representaciones semigráficas de un conjunto de datos que muestran las características principales de la distribución y señalan los datos atípicos (outliers). La distribución y la dispersión de la variable se muestra a través de su mediana y sus cuartiles. La posición de los valores atípicos y de los valores extremos, si existen, aparecen identificados por símbolos especiales.

Los diagramas de caja simples muestran cajas para una sola variable de escala. Los casos pueden agruparse por los valores de una variable categórica. En los diagramas de caja agrupados aparece una agrupación de cajas para cada valor de otra variable categórica.

La utilización de la mediana en este tipo de gráficos, en vez de la media, como medida central de los datos, viene justificada porque la mediana es poco influenciable por los valores atípicos.

FIGURA 14. DIVERSOS TIPOS DE DIAGRAMAS DE CAJA



Entre los programas más interesantes para detectar, imputar y corregir los datos anómalos se encuentran algunas librerías de R: outliers, mvoutlier y tsoutliers.

El paquete outliers nos ofrece, entre otras funciones, procedimientos fáciles para llevar a cabo diferentes test estadísticos que nos permite comprobar si un valor es o no un valor

atípico: Chi cuadrado, Cocran, Dixon, Grubbs,... así como diferentes formas de tratamiento de esos valores anómalos.

El paquete mvoutlier incluye varios métodos estadísticos y gráficos que detectan los outliers multivariantes, definidos éstos como observaciones extrañas no por el valor que toman en una determinada variable, sino en el conjunto de aquellas. Se ven los casos extremos para una combinación de variables. Estos outliers multivariantes, pueden tener efectos aún más perjudiciales que los que se encuentran en una sola variable, dado que no solo distorsionan los valores de la media o de la varianza de la variable, sino que también afecta a las correlaciones entre las variables (Morillas y Díaz, 2007). Esta librería, en muchas de sus funciones multivariadas, utiliza la distancia Robusta de Mahalanobis basada en la determinación de covarianza mínima.

Estas son algunas de las principales funciones este paquete:

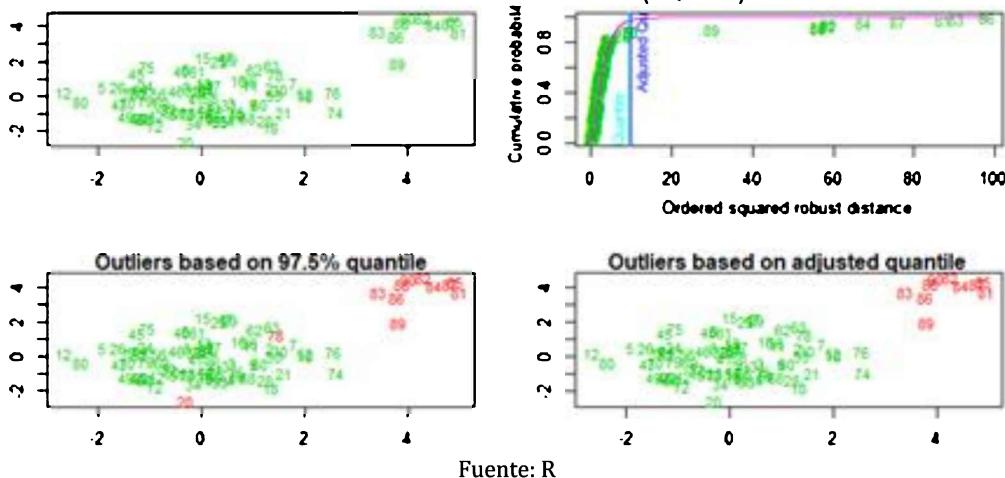
- ✓ aq.plot. Esta función presenta varios gráficos, entre ellos el del cuantil ajustado.
- ✓ arw. Estimador adaptativo ponderado. Los valores atípicos multivariados se definen de acuerdo con la diferencia entre la función de distribución de la distancia de Mahalanobis y la función de distribución teórica.
- ✓ chisq.plot. Gráfico Chi-Cuadrado
- ✓ corr.plot. Presenta gráfico y datos de correlación robusta bivariada versus correlación clásica.
- ✓ plot.mvoutlierCoDa. Ofrece un gráfico para la interpretación de los valores atípicos multivariados de Coda.

Para observar la potencia de esta librería del programa R se presentan algunos resultados con la base de datos que tiene incorporada la librería y con algunos de los comandos del manual de la misma librería.

```
set.seed(134)
x <- cbind(rnorm(80), rnorm(80), rnorm(80))
y <- cbind(rnorm(10, 5, 1), rnorm(10, 5, 1), rnorm(10, 5, 1))
z <- rbind(x,y)

aq.plot(z, alpha=0.1)
Projection to the first and second robust principal components.
Proportion of total variation (explained variance): 0.7402121
```

FIGURA 15. IMÁGENES DE LA SALIDA (AQ.PLOT)



Fuente: R

```

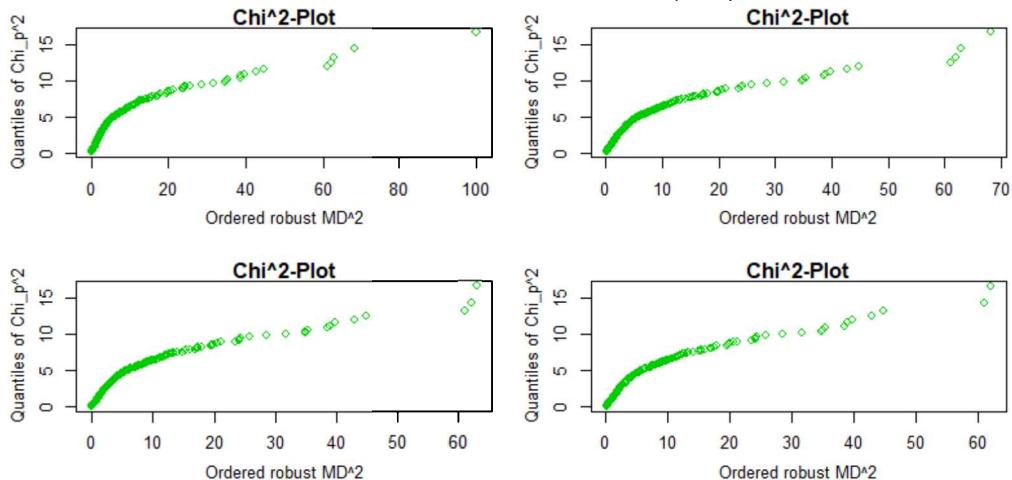
x <- cbind(rnorm(100), rnorm(100))
arw(x, apply(x,2,mean), cov(x))
$`m`
[1] -0.10445030 -0.07334337

$c
[,1]      [,2]
[1,] 1.1423630 -0.1243862
[2,] -0.1243862  0.9518827

data(humus)
res <- chisq.plot(log(humus[,c("Co","Cu","Ni")]))
res$outliers # these are the potential outliers

```

FIGURA 16. IMÁGENES DE LA SALIDA (ARW)



Fuente: R

```

x <- cbind(rnorm(100), rnorm(100))
y <- cbind(rnorm(10, 3, 1), rnorm(10, 3, 1))
z <- rbind(x,y)

# execute:
corr.plot(z[,1], z[,2])
$`cor.cla` 

[1] 0.2659496

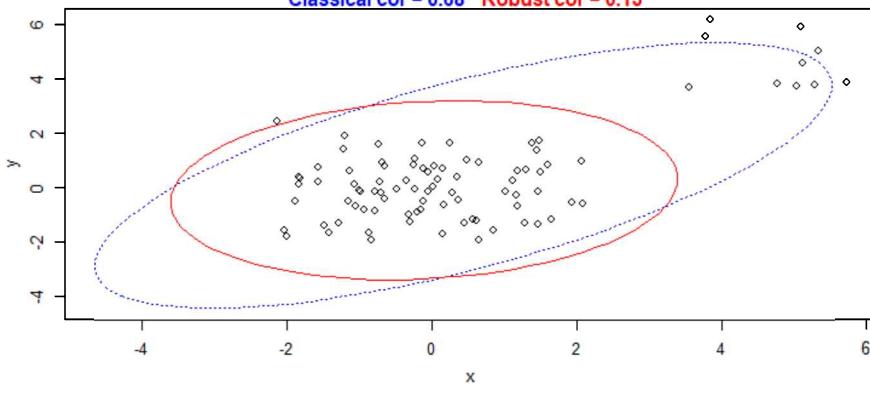
$`cor.rob` 

[1] 0.03057642

```

FIGURA 17. CORRELACIÓN CLÁSICA Y ROBUSTA

Classical cor = 0.68 Robust cor = 0.13



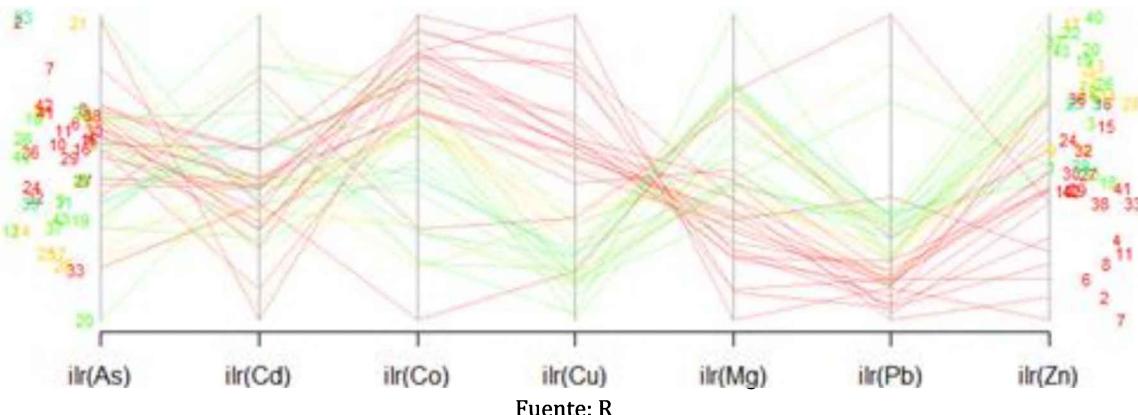
Fuente: R

```

data(humus)
el=c("As", "Cd", "Co", "Cu", "Mg", "Pb", "Zn")
dsel <- humus[,el]
data(kola.background) # contains different information (coast, borders, etc.)
coo <- rbind(kola.background$coast,kola.background$boundary,kola.background$borders)
XY <- humus[,c("XCOO", "YCOO")]
set.seed(123)
res <- mvoutlier.CoDa(dsel)
par(ask=TRUE)
plot(res,onlyout=TRUE,bw=FALSE,which="parallel",symb=TRUE,symbtxt=TRUE,transp=0.3)

```

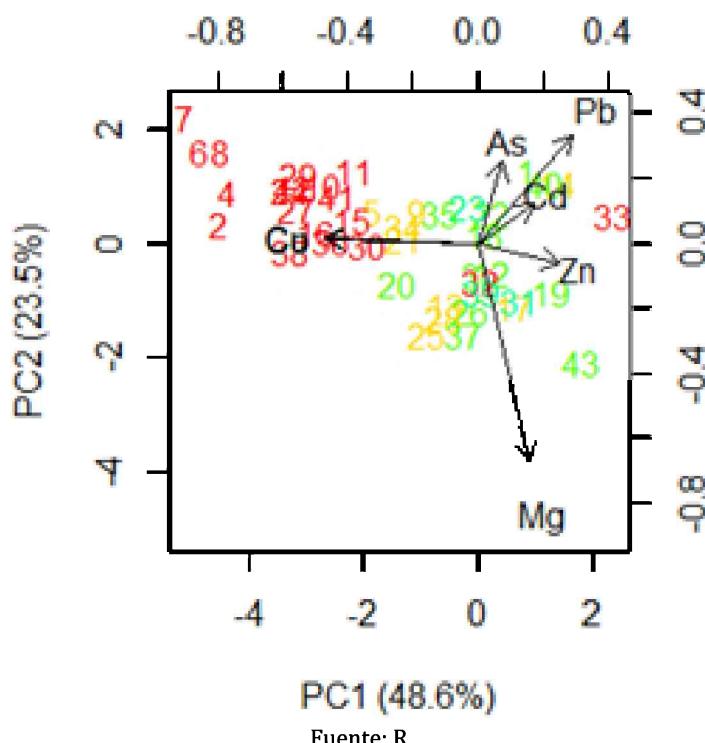
FIGURA 18. GRÁFICO COORDENADAS PARALELAS



Fuente: R

```
plot(res,onlyout=TRUE,which="biplot",bw=FALSE,symb=TRUE,symbtxt=TRUE)
```

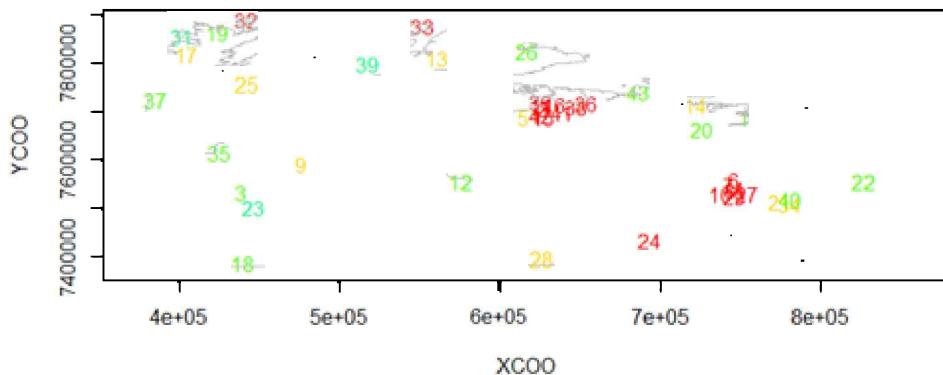
FIGURA 19. GRÁFICO BIPILOT DE COMPONENTES PRINCIPALES



Fuente: R

```
plot(res,coord=XY,map=coo,onlyout=TRUE,which="map",bw=FALSE,symb=TRUE,symbtxt=TRUE)
```

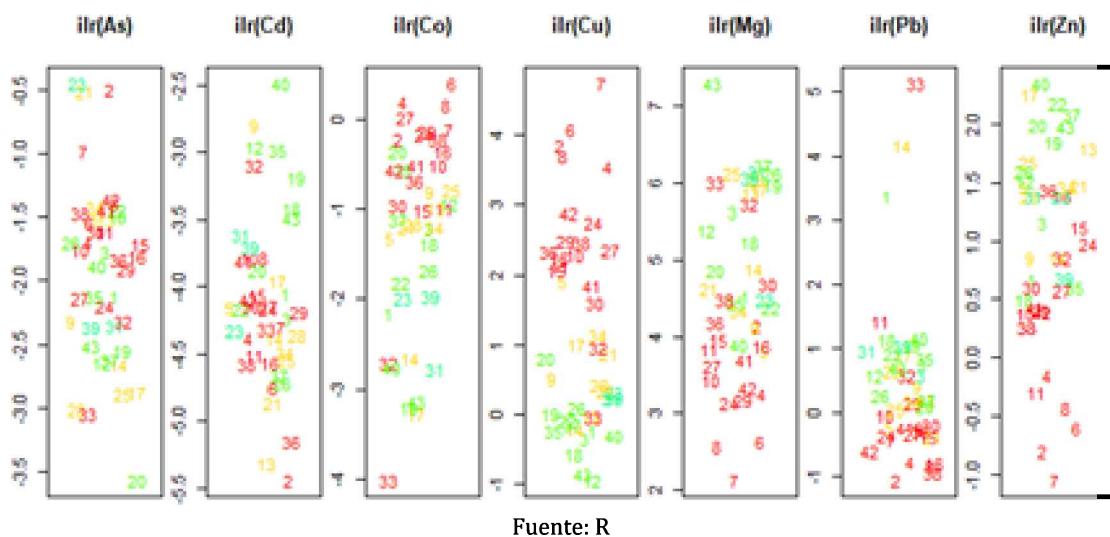
FIGURA 20. GRÁFICO MAP



Fuente: R

```
plot(res,onlyout=TRUE,which="uni",symb=TRUE,symbtxt=TRUE)
```

FIGURA 21. GRÁFICOS UNIVARIANTES



Fuente: R

4. TRANSFORMACIÓN DE LA BASE DE DATOS

En la transformación de la base de datos se incluyen aquellas operaciones que modifican los atributos o bien se derivan nuevos atributos, y aquellas otras que transforman el tipo de datos a través de la discretización o numerización, o cambian el rango a través del escalado de las variables.

Estandarizar es transformar una variable aleatoria que tiene alguna distribución en una nueva variable aleatoria con distribución normal o aproximadamente normal, restando a todos los datos su media y dividiéndolos por su desviación típica. La nueva distribución tendrá media cero y desviación típica igual a uno.

$$x' = \frac{x - \bar{x}}{\sigma} \quad [3]$$

También se puede normalizar una variable de tal forma que el nuevo rango de valores se encuentre entre cero y uno. Lo que se consigue aplicando la siguiente fórmula:

$$x' = \frac{x - \text{mínimo}}{\text{máximo} - \text{mínimo}} \quad [4]$$

El escalado de variables se lleva a cabo al dividir todos los valores de la variable por su valor máximo.

$$x' = \frac{x}{\text{máximo}} \quad [5]$$

4.1. DISCRETIZACIÓN DE VARIABLES

En muchas ocasiones hay que convertir los valores numéricos en nominales (discretización) o un valor nominal en numérico (numerización). En general, la discretización se realiza cuando el error de medida es grande o queremos expresar las conclusiones en ciertos umbrales significativos. También debemos discretizar cuando existen escalas diferentes. Otras veces crear intervalos es imperativo dado que algunos procedimientos necesitan que las variables numéricas sean transformadas. Es muy importante escoger el mejor método para discretizar las variables numéricas en la utilización de algunos métodos de minería de datos como las redes bayesianas o los árboles de decisión.

Discretización de variables cuantitativas:

En algunos de los procedimientos utilizados en minería de datos, como es el caso del modelo multinomial de redes bayesianas, los árboles de decisión o en otros procedimientos de minería de datos como las reglas de asociación es necesario transformar los valores de las variables numéricas en conjuntos de variables nominales ordenadas. Transformamos los valores en otro conjunto de intervalos disjuntos que cubren completamente el dominio de la variable continua.

Los diferentes métodos que existen se pueden clasificar en base a cuatro grandes criterios:

- **Locales o globales.** Los métodos globales emplean toda la información de la variable continua en el proceso de discretización, mientras que los locales solo utilizan un subconjunto de valores. Los métodos locales se relacionan con la discretización dinámica. Los métodos globales son más eficientes, pues solamente se usa una discretización en todo el proceso de data mining, pero los métodos locales podrían provocar el descubrimiento de puntos de corte más útiles.
- **Métodos supervisados o no supervisados.** Cuando se trabaja con datos que están divididos en clases, los métodos de discretización supervisados utilizan la información contenida en la clase de cara a seleccionar los puntos de corte en la discretización. Estos métodos pueden estar basados en el error (aplicando un clasificador a los datos transformados), en la entropía o en estadísticas. Si no utilizamos la información de la clase, estamos ante un método no supervisado.
- **Métodos bottom up (agrupamiento) o top-down (separación).** Los métodos de agrupamiento comienzan el proceso con una lista completa de todos los valores de la variable como puntos de corte. El proceso de discretización se forma mientras se van eliminando puntos, “agrupando” los intervalos mientras progresa la discretización. Los métodos de separación comienzan con una lista vacía y se van

agregando puntos a la lista, “separando” los intervalos a medida que la discretización progrésa.

- **Métodos estáticos o dinámicos.** Algunos algoritmos de clasificación, como por ejemplo los árboles de decisión CRT y C5.0, tienen incorporados mecanismos para la discretización de los atributos continuos durante el proceso de clasificación (discretización dinámica). La discretización estática constituye un paso más en el preprocessamiento de los datos., donde los atributos continuos son previamente discretizados antes de la tarea de clasificación.

Los cuatro métodos más utilizados son los siguientes:

- Método de igual longitud.
- Método de igual frecuencia.
- Discretización por mínima entropía.
- Método Chi-Merge.

El **método de igual longitud** es el más simple de todos los métodos no supervisados. Los intervalos se construyen calculando el máximo y el mínimo de la variable y se divide por el rango en k intervalos de longitud.

El **método de igual frecuencia** también es no supervisado. Este procedimiento construye intervalos que contienen el mismo número de observaciones.

El **método Chi-Merge** fue diseñado por Keber (1992) y es un tipo de discretización bottom-up supervisada, ya que utiliza la información contenida en la clase, donde las frecuencias relativas de las clases deben ser consistentes entre intervalos. Como su propio nombre indica, en el proceso de discretización se utiliza la prueba estadística χ^2 para discriminar si dos intervalos adyacentes son independientes. Si este es el caso, se juntan los intervalos y se separan, en caso contrario.

La primera etapa de este algoritmo consiste en ordenar las observaciones que se pretenden discretizar y empezar el proceso colocando cada observación en un intervalo. La segunda etapa abarca dos pasos que continuarán hasta que el proceso finalice. En el primer paso se calcula el valor del estadístico χ^2 para cada par de intervalos y en el segundo paso se agrupan los intervalos con menor valor del estadístico. Este procedimiento continuará hasta que todos los valores de la χ^2 sean mayores que cierto umbral asociado al nivel de significación de la prueba estadística.

Respecto a los **métodos que utilizan el concepto de mínima entropía** existen en la literatura numerosos procedimientos que aplican este criterio para discretizar atributos continuos. Entre los primeros autores podemos citar a Chiu *et al.* (1990) y a Wong y Chiu (1987). Entre los métodos más utilizados destacan las propuestas de Catlett (1991) y Fayyard e Irani (1993) que recurren a la entropía de la clase para establecer los límites de los intervalos (cortes) en los que se dividirá el rango de un atributo continuo. En este método se seleccionan los puntos de corte de forma recursiva mediante un algoritmo de minimización de la entropía, usando el criterio de Longitud de Descripción Mínima desarrollado por Suzuki (1996). Este es el procedimiento de discretización que utilizan por defecto algunos softwares de minería de datos como WEKA.

La entropía es la medida del desorden de un sistema mediante la incertidumbre existente ante un conjunto de casos, del cual se espera uno solo. Sea D un conjunto de datos

etiquetados con clases del conjunto $C = (C_1, \dots, C_k)$ y $frec(C_i, D)$ el número de ejemplos de D con clase C_i . Entonces se define la entropía del conjunto D de la forma siguiente:

$$Ent(D) = -\sum_{i=1}^k \frac{frec(C_i, D)}{|D|} x \log_2 \left(\frac{frec(C_i, D)}{|D|} \right) \quad [6]$$

donde $\frac{frec(C_i, D)}{|D|}$ representa la probabilidad de que se dé un ejemplo con clase C_i , y $\log_2 \left(\frac{frec(C_i, D)}{|D|} \right)$ identifica la información que transmite un ejemplo de la clase C_i . La entropía es máxima si todas las clases representan la misma proporción.

Utilizando la notación de Fayyad e Irani, si el conjunto de datos lo representamos como S , un atributo como A , y el corte como T , la entropía de clase de los intervalos S_1 y S_2 inducidos por T es calculada con la siguiente expresión:

$$E(A, T; S) = \frac{|S_1|}{|S|} x Ent(S_1) + \frac{|S_2|}{|S|} Ent(S_2) \quad [7]$$

Donde $|S|$, $|S_1|$ y $|S_2|$ indican el número de instancias de las particiones de cada conjunto y $Ent(\cdot)$ es la entropía, la cual se calcula a través de la ecuación 6. Así, para cada atributo se selecciona el corte T entre todas las posibles particiones que minimiza $E(A, T; S)$.

Una vez establecido el corte, se aplica recursivamente esta heurística a cada una de las dos particiones resultantes (S_1 y S_2) hasta que se satisface un criterio de parada.

La diferencia entre el algoritmo de Catlett y la propuesta de Fayyad e Irani radica en ese criterio. Mientras el método de Catlett se detiene cuando el número de ejemplos en un intervalo es suficientemente pequeño o el número de intervalos alcanza un máximo, Fayyad e Irani usan el principio de Longitud de Descripción Mínima como condición de parada, deteniendo el algoritmo si y solo si:

$$Ganancia(A, T; S) < \frac{\log_2(|S|-1)}{|S|} + \frac{\Delta(A, T; S)}{|S|} \quad [8]$$

donde

$$Ganancia(A, T; S) = Ent(S) - E(A, T; S) \quad \text{y}$$

$$\Delta(A, T; S)|S| = \log_2(3^k - 2) - (k \cdot ent(S) - k_1 \cdot Ent(S_1) - k_2 \cdot Ent(S_2))$$

y k , k_1 y k_2 son el número de clases distintas de S , S_1 y S_2 respectivamente.

Este criterio puede producir intervalos muy desiguales para un mismo atributo, ya que, una vez establecido un corte, la evaluación de los dos subespacios resultantes es independiente. De este modo, zonas del espacio que presenten una baja entropía serán divididas muy pocas veces, dando intervalos relativamente grandes, mientras que, en otras zonas con alta entropía, los cortes serán mucho más próximos.

Una función de R para crear intervalos

```
# Función para crear intervalos de las variables explicativas
# Argumentos: variable y número de niveles a crear

factorizarVariable <- function( variable, cortes ){
  resultado <- NULL
  breaks <- NULL
  levels <- NULL
  etiquetas <- NULL

  for (i in 0:cortes){
    if (i == 0) breaks <- min(variable)-1
    else if (i < cortes){
      quantil = i / cortes
      breaks <- c(breaks, quantile(variable, quantil))
    }
    else if (i == cortes) breaks <- c(breaks, max(variable)+1)
  }

  for (j in 1:cortes){
    if (j == 1){
      quantil = 1 / cortes
      etiquetas <- paste("Menos de", quantile(variable, quantil))
    }
    else if (j < cortes){
      quantila = (j-1) / cortes
      quantilb = j / cortes
      etiquetas <- c(etiquetas, paste("De", quantile(variable, quantila), "a",
      quantile(variable, quantilb)))
    }
    else if (j == cortes){
      quantil = (j-1) / cortes
      etiquetas <- c(etiquetas, paste("Más de", quantile(variable, quantil)))
    }
  }

  for (k in 1:cortes) levels <- c(levels, k)

  resultado <- cut(variable,
                     breaks = breaks,
                     labels = etiquetas,
                     levels = levels)
  return ( resultado )
}
```

5. BALANCEO DE LAS CLASES

El tamaño de la muestra juega un papel determinante en la bondad de los modelos de clasificación. Cuando el desbalanceo es considerable descubrir regularidades inherentes a la clase minoritaria se convierte en una tarea ardua y de poca fiabilidad. Japkowicz y Stephen (2002) concluyen que si los dominios son separables linealmente los modelos no son sensibles al problema del desequilibrio de las clases.

El tamaño de la muestra juega un papel determinante en la bondad de los modelos de clasificación y regresión. Se dice que una clase está desbalanceada, cuando el número de frecuencias que tiene la clase en la muestra es tan bajo, que impide caracterizar convenientemente a la clase en el conjunto de los datos. Cuando el desbalanceo es considerable, descubrir regularidades inherentes a la clase minoritaria se convierte en una

tarea ardua y de poca fiabilidad. Japkowicz y Stephen (2002) concluyen que si los dominios son separables linealmente los modelos no son sensibles al problema del desequilibrio de las clases.

Las soluciones para tratar el desbalanceo se pueden encuadrar en dos grupos: soluciones a nivel de datos y a nivel de algoritmos.

Las técnicas dirigidas a modificar los datos tratan de remuestrear las tallas de entrenamiento, bien sea a través del sobremuestreo de la clase minoritaria o del submuestreo de la clase que tiene mayores instancias. Aunque estas técnicas han demostrado su efectividad, no dejan de tener ciertos inconvenientes: pueden eliminar ejemplos útiles e incrementar los costes. Otra crítica a esta estrategia se refiere al cambio que se realiza en la distribución original del conjunto de entrenamiento de los datos.

La técnica más sencilla de sobremuestreo es la aleatoria simple a través de la réplica de ejemplos en la misma clase, pero este método puede ocasionar un alto sobreajuste de los clasificadores.

Como técnica más inteligente para incrementar los ejemplos de la clase minoritaria se encuentra el algoritmo SMOTE (Synthetic Minority Over-sampling TTechnique) originario de Chawla y otros (2002). En este método la creación de nuevas muestras se origina a través de la interpolación. En un primer paso elegimos los K vecinos más cercanos y que pertenecen a su misma clase. Posteriormente elegimos el número de muestras artificiales que se generarán y, finalmente, para generar una nueva muestra, se calcula la diferencia entre el vector de atributos bajo consideración y uno de los vecinos más cercanos de los k vecinos elegidos al azar. El resultado de la diferencia se multiplica por un valor aleatorio entre cero y uno.

El algoritmo SMOTE se ha modificado de diferentes maneras para adaptarse mejor a muchos ejemplos. Algunas de estas aportaciones son las efectuadas por Han *et al.* (2005) que proponen el algoritmo Borderline-SMOTE para generar ejemplos positivos cercanos a una frontera. Wang *et al.* (2006) presentan el algoritmo LLE-SMOTE (Locally Linear Embedding) que proyecta conjuntos de alta dimensionalidad a otro de menor dimensionalidad. En este espacio de reducida dimensionalidad es donde se aplica SMOTE y después los ejemplos generados son transformados a su espacio de representación original.

Otras formas de obtener una representación mayor de la clase minoritaria se basan en técnicas de agrupamiento, por ejemplo Japkowicz (2001) emplea el algoritmo de clustering k-medias sobre cada clase por separado. Los clusters resultantes se sobremuestrean aleatoriamente hasta conseguir un equilibrio entre las clases. Otro trabajo en esta línea de investigación es el de Cohen *et al.* (2006) que también explora la generación de nuevas instancias a través de algoritmos de clustering, pero en este caso los centroides de los clústeres se obtienen a través de un algoritmo aglomerativo jerárquico.

En cuanto a las técnicas de submuestreo una de las primeras propuestas para editar o filtrar las muestras de entrenamiento fue el algoritmo de Edición de Wilson (1972), también conocido como la regla del vecino más cercano editado (Edited Nearest Neighbor). Actualmente existen muchas formas de proceder, una buena parte de ellas se realizan a través del submuestreo aleatorio y otras de diferentes formas: con submuestreo dirigido, algoritmo One-sides selection de Kubat y Matwin, (1997), con técnicas de vecindad, algoritmo Neighborhood Cleaning Rule de Laurikkala, (2002). con submuestreo aplicando algoritmos genéticos, Kuncheva y Jain, 1999, con submuestreo por distancia, Zhanng y Mani,

(2003), con submuestreo por clustering, Cohen *et al.* (2006), a través del aprendizaje activo de Provost, (2003). Respecto a los métodos de clasificación en entornos no balanceados que no cambian la distribución a priori de las clases, nos encontramos con las soluciones a nivel de algoritmos: aprendizaje sensible al coste, algoritmos de clasificación con sesgo hacia la clase minoritaria y los clasificadores de una clase.

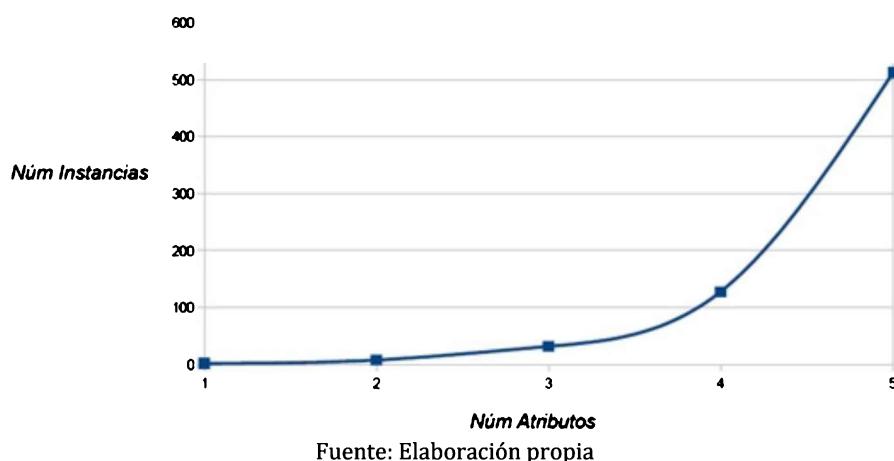
En muchas investigaciones es práctica normal que los resultados del conjunto de datos se balanceen a través de un método mixto donde se aplica el método SMOTE a la clase minoritaria y se reduce la muestra de la clase mayoritaria a través de diferentes métodos de muestreo. Un método especialmente útil es el denominado método del submuestreo equilibrado del cubo, propuesto por Deville y Tillé (2004). Este método de muestreo es el único que nos permite seleccionar una muestra equilibrada sobre variables auxiliares con probabilidades de inclusión iguales o no. El método del cubo selecciona únicamente las muestras cuyos estimadores de Horvitz-Thompson son iguales a los totales de las variables auxiliares conocidas.

Otra forma que disponemos para combatir el desbalanceo de clases, es a través del establecimiento de una matriz de costes, lo que se ha llamado método del costo-sensitivo (cost-sensitive). Este método se basa en la aseveración de que el precio de cometer un error de clasificación debe ser distinto para cada clase.

6. REDUCCIÓN DE VARIABLES O DE LA DIMENSIONALIDAD

La selección de las variables que van a formar parte del fichero inicial es una fase vital y transcendente de la minería de datos. El alto número de variables recogidas para el estudio de un fenómeno a veces es un problema para el aprendizaje si el número de instancias o ejemplos de la muestra es reducido. Este es el problema conocido como la maldición de la multidimensionalidad.

FIGURA 26. LA MALDICIÓN DE LA DIMENSIONALIDAD



La selección de atributos es uno de los problemas más complejos al que pretende hacer frente el aprendizaje automático. El objetivo es eliminar variables redundantes, atributos espurios y, en general, todas aquellas variables donde su presencia en la base de datos no aporte un aumento de la información de la misma. En bases de datos con muchas características y pocas instancias resulta imposible construir modelos.

En problemas de clasificación donde el número de instancias o ejemplos de la muestra en una de las clases es reducido, es de vital importancia reducir las variables del modelo lo que, por otra parte, hará más fácil entender las relaciones existentes entre las variables.

Existen diferentes estrategias a la hora de realizar una selección de variables y, evidentemente, una amplia literatura sobre este tema tan importante y apasionante. Aquí solo se explican algunos de los métodos más utilizados.

En la literatura de selección de variables existen dos métodos generales para escoger las mejores características de la base de datos: métodos de filtro y métodos basados en modelos.

6.1. APROXIMACIÓN INDIRECTA O FILTER

La aproximación indirecta o filter (Ben-Bassat, 1982) hace uso de métodos heurísticos¹⁶ para determinar el subconjunto de atributos óptimo. Los atributos irrelevantes son filtrados antes de aplicar las técnicas de minería de datos. Los atributos se seleccionan y evalúan de forma independiente del algoritmo de aprendizaje y, por tanto, el criterio que establece las variables óptimas se basa en una medida de calidad que se calcula a partir de los mismos datos. Una de sus principales ventajas es la rapidez de cálculo, lo que hace a esta aproximación especialmente interesante de cara al análisis de grandes conjuntos de datos.

Se ha de definir un algoritmo que evaluará individualmente cada atributo del conjunto de datos inicial, y que se denomina “attribute evaluator”, y un método de búsqueda que realizará una búsqueda en el espacio de posibles combinaciones de todos los subconjuntos del conjunto de atributos. De esta forma, podremos evaluar independientemente cada una de las combinaciones de atributos y, con ello, seleccionar aquellas configuraciones de atributos que maximicen la función de evaluación de atributos.

Para resolver el problema de plantear combinaciones de atributos, la función que evalúa cada subconjunto de atributos debe utilizar un algoritmo de búsqueda que recorra el espacio de posibles combinaciones de una forma organizada, o adecuada al problema. Habitualmente, en las situaciones en las que se utiliza la selección de atributos no es posible hacer un recorrido exhaustivo en el espacio de combinaciones, por lo que la selección adecuada de un algoritmo de búsqueda resulta crítica.

Además del método de las componentes principales¹⁷, existen dos tipos de evaluadores: evaluadores de subconjuntos o selectores (SubSetVal) y prorrataeadores de atributos (AttributeEval).

Los SubSetVal necesitan una estrategia de búsqueda (Search Method) y los AttributeEval ordenan las variables según su relevancia, así que necesitan un Ranker.

Para seleccionar las variables de mayor relevancia se pueden utilizar varios métodos de selección de atributos disponibles, entre otros, en la librería caret en R o en el programa WEKA (Waikato Environment for Knowledge Analysis).

Dos de estos algoritmos evaluadores de atributos son el “CfsSubsetEval” y el ConsistencySubsetEval. El primer algoritmo es el más sencillo, ya que puntúa a cada atributo

¹⁶ Heurístico: regla matemática capaz de guiar un proceso de búsqueda hacia una solución.

¹⁷ Estudiado en el tema 4.

en función de su entropía. Como algoritmos de búsqueda se utilizan cuatro métodos: Best First, algoritmos genéticos, Greedy y Tabu Search.

CfsSubsetEval evalúa un subconjunto de atributos considerando la habilidad predictiva individual de cada variable, así como el grado de redundancia entre ellos. Se prefieren los subconjuntos de atributos que estén altamente correlacionados con la variable predictora y tengan baja intercorrelación entre ellos.

ConsistencySubsetEval: Evalúa un subconjunto de atributos por el nivel de consistencia en los valores de la clase al proyectar las instancias de entrenamiento sobre el subconjunto de atributos.

El método Ranker nos facilita una lista ordenada de los atributos atendiendo a su calidad:

1. ChiSquaredAttributeEval: calcula el valor estadístico Chi-cuadrado de cada atributo con respecto a la clase y así obtiene el nivel de correlación entre la clase y cada atributo.
2. GainRatioAttributeEval: evalúa cada atributo midiendo su razón de beneficio con respecto a la clase.
3. InfoGainAttributeEval: evalúa los atributos midiendo la ganancia de información de cada uno con respecto a la clase. Previamente discretiza los atributos numéricos.
4. OneRAttributeEval: evalúa la calidad de cada atributo utilizando el clasificador OneR, el cual usa el atributo de mínimo error para predecir, discretizando los atributos numéricos.

6.2. APROXIMACIÓN DIRECTA O WRAPPER (ENVOLTURA)

Los métodos de selección de variables utilizando Wrappers (envoltorios) usan el desempeño de algún clasificador (algoritmo de aprendizaje) para determinar el conjunto de atributos óptimos (WrapperSubsetEval). Cada posible subconjunto de variables candidato es evaluado por medio del modelo clasificatorio inducido -con el paradigma utilizado-.

Este procedimiento emplea validación cruzada para estimar la exactitud del esquema de aprendizaje en cada conjunto.

Señalar que han sido utilizados diferentes heurísticos estocásticos de búsqueda (enfriamiento estadístico, algoritmos genéticos¹⁸ y algoritmos de estimación de distribuciones), así como algoritmos clásicos (hill-climbing, best-first, branch& bound, etc.) con éxito (Kohavi, 1995).

Para completar este epígrafe sobre la selección de variables, vamos a mostrar de forma muy resumida las aportaciones que realizan la librería caret de R y el programa WEKA, que ofrece una notable colección de métodos para la selección de variables.

¹⁸ Esta técnica, propuesta por Holland (1975), supone uno de los enfoques más originales en la minería de datos. Se inspiran en el comportamiento natural de la evolución, y para ello se codifica cada uno de los casos de prueba como una cadena binaria (que se asemejaría a un gen). Esta cadena se replica o se inhibe en función de su importancia, determinada por una función denominada de ajuste o fitness. Los algoritmos genéticos son adecuados para obtener buenas aproximaciones en problemas de búsqueda, aprendizaje y optimización (Marczyk, 2004).

6.3. SELECCIÓN DE VARIABLES CON LA LIBRERÍA CARET DE R

Una buena parte de los modelos contenidos en caret nos ofrecen una evaluación de las variables, tanto en problemas de regresión como de clasificación, a través de una cuantificación que nos permite saber qué variables de la base de datos son importantes para el modelo aplicado. Presenta las variables predictoras ordenadas de mayor a menor importancia, a través de la función Rank.

Caret dispone de métodos filter y wrapper. Utiliza la selección de características a través de un método recursivo de eliminación de variables implementado con dos algoritmos, uno a través de la función rfelter y el otro, basado en el remuestreo, está en la función rfe.

Otras formas de selección de variables que contempla caret en los métodos de búsqueda es a través de algoritmos genéticos y de simulado recocido (simulated annealing).

El código de estas dos formas de proceder se ve en el siguiente ejemplo:

```
library(caret)
ctrl <- gafsControl(functions = caretGA)
obj <- gafs(x = predictors,
            y = outcome,
            iters = 100,
            gafsControl = ctrl,
            Now pass options to `train`  

            method = "lm")  

library(mlbench)
n <- 100
p <- 40
sigma <- 1
set.seed(1)
sim <- mlbench.friedman1(n, sd = sigma)
colnames(sim$x) <- c(paste("real", 1:5, sep = ""),
                      paste("bogus", 1:5, sep = ""))
bogus <- matrix(rnorm(n * p), nrow = n)
colnames(bogus) <- paste("bogus", 5+(1:ncol(bogus)), sep = "")
x <- cbind(sim$x, bogus)
y <- sim$y  

normalization <- preProcess(x)
x <- predict(normalization, x)
x <- as.data.frame(x)  

ga_ctrl <- gafsControl(functions = rfGA,
                        method = "repeatedcv",
                        repeats = 5)
set.seed(10)
library(doParallel)
detectCores()
registerDoParallel(cores=12)  

system.time(rf_ga <- gafs(x = x, y = y,
                            iters = 200,
                            gafsControl = ga_ctrl))  

# user    system   elapsed
# 3001.36    21.54  20981.66  

library(ggplot2)
plot(rf_ga) +theme_bw()  

user    system   elapsed
```

3001.36 21.54 20981.66

rf_ga

Genetic Algorithm Feature Selection

**100 samples
50 predictors**

**Maximum generations: 200
Population per generation: 50
Crossover probability: 0.8
Mutation probability: 0.1
Elitism: 0**

**Internal performance values: RMSE, Rsquared
Subset selection driven to minimize internal RMSE**

**External performance values: RMSE, Rsquared, MAE
Best iteration chose by minimizing external RMSE
External resampling method: Cross-Validated (10 fold, repeated 5 times)**

During resampling:

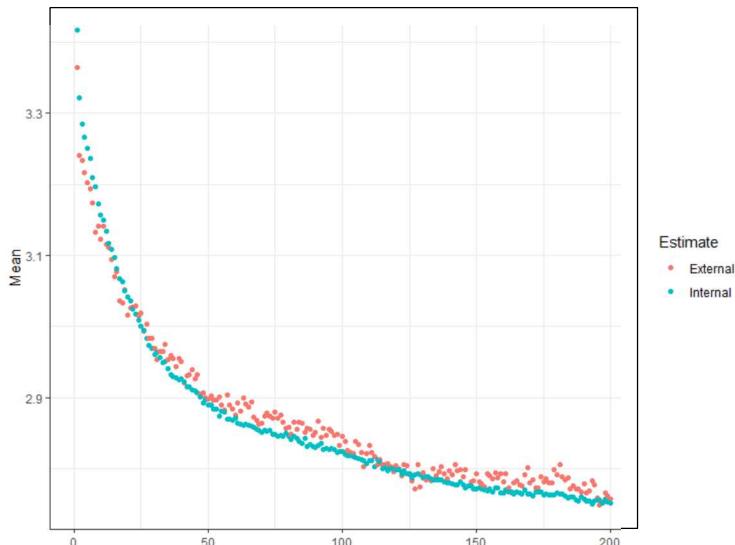
- * the top 5 selected variables (out of a possible 50):
real1 (100%), real2 (100%), real4 (100%), real5 (100%), real3 (90%)
- * on average, 8.4 variables were selected (min = 6, max = 12)

In the final search using the entire training set:

- * 12 features selected at iteration 196 including:
real1, real2, real3, real4, real5 ...
- * external performance at this iteration is

RMSE	Rsquared	MAE
2.7483	0.7677	2.3141

FIGURA 27. AJUSTE INTERNO Y EXTERNO. RANDOM FOREST CON ALGORITMOS GENÉTICOS



Fuente: Elaboración propia

Selección con simulated annealing

```
# Simulated Annealing
system.time(
sa_ctrl <- safsControl(functions = rfSA, method = "repeatedcv", repeats = 5,i
```

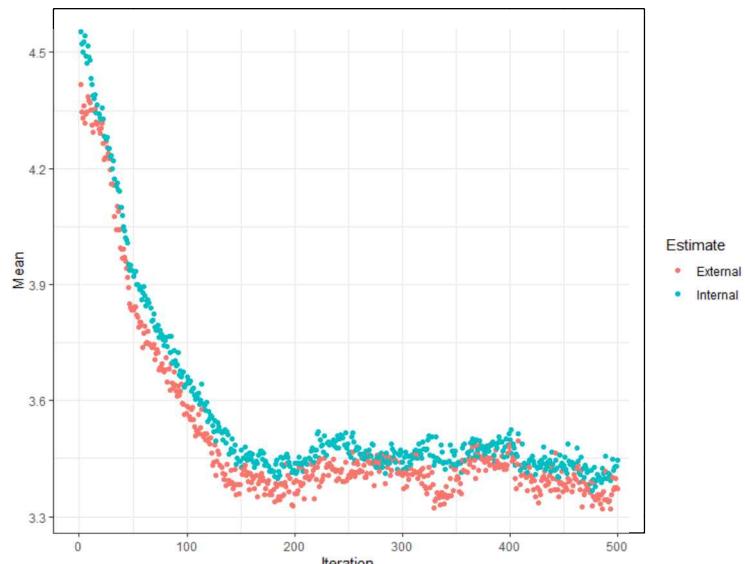
```

mprove = 50))

set.seed(10)
library(doParallel)
detectCores()
registerDoParallel(cores=12)
system.time(rf_sa <- safs(x = x, y = y,
                           iters = 500,
                           safsControl = sa_ctrl))
rf_sa
library(ggplot2)
plot(rf_sa) +theme_bw()

```

FIGURA 28. AJUSTE INTERNO Y EXTERNO. RANDOM FOREST CON ALGORITMOS GENÉTICOS



Fuente: Elaboración propia

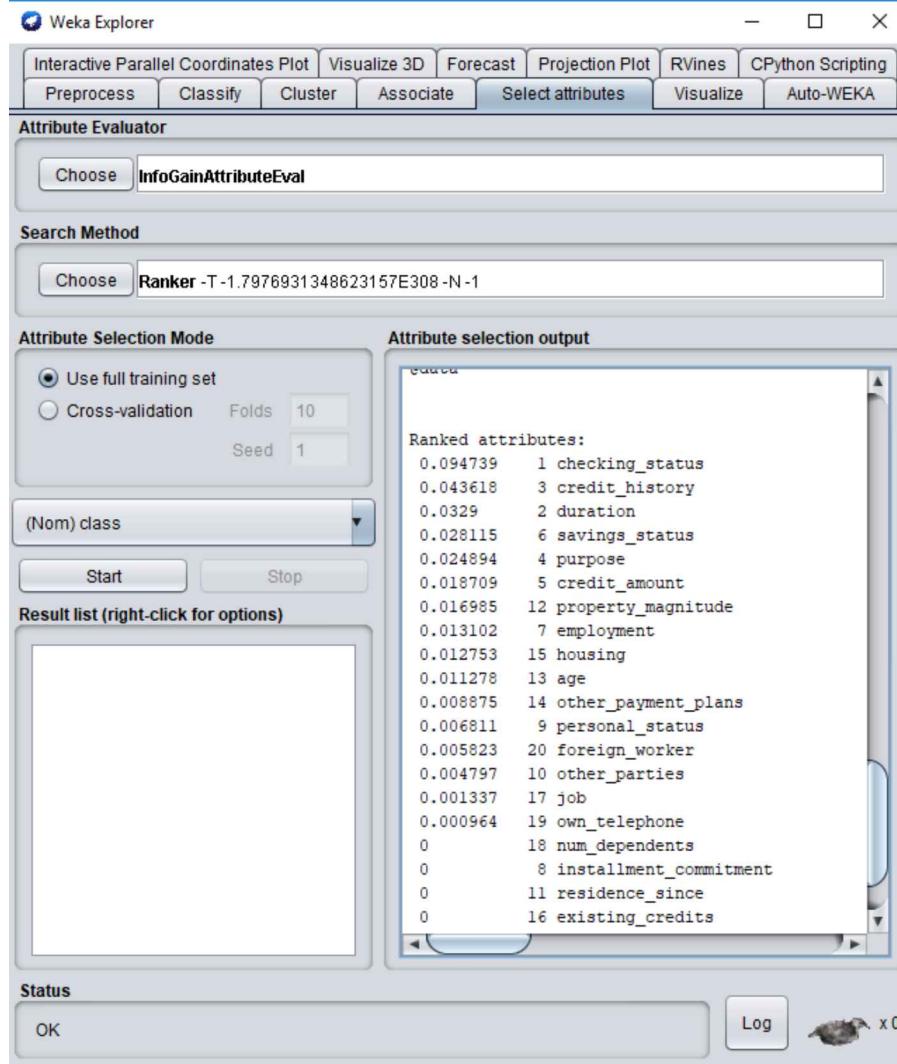
6.4. SELECCIÓN DE VARIABLES CON EL PROGRAMA WEKA

En las imágenes que siguen a continuación se muestran las potencialidades del programa WEKA a la hora de efectuar un procedimiento de selección de variables.

La selección de variables en WEKA está en dos lugares distintos que ofrecen diferente información: en el Menú Select attributes y en Classify.

En la Figura nº 29 se observa el resultado de aplicar el método ranker de WEKA a la base de datos german credit. En todos los métodos que dispone WEKA para la selección de atributos siempre hay que elegir el método de búsqueda (Search Method) y la función de evaluación (Attribute Evaluator).

FIGURA 29. CONFIGURACIÓN DEL MÉTODO RANKER EN WEKA



Fuente: Weka

FIGURA 30. SELECCIÓN DE ATRIBUTOS ORDENADOS

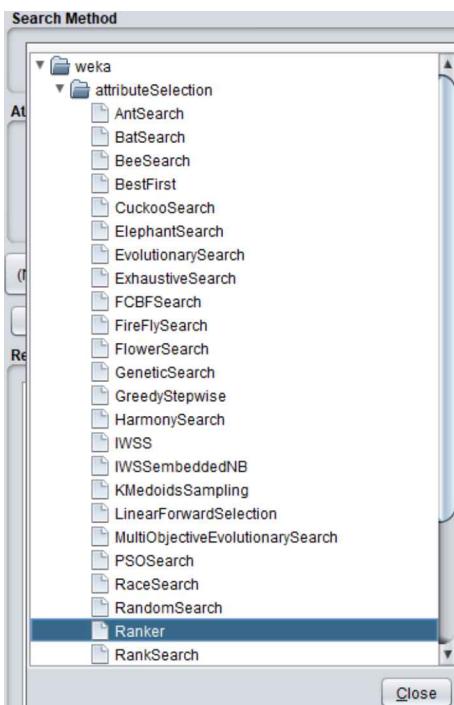
```

Ranked attributes:
0.094739 1 checking_status
0.043618 3 credit_history
0.0329 2 duration
0.028115 6 savings_status
0.024894 4 purpose
0.018709 5 credit_amount
0.016985 12 property_magnitude
0.013102 7 employment
0.012753 15 housing
0.011278 13 age
0.008875 14 other_payment_plans
0.006811 9 personal_status
0.005823 20 foreign_worker
0.004797 10 other_parties
0.001337 17 job
0.000964 19 own_telephone
0 18 num_dependents
0 8 installment_commitment
0 11 residence_since
0 16 existing_credits

Selected attributes: 1,3,2,6,4,5,12,7,15,13,14,9,20,10,17,19,18,8,11,16 : 20
  
```

Fuente: Weka

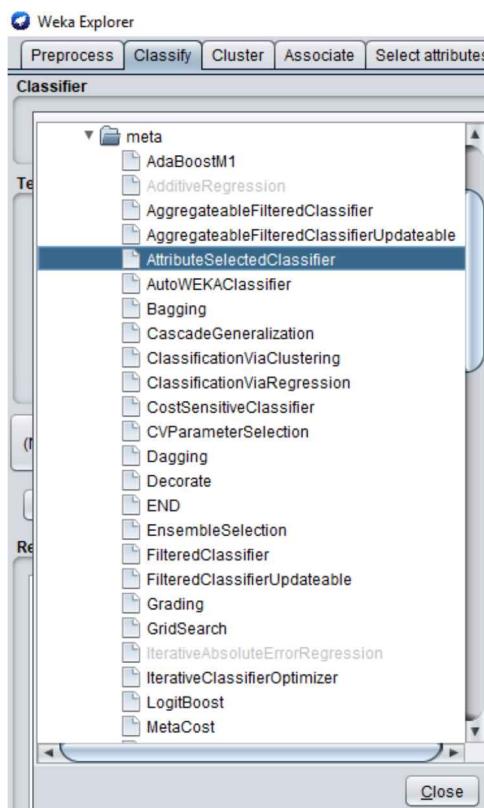
FIGURA 31. MÉTODOS DE BÚSQUEDA DEL PROGRAMA WEKA



Fuente: Weka

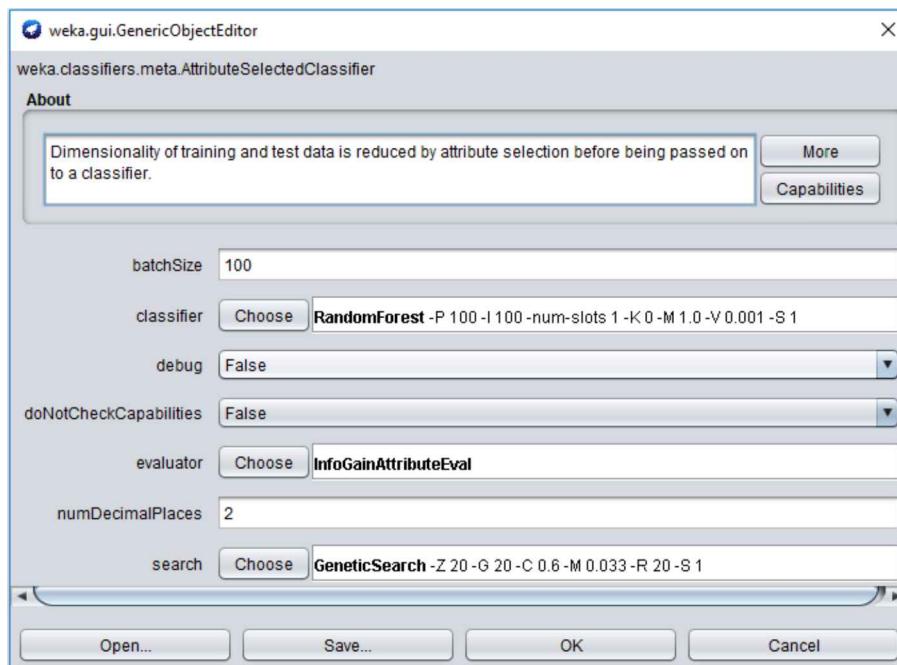
En el Menú Classify/Meta/AttributeSelectedClassifier encontramos la forma de seleccionar más completa que tiene WEKA del tipo wrapper, es decir, selección de variables con modelos.

FIGURA 32. ALGORITMOS METACLASIFICADORES DEWEKA



Fuente: Weka

FIGURA 33. PANTALLA DE INTRODUCCIÓN DE DATOS PARA LA SELECCIÓN DE VARIABLES



Fuente: Weka

FIGURA 34. RESULTADOS DE LA CLASIFICACIÓN CON EL MÉTODO RANDOM FOREST CON SELECCIÓN DE VARIABLES

```

Header of reduced data:
@relation 'german_credit-weka.filters.unsupervised.attribute.Remove-V-R1-3,21'

@attribute checking_status {<0,0<=X<200,>=200,'no checking'}
@attribute duration numeric
@attribute credit_history {'no credits/all paid','all paid','existing paid','delayed previously','critical/other existing credit'}
@attribute class {good,bad}

@data

Classifier Model
RandomForest

Bagging with 100 iterations and base learner

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.18 seconds

==== Stratified cross-validation ====
==== Summary ====

Correctly Classified Instances      718      71.8    %
Incorrectly Classified Instances   282      28.2    %
Kappa statistic                   0.3047
Mean absolute error               0.3278
Root mean squared error          0.4407
Relative absolute error           78.0049 %
Root relative squared error      96.1659 %
Total Number of Instances        1000

==== Detailed Accuracy By Class ====

      TP Rate   FP Rate   Precision   Recall   F-Measure   MCC     ROC Area   PRC Area   Class
      0,824     0,530     0,784     0,824     0,804     0,306     0,723     0,839     good
      0,470     0,176     0,534     0,470     0,500     0,306     0,723     0,527     bad
Weighted Avg.   0,718     0,424     0,709     0,718     0,713     0,306     0,723     0,745

==== Confusion Matrix ====
      a   b   <- classified as
  577 123 |   a = good
  159 141 |   b = bad

```

Fuente: Weka

En la pantalla de introducción de datos de AttributeSelectedClassifier hay que configurar el método de clasificación y sus parámetros, en este caso el Random Forest, el evaluador que se ha elegido ha sido la ganancia de información (InfoGainAttributeEval) y el método de búsqueda (búsqueda a través de algoritmos genéticos).

Los resultados de la clasificación de los clientes del banco alemán German Credit con selección de variables se muestran en la Figura 34.

BIBLIOGRAFÍA

- Balahur, A. (2011). *Methods and resources for sentiment analysis in multilingual documents of different text types*. Tesis doctoral. Universidad de Alicante.
- Bankinter. (2011). *El Internet de las Cosas. En un mundo conectado de objetos inteligentes*. Fundación de la Innovación y Acceture.
- Beltrán, P.M. (2015). *Diseño e implementación de un nuevo clasificador de préstamos bancarios a través de la minería de datos*. Tesis doctoral. UNED.
- Berzal *et al.* (2001). "TBAR: An efficient method for association rule mining in relational databases". *Data & Knowledge Engineering*, 37(1):47-64.
- Berzal, F., Cubero, J.C., Cuenca, F. y Martín-Bautista, M.J. (2003). "On the quest for easy-to-understand splitting rules". *Data & Knowledge Engineering*, 44(1):31-48.
- Biganzoli, E., Boracchi, P., Mariani, L. y Marubini, E. (1998). "Feed-forward neural networks for the analysis of censored survival data: a partial logistic regression approach". *Statistics in Medicine*, 17(10):1169-1186.
- Bilge, U., Refenes, A.N., Diamond, C. y Shadbolt, J. (1993). "Application of sensitivity analysis techniques to neural network bond forecasting". En A.N. Refenes (Ed.), *Proceedings of 1st International Workshop on Neural Networks in the Capital Markets* (12). London: London Business School.
- Bishop, C.M. (1994). "Neural networks and their applications". *Review of Scientific Instruments*, 65(6):1803-1832.
- Bishop, C.M. (1995). *Neural networks for pattern recognition*. Oxford: Oxford University Press.
- Boser, B.E.; Guyon, I.M. y Vapnik, V.N. (1992). "A training algorithm for optimal margin classifiers". En *Proceedings of the fifth annual workshop on Computational learning theory* (144-152).
- Bourouche, J.M. y Tennenhaus, M. (1972). "Some segmentation methods". *Metra*, 7:407-418.
- Breiman, L. (2001). "Random forests". *Machine Learning*, 45:5-32.
- Breiman, L., Friedman, J.H., Olshen, R.A. y Stone, C.J. (1984). *Classification and regression trees*. Monterey, CA: Wadsworth & Brooks/Cole Advanced Book & Software.
- Britos, P.; Dieste, O. y García, R. (2008). "Requirements Elicitation in Data Mining for Business Intelligence Projects.". En David, G.M. *et al.* (Eds.), *IFIP International Federation for Information Processing. Advances in Information Systems Research, Education and Practice*, 274:139-150.
- Castellanos, J., Pazos, A., Ríos, J. y Zafra, J. L. (1994). "Sensitivity analysis on neural networks for meteorological variable forecasting". En J. Vlontzos, J.N. Hwang y E. Wilson (Eds.), *Proceedings of IEEE Workshop on Neural Networks for Signal Processing* (587-595). New York: IEEE.

- Catlett, J. (1991). "On changing continuous attributtes into ordered discrete attributes". In *Proceedings of European Working Session on Learning*, pages 164–178. Springer-Verlag.
- Cellard, J.C., Labbe, B. y Savitsky, G. (1967). "Le programme ELISEE, presentation et application". *Metra*, 3(6):511-519.
- Chapman, P.; Clinton, J.; Kerber, R.; Khabaza, T.; Reinartz, T.; Shearer, C. y Wirth, R. (2000). *CRISP-DM 1.0 Step-by-step Data Mining guide*. CRISP-DM, SPSS, <https://www.the-modeling-agency.com/crisp-dm.pdf>
- Chawla, N.V., Bowyer, K.W., Hall, L.O. y Kegelmeyer, W.P. (2002). "SMOTE: Synthetic Minority Over-Sampling Technique". *Journal of Artificial Intelligence research*: 321-357.
- Cheng, B. y Titterington, D.M. (1994). "Neural networks: a review from a statistical perspective". *Statistical Science*, 9(1):2-54.
- Chi, D.K.Y., Cheung, B. y Wong, A.K.C. (1990). "Information synthesis based on hierarchical entropy discretization". *Experimental and Theoretical Artificial Intelligence*, 2:117-129.
- Cohen, G., Hilario, M., Sax, H., Hugonnet, S. y Geissbuhler, A. (2006). "Learning from imbalancing Data in Surveillance of Nosocomial Infection". *Artificial Intelligence in Medicine*: 7-18.
- Cortes, C. y Vapnik, V. (1995). "Support-vector networks". *Machine Learning*, 20:273-297.
- Cottrell, G.W., Munro, P. y Zipser, D. (1989). "Image compression by back propagation: an example of extensional programming". En N.E. Sharkey (Ed.), *Models of cognition: a review of cognitive science* (208-240). Norwood, NJ: Ablex Publishing Corp.
- Cox, M. y Ellsworth, D. (1997). "Application controlled demand paging for out of core visualization". *Report NAS-97-010, July*. Moffet Field: NASA Ames Research Centre.
- Cybenko, G. (1989). "Approximation by superpositions of a sigmoidal function". *Mathematical Control, Signal and Systems*, 2: 303-314.
- De Laurentiis, M. y Ravdin, P.M. (1994). "A technique for using neural network analysis to perform survival analysis of censored data". *Cancer Letters*, 77:127-138.
- Deville, J.C. y Tillé, Y. (2004). "Eficient balanced sampling: The cube method". *Biometrika*, 91:893-912.
- Edwards, W. (1998). Hailfinder. Tools for and experiences with bayesian normative modeling". *American Psychologist*, 53:416-428.
- Elman, J.L. (1990). "Finding structure in time". *Cognitive Science*, 14:179-211.
- Engelbrecht, A.P., Cloete, I. y Zurada, J.M. (1995). "Determining the significance of input parameters using sensitivity analysis". En J. Mira y F. Sandoval (Eds.), *Proceedings of International Workshop on Artificial Neural Networks* (382-388). New York: Springer.
- Engelbrecht, A.P., Fletcher, L. y Cloete, I. (1999). *Variance analysis of sensitivity information for pruning multilayer feedforward*. Neural networks.

- Catlett, J. (1991). "On changing continuous attributtes into ordered discrete attributes". In *Proceedings of European Working Session on Learning*, pages 164–178. Springer-Verlag.
- Cellard, J.C., Labbe, B. y Savitsky, G. (1967). "Le programme ELISEE, presentation et application". *Metra*, 3(6):511-519.
- Chapman, P.; Clinton, J.; Kerber, R.; Khabaza, T.; Reinartz, T.; Shearer, C. y Wirth, R. (2000). *CRISP-DM 1.0 Step-by-step Data Mining guide*. CRISP-DM, SPSS, <http://www.whitepapercentral.com/browse/marketing/crisp-dm-1-0-step-by-step-data-mining-guide/>
- Chawla, N.V., Bowyer, K.W., Hall, L.O. y Kegelmeyer, W.P. (2002). "SMOTE: Synthetic Minority Over-Sampling Technique". *Journal of Artificial Intelligence research*: 321-357.
- Cheng, B. y Titterington, D.M. (1994). "Neural networks: a review from a statistical perspective". *Statistical Science*, 9(1):2-54.
- Chiu, D.K.Y., Cheung, B. y Wong, A.K.C. (1990). "Information synthesis based on hierarchical entropy discretization". *Experimental and Theoretical Artificial Intelligence*, 2:117-129.
- Cohen, G., Hilario, M., Sax, H., Hugonnet, S. y Geissbuhler, A. (2006). "Learning from imbalancing Data in Surveillance of Nosocomial Infection". *Artificial Intelligence in Medicine*: 7-18.
- Cortes, C. y Vapnik, V. (1995). "Support-vector networks". *Machine Learning*, 20:273-297.
- Cottrell, G.W., Munro, P. y Zipser, D. (1989). "Image compression by back propagation: an example of extensional programming". En N.E. Sharkey (Ed.), *Models of cognition: a review of cognitive science* (208-240). Norwood, NJ: Ablex Publishing Corp.
- Cox, M. y Ellsworth, D. (1997). "Application controlled demand paging for out of core visualization". *Report NAS-97-010, July*. Moffet Field: NASA Ames Research Centre.
- Cybenko, G. (1989). "Approximation by superpositions of a sigmoidal function". *Mathematical Control, Signal and Systems*, 2: 303-314.
- De Laurentiis, M. y Ravdin, P.M. (1994). "A technique for using neural network analysis to perform survival analysis of censored data". *Cancer Letters*, 77:127-138.
- Deville, J.C. y Tillé, Y. (2004). "Eficient balanced sampling: The cube method". *Biometrika*, 91:893-912.
- Edwards, W. (1998). Hailfinder. Tools for and experiences with bayesian normative modeling". *American Psychologist*, 53:416-428.
- Elman, J.L. (1990). "Finding structure in time". *Cognitive Science*, 14:179-211.
- Engelbrecht, A.P., Cloete, I. y Zurada, J.M. (1995). "Determining the significance of input parameters using sensitivity analysis". En J. Mira y F. Sandoval (Eds.), *Proceedings of International Workshop on Artificial Neural Networks* (382-388). New York: Springer.
- Engelbrecht, A.P., Fletcher, L. y Cloete, I. (1999). *Variance analysis of sensitivity information for pruning multilayer feedforward*. Neural networks.

- Escobar, M. (2007). "El análisis de segmentación: técnicas y aplicaciones de los árboles de clasificación". *Centro de Investigaciones Sociológicas. Cuadernos metodológicos nº 39*.
- Fayyad, U.; Piatetsky-Shapiro, G. y Smyth, P. (1996). "From Data Mining to Knowledge Discovery in Databases". *AI Magazine*, 17(3):37-54.
- Fayyard, U.M. y Irani, K.B. (1993). "Multi-interval discretization of continuous valued attributes for classification learning". In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence* (1022-1027). San Francisco, CA: Morgan Kaufmann.
- Fletcher, R. (1987). *Practical Methods of Optimization*. John Wiley & Sons, Nueva York. 2^a edición.
- Flexer, A. (1995). *Connectionist and statisticians, friends or foes*. The Austrian Research Institute for Artificial Intelligence. Recuperado 20/01/01, desde https://link.springer.com/chapter/10.1007%2F3-540-59497-3_209
- Frost, F. y Karri, V. (1999). "Determining the influence of input parameters on BP neural network output error using sensitivity analysis". En B. Verma, H. Selvaraj, A. Carvalho y X. Yao (Eds.), *Proceedings of the Third International Conference on Computational Intelligence and Multimedia Applications* (45-49). Los Alamitos, CA: IEEE Computer Society Press.
- Fu, L. y Chen, T. (1993). "Sensitivity analysis for input vector in multilayer feedforward neural networks". En IEEE (Ed.), *Proceedings of IEEE International Conference on Neural Networks* (215-218). New York: IEEE.
- Funahashi, K. (1989). "On the approximate realization of continuous mapping by neural networks". *Neural Networks*, 2:183-192.
- Garson, G.D. (1991a). "Interpreting neural-network connection weights". *AI Expert*, April, 47-51.
- Garson, G.D. (1991b). "A comparison of neural network and expert systems algorithms with common multivariate procedures for analysis of social science data". *Social Science Computer Review*, 9(3):399-434.
- Gedeon, T.D. (1997). "Data mining of inputs: analysing magnitude and functional measures". *International Journal of Neural Systems*, 8(2):209-218.
- Gehrke, J., Loh, W.Y. y Ramakrishnan, R. (1999b). "Classification and regression: money can grow on trees". *Tutorial notes for ACM SIGKDD 1999 international conference on Knowledge Discovery and Data Mining*, August 15-18, 1999, San Diego, California, USA, pp. 1-73.
- Goodman, L.A. (1979). "Simple Model for Statistica Data analysis of Multivariate Observations". *Journal of the American Statistical Association*, 74: 537-552.
- Guo, Z. y Uhrig, R.E. (1992). "Sensitivity analysis and applications to nuclear power plant". En IEEE (Ed.), *International Joint Conference on Neural Networks* (453-458). Piscataway, NJ: IEEE.
- Haberman, S.J. (1978). *Analysis of Qualitative Data*. New York. Academic press.

- <http://www-05.ibm.com/services/es/bcs/html/bao/bao-revolucion-en-las-capacidades-analiticas-y-optimizacion-de-negocio.pdf>
- IBM (2012). *Manual CRISP-DM de IBM SPSS. Modeler*. IBM, <ftp://public.dhe.ibm.com/software/analytics/spss/documentation/modeler/15.0/es/CRISP-DM.pdf>
- IBM (2017). *Big Data Analytics. Employ the most effective big data technology*. IBM. <http://www.ibm.com/analytics/us/en/big-data/>
- Japkowicz, N. (2001). "Concept-Learning in the Presence of Between-Class and Within-Class Imbalances". En: E. Stroulia y S. Matwin (Eds.), *Canadian Conference on AI*, 2056 de LNCS: 67-77.
- Japkowicz, N. y Stephen, S. (2002). "The Class Imbalance Problem: A Systematic Study Intelligent Data". *Analysis Journal*, 6(5):1-32.
- Jiménez, R. (2002). *Aportaciones del proceso Data Mining en el análisis de datos*. (Memoria de investigación en el programa de doctorado del Departamento de Psicología, no publicada). Palma de Mallorca: Universidad de las Islas Baleares.
- Joyanes, L. (2014). *Big Data. Análisis de grandes volúmenes de datos en organizaciones*. Ed. Marcombo, S.A.
- Kadie, C.M.; Hovel, D. y Hovitz, E. (2001). "A component-centric toolkit for modeling and inference with bayesian networks". (*Tech. Rep. MSR-TR-2001-67*). Redmond, WA: Microsoft Corporation. 13(1):13-25.
- Kashani, J.H., Nair, S.S., Rao, V.G., Nair, J. y Reid, J.C. (1996). "Relationship of personality, environmental, and DICA variables to adolescent hopelessness: a neural network sensitivity approach". *Journal of American Children and Adolescent Psychiatry*, 35(5):640-645.
- Kass, G.V. (1980). "An Exploratory Technique for Investigating Large Quantities of Categorical Data". *Applied Statistic*, 29:119-117.
- Kemp, R.A., McAulay, C. y Palcic, B. (1997). "Opening the black box: the relationship between neural networks and linear discriminant functions". *Analytical Cellular Pathology*, 14:19-30.
- Kerber, R. (1992). "Lockheed Artificial Intelligence Center". *AAAI'92 Proceedings of the tenth*.
- Kolmogorov, A.N. (1957). "On the representation of continuous functions of several variables by means of superpositions of continuous functions of one variable". *Doklady Akademii Nauk SSSR*, 114:953-956.
- Kubat, M. y Matwin, S. (1997). "Addressing the Course of Imbalanced Training Sets: One-Sided Selection". En: D.H.Fisher (Ed.), *ICML*: 179-186.
- Kuncheva, L. y Jain. L.C. (1999). "Nearest neighbor classifier: Simultaneous editing and feature selection". *Pattern Recognition Letters*: 1149-1156.
- Laurikkala, J. (2002). "Instance-based data reduction for improved identification of difficult small classes". *Intelligent Data Analysis*: 311-322.

- Lee, S. y Kil, R. (1991). "A Gaussian potential function network with hierarchically self-organizing learning". *Neural Networks*. 4(2):207-224.
- Loh, W.Y. y Shih, Y.S. (1997). "Split Selection Method for Classification Trees". *Statistica Sinica*, 7:815-840.
- Loh, W.Y. y Vanichsetakul, N. (1988). "Tree-structured classification via generalized discriminant analysis (with discussion)". *J. Amer. Statist. Assoc.* 83:715-728.
- López de Mantaras, R. (1991). "A Distance-Based Attribute Selection Measure for Decision Tree Induction". *Machine Learning*, 6:81-92.
- López, J.; García, J. y De la Fuente, L. (2006). "Modelado causal con redes bayesianas". *Actas de las XXVII Jornadas de Automática*: 198-202.
- Luong, H.; Gauch, S. y Wang, Q. 2012. "Ontology Learning Using Word Net Lexical Expansion and Text Mining". En Shakurais, S. (Ed.), *Theory and Applications for Advanced Text Mining*. Chapter 5. <http://dx.doi.org/10.5772/51141>
- Madgison, J. (1989). *SPSS/PC + CHAID*. Chicago, SPSS Inc.
- Madgison, J. (1992). "Chi-Squared Analysis of a Scalable Dependen Variable". *Procedings of the 1992 Annual Meeting of the American Statistical Association*.
- Marczyk, A. (2004). "Genetic algorithms and evolutionary computation". *The Talk, Origins Archive*. 23 Apr. 2004. 7 Oct. 2006.
- Mardia, K.V., Kent, J.T. y Bibby, J.M. (1979). *Multivariate Analysis*. Academic Press, London.
- Martín, B., Sanz, A. (2001). *Redes neuronales y Sistemas Borrosos*. Ed. Rama.
- Martin, J.K. (1997). "An Exact Probability Metric for Decision Tree Splittingand Stopping". *Machine Learning*, 28:257-291.
- Masters, T. (1993). *Practical Neural Network Recipes in C++* (1ra. ed.). San Diego, CA: Academic Press Professional.
- Mayer-Schönberger, V. y Cukier, K. (2013). *Big Data. A Revolution That Will Transform How We Live, Work and Think*. Eds. John Murray.
- Méndez, P.D. y Rodríguez, A.D. (2009). *Herramienta de Estudio de Viabilidad para Proyectos que Utilizan la Metodología P³TQ*. Trabajo Profesional de Ingeniería en Informática. Laboratorio de Sistemas Inteligentes. Facultad de Ingeniería. Universidad de Buenos Aires.
- Milne, K. (1995). "Feature selection using neural networks with contribution measures". En IEEE (Ed.), *Proceedings of Australian Conference of Artificial Intelligence* (124-136). Sydney: IEEE West Australian Section.
- Modai, I., Saban, N. I., Stoler, M., Valevski, A. y Saban, N. (1995). Sensitivity profile of 41 psychiatric parameters determined by neural network in relation to 8-week outcome". *Computers in Human Behavior*, 11(2):181-190.
- Molina, J. y García, J. (2006). *Técnicas de análisis de datos. Aplicaciones prácticas utilizando Microsoft Excel y WEKA*. Universidad Carlos III de Madrid. Madrid. España.

- Montaño, J.J. (2005). *Herramientas informáticas de redes neuronales artificiales. Aplicaciones prácticas*. Universidad de Les Illes Balears.
- Montaño, J.J. y Palmer, A. (2003). "Numeric sensitivity analysis applied to feedforward neural networks". *Neural Computing & Applications*, 12:119-125.
- Montaño, J.J., Palmer, A. y Muñoz, P. (2011). "Artificial neural networks applied to forecasting time series". *Psicothema*, 23(2):322-329.
- Moody, J. y Darken, C.J. (1989). "Fast learning in networks of locally-tuned processing units". *Neural Computation*, 1:281-294.
- Morgan, J.N. y Sonquist, J.A. (1963). "Problems in the Analysis of Survey Data and a Proposal". *Journal of the American Statistical Association*, 58: 415-434.
- Morillas. A, Díaz. B. (2007). "El problema de los outliers multivariantes en el análisis de sectores clave y cluster industrial".
http://www.unizar.es/jornadasiozaragoza/archivos/pdf/Ponencia_Morillas_Antonio.pdf.
- Nelson, M.M., y Illingworth, W.T. (1991). *A practical guide to neural nets (Vol. 1)*. Reading, MA: Addison-Wesley.
- Nisbet, R., Elder IV, J. y Miner, G. (2009). *Handbook of statistical analysis and data mining applications*. Academic Press.
- Ohno-Machado, L. y Musen, M.A. (1997). "Sequential versus standard neural networks for pattern recognition: an example using the domain of coronary heart disease". *Computational Biology in Medicine*, 27(4):267-281.
- Oja, E. (1982). "A simplified neuron model as a principal component analyzer". *Journal of Mathematical Biology*, 15:267-273.
- Oja, E. (1989). "Neural networks, principal components, and subspaces". *International Journal of Neural Systems*, 1:61-68.
- Opara, J., Primozic, S. y Cvelbar, P. (1999). "Prediction of pharmacokinetic parameters and the assessment of their variability in bioequivalence studies by artificial neural networks". *Pharmaceutical Research*, 16(6):944-948.
- Palmer, A., Montaño, J.J. y Calafat, A. (2000). "Predicción del consumo de éxtasis a partir de redes neuronales artificiales". *Adicciones*, 12(1):29-41.
- Paniagua, E. (2015). *Big data. El poder de los datos*. Fundación Innovación Bankinter
- Pérez, C. y Santín, D. (2007). *Minería de datos. Técnicas y herramientas*. Ed. Paraninfo.
- Pérez, J.M. (2006). *Árboles Consolidados: Construcción de un árbol de clasificación basado en múltiples submuestras sin renunciar a la explicación*. Tesis doctoral. Universidad del País Vasco.
- Poggio, P. y Girosi, F. (1990). "Networks for Approximation and Learning", *Proc. IEEE*, 78(9).
- Pressman, R. (2005). *Ingeniería de software: un enfoque práctico*. Ed. McGraw-Hill.
- Provost, F. (2003). "Machine learning from imbalanced data sets 101 (Extended Abstract)". En: *AAAI: Workshop on Learning with Imbalanced Data Sets*.

- Montaño, J.J. (2005). *Herramientas informáticas de redes neuronales artificiales. Aplicaciones prácticas*. Universidad de Les Illes Balears.
- Montaño, J.J. y Palmer, A. (2003). "Numeric sensitivity analysis applied to feedforward neural networks". *Neural Computing & Applications*, 12:119-125.
- Montaño, J.J., Palmer, A. y Muñoz, P. (2011). "Artificial neural networks applied to forecasting time series". *Psicothema*, 23(2):322-329.
- Moody, J. y Darken, C.J. (1989). "Fast learning in networks of locally-tuned processing units". *Neural Computation*, 1:281-294.
- Morgan, J.N. y Sonquist, J.A. (1963). "Problems in the Analysis of Survey Data and a Proposal". *Journal of the American Statistical Association*, 58: 415-434.
- Morillas, A., Díaz, B. (2007). "El problema de los outliers multivariantes en el análisis de sectores clave y cluster industrial". *Jornadas Españolas de Análisis Input-Output (2)*: 134-154, Zaragoza
- Nelson, M.M., y Illingworth, W.T. (1991). *A practical guide to neural nets (Vol. 1)*. Reading, MA: Addison-Wesley.
- Nisbet, R., Elder IV, J. y Miner, G. (2009). *Handbook of statistical analysis and data mining applications*. Academic Press.
- Ohno-Machado, L. y Musen, M.A. (1997). "Sequential versus standard neural networks for pattern recognition: an example using the domain of coronary heart disease". *Computational Biology in Medicine*, 27(4):267-281.
- Oja, E. (1982). "A simplified neuron model as a principal component analyzer". *Journal of Mathematical Biology*, 15:267-273.
- Oja, E. (1989). "Neural networks, principal components, and subspaces". *International Journal of Neural Systems*, 1:61-68.
- Opara, J., Primozic, S. y Cvelbar, P. (1999). "Prediction of pharmacokinetic parameters and the assessment of their variability in bioequivalence studies by artificial neural networks". *Pharmaceutical Research*, 16(6):944-948.
- Palmer, A., Montaño, J.J. y Calafat, A. (2000). "Predicción del consumo de éxtasis a partir de redes neuronales artificiales". *Adicciones*, 12(1):29-41.
- Paniagua, E. (2015). *Big data. El poder de los datos*. Fundación Innovación Bankinter
- Pérez, C. y Santín, D. (2007). *Minería de datos. Técnicas y herramientas*. Ed. Paraninfo.
- Pérez, J.M. (2006). *Árboles Consolidados: Construcción de un árbol de clasificación basado en múltiples submuestras sin renunciar a la explicación*. Tesis doctoral. Universidad del País Vasco.
- Poggio, P. y Girosi, F. (1990). "Networks for Approximation and Learning", *Proc. IEEE*, 78(9).
- Pressman, R. (2005). *Ingeniería de software: un enfoque práctico*. Ed. McGraw-Hill.
- Provost, F. (2003). "Machine learning from imbalanced data sets 101 (Extended Abstract)". En: *AAAI: Workshop on Learning with Imbalanced Data Sets*.

- Shih, Y.S. (1999). "Families of splitting criteria for classification trees". *Statistics and Computing*, 9(4):309-315.
- Soares, S. (2013). *Big Data Governance. An Emerging Imperative*. Boise. MC Press Online.
- Spackman, K.A. (1992). "Maximum likelihood training of connectionist models: comparison with least-squares backpropagation and logistic regression". En IEEE (Ed.), *Proceedings of the 15th Annual Symposium of Computer Applications in Medical Care* (285-289). New York.
- Specht, D.F. (1991). "A generalized regression neural network". *IEEE Transactions on Neural Networks*, 2:568-576.
- Specht, D.F. (1990). "Probabilistic neural networks". *Neural Networks*, 3:110-118.
- Suzuki, J. (1996). "Learning Bayesian Belief Network Based on the Minimum Description Length Principle: An Efficient Algorithm Using the B&B Technique". In *Proceedings of the Thirteenth International Conference on Machine Learning* (462-470).
- Takenaga, H., Abe, S., Takatoo, M., Kayama, M., Kitamura, T. y Okuyama, Y. (1991). "Input layer optimization of neural networks by sensitivity analysis and its application to recognition of numerals". *Transactions of the Institute of Electrical Engineers Japan*, 111(1):36-44.
- Taylor, P.C. y Silverman, B.W. (1993). "Block diagrams and splitting criteria for classification trees". *Statistics and Computing*, 3(4):163-167.
- Tsaih, R. (1999). "Sensitivity analysis, neural networks, and the finance". En IEEE (Ed.), *International Joint Conference on Neural Networks* (3830-3835). Piscataway, NJ: IEEE.
- Urquiza, D. y Mendivil, F. (2011). "Aplicación de Redes Neuronales Artificiales para el Análisis de la Inflación en Bolivia". *4º Encuentro de Economistas de Bolivia*.
- Van Ooyen, A. y Nienhuis, B. (1992). "Improving the convergence of the backpropagation algorithm". *Neural Networks*, 5:465-471.
- Vanrell, J.A. (2011). *Un Modelo de Procesos para Proyectos de Explotación de Información*. Tesis Doctoral, Universidad Tecnológica Nacional, Argentina.
- Vapnik, V. (1998). *Statistical Learning Theory*. Jhon Wiley and Sons, New York.
- Vapnik, V. (2000). *The Nature of Statistical Learning Theory*. New York, Ed. Springer Verlag.
- Vicino, F. (1998). "Some reflections on artificial neural networks and statistics: two ways of obtaining solutions by working with data". *Substance Use & Misuse*, 33(2):221-231.
- Virseda, F. y Román, J. 2006. "Minería de Datos". *Working papper*, Universidad Carlos III. <http://www.it.uc3m.es/jvillena/irc/practicas/06-07/22.pdf>
- Wang, J., Xu, M., Wang, H. y Zhang, J. (2006). "Clasification of Imbalanced Data by Using the SMOTE Algorithm and locally Linear Embedding". En: *ICSP*, 3:16-20.
- Wilson, D.L. (1972). "Asymptotic properties of nearest neighbor rules using edited data", En *IEEE Transactions on Systems, Man and Cybernetics*. IEEE Computer Society Press, Los Alamos.

- Witten, I.H.; Frank, E. y Hall, M.A. (2016). *Data Mining. Practical Machine learning. Tools and Techniques*. 4^a edición, Ed. Morgan Kaufmann Publishers.
- Wong, A.K.C. y Chiu, D.K.Y. (1987). "Synthesizing statistical knowledge from incomplete mixed-mode data". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(6):205-218.
- Yeung, D.S., Cloete, I., Shi, D., y Ng, W.W. (2010). *Sensitivity Analysis for Neural Networks*. Natural Computing.
- Yoon, Y., Swales, G. y Margavio, T.M. (1993). "A comparison of discriminant analysis versus artificial neural networks". *Journal of the Operational Research Society*, 44(1):51-60.
- Yoon, Y.O., Brobst, R.W., Bergstresser, P.R. y Peterson, L.L. (1989). "A desktop neural network for dermatology diagnosis". *Journal of Neural Network Computing*, 1:43-52.
- Zhang, J. y Mani, I. (2003). "kNN approach to unbalanced data distributions: a case study involving information extraction". En ICML: *Workshop on Learning from Imbalanced Dataset II*.
- Zurada, J.M., Malinowski, A. y Cloete, I. (1994). "Sensitivity analysis for minimization of input data dimension for feedforward neural network". En IEEE (Ed.), *Proceedings of IEEE International Symposium on Circuits and Systems* (447-450). New York: IEEE.

El libro pretende hacer una revisión de las técnicas más habituales en Data Science, desde los modelos estadísticos de regresión lineal hasta aquellos más avanzados de redes neuronales, pasando por algoritmos y modelos básicos en el análisis de datos.

Todo ello se plasma de manera práctica, utilizando como herramienta de análisis R, que es un lenguaje de programación de común utilización en el ámbito de las ciencias sociales, al que, además, se le dedica un tema introductorio para que el estudiante se maneje con facilidad.

De esta forma, el manual no solo contiene los aspectos teóricos de los diferentes modelos estadísticos, sino también ejemplos de carácter práctico, sobre bases de datos de referencia y de fácil de acceso en Internet, que permiten al lector tener una visión clara y útil de aquello en lo que previamente ha profundizado.

Los cuatro autores del libro son directores de la formación modular de Experto, Especialización y Máster en Big Data y Data Science Aplicados a la Economía y Administración de la Empresa, que se ha convertido en un gran banco de pruebas y experiencia para la gestión de este manual.

De igual forma, los cuatro autores son doctores, habiendo aplicado estas técnicas de análisis en las diferentes investigaciones que han realizado, así como en los proyectos en los que han participado.

Los cuatro autores forman un grupo heterogéneo y multidisciplinar, en función de su formación y ocupación. **Juan Antonio Vicente** y **Julio González** son profesores de la UNED, con formación en Estadística y Finanzas respectivamente. Por su parte, **Francisco Parra** y **Mauricio Beltrán** han desarrollado (y continúan en la actualidad) su labor en diferentes administraciones en el área de tratamiento y procesamiento de datos, así como en institutos de estadística.



Juan del Rosal, 14
28040 MADRID
Tel. Dirección Editorial: 913 987 521