

Fundamentos de ciencia de datos con R

Gema Fernández-Avilés y José-María Montero

2023-05-30

Índice general

Prefacio	5
¡Hola mundo!	5
¿Por qué este libro?	6
¿A quién va dirigido?	7
El paquete CDR	8
¿Por qué R?	8
Agradecimientos	9
1. Modelos aditivos generalizados	11
1.1. Introducción	11
1.2. Splines con penalizaciones	12
1.3. Aspectos metodológicos	13
1.4. Procedimiento con R: la función <code>gam()</code> del paquete <code>mgcv</code>	16
1.5. Casos prácticos	17
2. Modelos mixtos	29
2.1. Conceptos básicos	29
2.2. Formulación del modelo con efectos aleatorios o modelos mixtos	33
2.3. Procedimiento con R para ajustar modelos mixtos	35
2.4. Caso práctico	35
3. Modelos <i>sparse</i> y métodos penalizados de regresión	49
3.1. Introducción	49
3.2. Selección del mejor subconjunto	50
3.3. Métodos <i>shrinkage</i>	56

Prefacio

¡Hola mundo!

El siglo XXI está siendo testigo de grandes cambios vertiginosos en el contexto social y tecnológico, entre otros. Los tiempos han cambiado, la sociedad se ha globalizado y “exige” respuestas inmediatas a problemas muy complejos. Vivimos en el mundo de la **información**, de los **datos**, o mejor, de las **bases de datos masivas**, y los ciudadanos y, sobre todo, las empresas y los gobiernos, dirigen su mirada hacia el mundo científico para que les ayude a “**oír las historias**” que cuentan esos datos acerca de la realidad de la que han sido extraídos. Y dado su enorme volumen y sofisticación (en el nuevo mundo las imágenes y los textos, por ejemplo, también son datos), exigen algoritmos de nueva generación en el campo del *machine learning*, o incluso del *deep learning*, para “oír las historias” que cuentan. No parecen mirar al “antiguo” investigador científico, sino al “nuevo” *científico de datos*.

Ello, inevitablemente, se traduce en la necesidad de profesionales con una gran capacidad de adaptación a este nuevo paradigma: los científicos de datos, también llamados por algunos los “nuevos hombres del Renacimiento”, para lo cual las Universidades y demás instituciones educativas especializada se apresuran a incluir el grado de Ciencia de Datos en su oferta educativa y a ofrecer seminarios de software estadístico de acceso abierto para sus estudiantes de primeros cursos.

Con la emergencia de la nueva sociedad, en la que el manejo de la ingente cantidad de información que genera se hace absolutamente necesario para circular por ella, la **Ciencia de Datos** ha venido para quedarse. Sin embargo, el mundo de la Ciencia de Datos es cualquier cosa menos sencillo. En él, cualquier ayuda, cualquier guía es bienvenida. Por ello, es muy recomendable que la persona que se quiera introducir en él, sea con fines de investigación o con fines profesionales, se agarre de la mano de un guía especializado que le lleve, de una manera amena, comprensible y eficiente, desde el planteamiento de su problema y la captura de la información necesaria para poderle dar una solución, hasta la redacción de las conclusiones finales que ha obtenido con los modernos informes reproducibles colaborativos. Y como en la parte central de ese camino tendrá que luchar con grandes gigantes (en la actualidad denominados técnicas estadísticas y algoritmos), el guía tendrá que explicarle, de manera sencilla y amena, en qué consiste la lucha (las técnicas y los algoritmos) y cómo llegar a la victoria lo más rápido posible, enseñándole a moverse por el mundo del software estadístico, en nuestro caso **R**, que le permitirá realizar los cálculos necesarios para vencer al problema planteado a una velocidad vertiginosa.

En resumen, la información masiva y el moderno tratamiento estadístico de la misma son la “mano invisible” que gobierna la sociedad del siglo XXI, y este manual pretende ser el guía anteriormente mencionado que le llevará de la mano cuando quiera caminar por ella.

¿Por qué este libro?

Lo dicho anteriormente ya justifica por sí solo la aparición de este manual. Afortunadamente, no es el primero en la materia, pues son ya bastantes los materiales de calidad publicados sobre Ciencia de Datos. Sin embargo, quizás, éste pueda ser considerado el más completo. Y ello por varias razones.

La primera es su **completitud**: este manual lleva de la mano al lector desde el planteamiento del problema hasta el informe que contiene la solución al mismo; o desde no saber qué hacer con la información de la que dispone, hasta ser capaz de transformar tales bases de datos masivas, y casi imposibles de manejar, en respuestas a problemas fundamentales de una empresa, institución o cualquier agente social.

La segunda es su **amplitud temática**:

- (I) Parte de las dos primeras preguntas que un neófito se puede hacer sobre esta temática: ¿qué es eso de la Ciencia de Datos que está en boca de todos? Y, ¿qué diablos es **R** y cómo funciona?
- (II) Enseña cómo moverse en la jungla del *Big Data* y de los “nuevos” tipos de datos, siempre bajo el paraguas de la ética de los datos y del buen gobierno de dichos datos.
- (III) Muestra al lector cómo obtener conocimiento de la oscuridad del enorme banco de información a su disposición, que no sabe cómo abordar ni manejar.
- (IV) No deja a nadie atrás, y de forma previa al contenido central del manual (las técnicas de Ciencia de Datos), incluye unas breves, pero magníficas, secciones sobre los rudimentos de la probabilidad, la inferencia estadística y el muestreo, para aquéllos no familiarizados con estas cuestiones.
- (V) Aborda una treintena de técnicas de Ciencia de Datos en el ámbito de la modelización, análisis de datos cualitativos, discriminación, *machine learning* supervisado y no supervisado, con especial incidencia en las tareas de clasificación y clusterización -así como, en el caso no supervisado, de reducción de la dimensionalidad, escalamiento multidimensional y análisis de correspondencias-, *deep learning*, análisis de datos textuales y de redes, y, finalmente, ciencia de datos espaciales (desde las perspectivas de la geoestadística, la econometría espacial y los procesos de punto).
- (VI) Hace especial hincapié en la reproducibilidad en tiempo real (o no) entre los distintos miembros de un equipo (sea universitario, empresarial, o del tipo que sea) y en la difusión de los resultados obtenidos, enseñando al lector cómo generar informes reproducibles mediante RMarkdown y documentos Quarto o en otros modernos formatos.
- (VII) Dedica un capítulo a la creación de aplicaciones web interactivas (con Shiny).

Índice general

7

- (viii) Para aquéllos con pasión por la codificación, y que quieran compartir código y colaborar con otros desarrolladores, este manual aborda la gestión rápida y eficaz de proyectos (del tamaño que sean) mediante Git, un sistema de control de versiones distribuido, gratuito y de código abierto, y GitHub, un servicio de alojamiento de repositorios Git del cual, aquellos no familiarizados con la cuestión de la codificación, o con aversión a ella, podrán tomar el código que necesitan.
- (ix) Muestra al lector los primeros pasos para iniciarse en el geoprocесamiento en la nube.
- (x) Y, finalmente, aborda más de una docena de casos de uso (en medicina, periodismo, economía, criminología, marketing, moda, demanda de electricidad, cambio climático, reconocimiento de patrones en la forma de tuitear...) que ilustran la puesta en práctica de todos los conocimientos anteriormente adquiridos.

La cuarta razón es que todo lo que el lector aprende en este manual lo puede reproducir y poner en práctica inmediatamente con **R**, puesto que el manual está trufado de *chunks* (o trozos de código **R**) que no tiene más que cortar y pegar para reproducir los ejemplos que se muestran en el libro, cuyos datos están en el paquete CDR; o utilizar dichas *chunks* para abordar el problema que le ocupa con los datos que tenga a su disposición. Una buena razón, sin duda. Por consiguiente, el manual es una buena combinación “teoría-práctica-software” que permite abordar cualquier problema que el científico de datos se plante en cualquier disciplina o situación empresarial, médica, periodística...

La quinta es su **variedad de perspectivas**. Son **más de 40 los participantes** en este manual. Algunos de ellos, prestigiosos profesores universitarios; otros, destacados miembros de instituciones públicas; otros, CEOs de empresas en la órbita de la ciencia de datos; otros, *big names* del mundo de **R** software... El manual es, sin duda, un magnífico ejemplo de colaboración Universidad-Empresa para buscar soluciones a los problemas de las sociedades modernas.

¿A quién va dirigido?

Fundamentos de ciencia de datos con R está dirigido a todos aquellos que desean desarrollar las habilidades necesarias para abordar proyectos complejos de Ciencia de Datos y “pensar con datos” (como lo acuñó Diane Lambert, de Google). El deseo de resolver problemas utilizando datos es su piedra angular. Por tanto, como se avanzó anteriormente, este manual no deja a nadie atrás, y lo único que requiere es “el deseo de resolver problemas utilizando datos”. No excluye ninguna disciplina, no excluye a las personas que no tengan un elevado nivel de análisis estadístico de datos, no excluye a nadie. Se ha procurado una combinación de rigor y sencillez, y de teoría y práctica, todo ello con sus correspondientes códigos en **R**, que satisfaga tanto a los más exigentes como a los principiantes.

También está destinado a todos aquellos que quieran sustituir la navegación por la web (la búsqueda del video, publicación de blog o tutorial *online* que solucione su problema –frustración tras frustración por la falta de consistencia, rigor e integridad de dichos materiales, así como por su sesgo hacia paquetes singulares para la implementación de las cuestiones que tratan–), por

una “**biblia de la ciencia de datos**” rigurosa pero sencilla, práctica y de aplicación inmediata sin ser ni un experto estadístico ni un experto informático.

Pero si a alguien está destinado especialmente, es a la comunidad hispano hablante. Este manual es un guiño a dicha comunidad, para que tenga a su disposición, en su lengua nativa, uno de los mejores manuales de Ciencia de Datos de la actualidad.

El paquete CDR



El paquete **CDR** contiene la mayoría de conjuntos de datos utilizados en este libro que no están disponibles en otros paquetes. Para instalarlo use la función `install_github()` del paquete **remotes**.

```
# este comando sólo necesita ser ejecutado una vez
# si el paquete remotes no está instalado, descomentar para instalarlo

# install.packages("remotes")
remotes::install_github("cdr-book/CDR")
```

La lista de todos los conjuntos de datos puede obtenerse haciendo `data()`.

```
library('CDR')
data(package = "CDR")
```

Este paquete ayudará al lector a reproducir todos los ejemplos del libro. De acuerdo con las mejores prácticas en **R**, el paquete **CDR** sólo contiene los datos utilizados en el libro.

¿Por qué R?

R es un lenguaje de código abierto para computación estadística que se ha consolidado entre la comunidad científica internacional, en las últimas dos décadas, como una herramienta de primer

Índice general

9

nivel, consolidándose como líder permanente en el ámbito de la implementación de metodologías estadísticas para el análisis de datos. La utilidad de **R** para la Ciencia de Datos deriva de un fantástico ecosistema de paquetes (activo y en crecimiento), así como de un buen elenco de otros excelentes recursos: libros, manuales, *blogs*, foros y *chats* interactivos en las redes sociales, y una gran comunidad dispuesta a colaborar, a orientar y a resolver diferentes cuestiones relacionadas con **R**.

Por otra parte, **R** es el lenguaje estadístico y de análisis de datos más utilizado en la mayoría de los entornos académicos y, cómo no, por una larga lista de importantes empresas, entre las que se cuentan Facebook (análisis de patrones de comportamientos relacionado con actualizaciones de estado e imágenes de perfil), Google (para la efectividad de la publicidad y la previsión económica), Twitter (visualización de datos y agrupación semántica), Microsoft (adquirió la empresa Revolution R), Uber (análisis estadístico), Airbnb (ciencia de datos), IBM (se unió al grupo del consorcio R), New York Times (visualización)...

La comunidad **R** también es particularmente generosa e inclusiva, y hay grupos increíbles, como *R-Ladies* y *Minority R Users*, diseñados para ayudar a garantizar que todos aprendan y usen las capacidades de **R**.

Agradecimientos

No queremos dar por finalizado este prefacio sin agradecer a los 44 autores participantes en esta obra su esfuerzo por condensar, en no más de 20 páginas, la teoría, práctica y tratamiento informático de la parte de la Ciencia de Datos que les fue encargada. Y no sólo eso; el “más difícil todavía” fue que debían dirigirse a un abanico de potenciales lectores tan grande como personas haya con “el deseo de resolver problemas utilizando datos”. Era misión imposible. Sin embargo, a la vista del resultado, ha sido misión cumplida. El esfuerzo mereció la pena.

Además, nos gustaría agradecer el apoyo incondicional recibido por (en orden alfabético): Itzcoatl Bueno, Ismael Caballero, Emilio L. Cano, Diego Henangómez, Ricardo Pérez, Manuel Vargas y Jorge Velasco.

También queremos poner de manifiesto que la edición de este texto ha sido financiada por diversos entes de la Universidad de Castilla-La Mancha. En su mayor parte, por el **Máster en Data Science y Business Analytics (con R software)** (a través de la orgánica: 02040M0280), pero también por la Facultad de Ciencias Jurídicas y Sociales de Toledo (a través de su contrato programa: orgánica 00440710), el Departamento de Economía Aplicada I (mediante sus fondos departamentales, DEAI 00421I126) y el Grupo de Investigación Economía Aplicada y Métodos Cuantitativos (que ha dedicado parte de sus fondos a la edición de esta obra, orgánica 01110G3044-2023-GRIN-34336).

A todos, eternamente agradecidos por ayudarnos en este reto de transformar la oscuridad en conocimiento, de convertir en una ciencia y en un arte la difícil tarea de sacar valor de los datos, el petróleo del futuro. Quizás en este momento no seamos conscientes de que hemos puesto nuestro granito de arena a la ciencia que, a buen seguro, juegue uno de los papeles más importantes de este siglo, caracterizado por el predominio de la información. Una ciencia, la Ciencia de Datos, que combina el análisis estadístico de datos, la algoritmia y el conocimiento del

negocio para sacar valor del bien más abundante de la sociedad en la que vivimos: la información. Una disciplina cuyo dominio caracteriza a los científicos de datos (también denominados los nuevos personajes del Renacimiento), profesión que ya fue calificada hace más de veinte años en la *Harvard Business Review* y en *The New York Times*, entre otros, como la “más sexy del siglo XXI”.

Nota

Este manual está publicado por [McGraw Hill](#). Las copias físicas están disponibles en [McGraw Hill](#). La versión *online* se puede leer de forma gratuita en <https://cdr-book.github.io/> y tiene la [licencia de Creative Commons Reconocimiento-NoComercial-SinObraDerivada 4.0 Internacional](#).

Si tiene algún comentario o sugerencia, no dude en contactar con los editores y los autores. ¡Gracias!

Capítulo 1

Modelos aditivos generalizados

María Durbán^a y Víctor Casero-Alonso^b

^aUniversidad Carlos III de Madrid ^bUniversidad de Castilla-La Mancha

1.1. Introducción

Los modelos lineales, o los lineales generalizados (GLM) vistos en los capítulos ?? y ?? tienen la ventaja de ser fáciles de ajustar e interpretar. Además se dispone de técnicas para contrastar las hipótesis del modelo. Sin embargo, cuando la variable respuesta no está relacionada de forma lineal con las variables explicativas no tiene sentido utilizar modelos lineales (generalizados o no) y hay que acudir a modelos que flexibilicen esta relación, que, en el caso de una única variable explicativa, se puede expresar como sigue:

$$Y = \beta_0 + f(X) + \varepsilon.$$

Puede que la función $f()$ sea conocida de antemano, como ocurre en muchos modelos biológicos, donde existe una dependencia de tipo exponencial, $f(x) = e^{\beta_0 + \beta_1 x}$. En otras ocasiones dicha función es desconocida y se puede utilizar una aproximación. Por ejemplo, la muy utilizada regresión polinómica:

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \dots + \beta_p X^p + \varepsilon, \quad \varepsilon \sim N(0, \sigma^2). \quad (1.1)$$

Sin embargo, la regresión polinómica tiene un gran inconveniente, que no se lleva a cabo de forma local: cada vez que se cambia un coeficiente del modelo, el cambio impacta a los valores ajustados en todo el rango de la variable explicativa. Sin embargo, es posible utilizar técnicas (como las que presentamos en este capítulo) en las que el valor predicho en un punto dado sólo depende de las observaciones en ese punto y de las observaciones vecinas, es decir, el ajuste se lleva a cabo de forma local.

En el caso de disponer de más de una variable explicativa, la extensión del modelo de regresión múltiple sería el **modelo aditivo** (en el caso de variable respuesta gaussiana), donde no se asume que la relación entre Y y cada una de las variables explicativas tenga que ser lineal:

$$Y = f(X_1) + \dots + f(X_j) + \epsilon, \quad \epsilon \sim N(0, \sigma^2). \quad (1.2)$$

Las funciones f incluyen también a las funciones lineales vistas en el Capítulo ?? . Los modelos aditivos generalizados, GAM, extienden el modelo anterior a respuestas no gaussianas, como lo hacen los GLMs respecto de los modelos lineales con respuesta gaussiana (véase el Capítulo 17).

1.2. Splines con penalizaciones

Las funciones de la Eq. (1.2) se estimarán mediante técnicas de suavizado o *smoothers*, cuyo objetivo extraer las tendencias (o señales) que existen en la relación entre la variable respuesta y las variables explicativas, sin presuponer una forma funcional a priori entre las mismas; sólo se asume que la relación entre Y y X es suave (tiene poco ruido). Las predicciones obtenidas mediante estas técnicas tienen menos variabilidad que Y , por eso se le llama suavizador (la regresión lineal es un suavizador llevado al extremo). Estas son algunas de las técnicas de suavizado existentes:

1. Regresión polinomial local con pesos, *lowess*.
2. Kernels.
3. Splines.

Este capítulo se centra en uso de los Splines ya que es la técnica de suavizado más utilizada. Los splines son funciones polinómicas a trozos de la variable explicativa, que se unen en puntos llamados nodos. Existen muchos tipos diferentes de splines (naturales, cílicos, B-splines, O-splines, etc.), nosotros nos centraremos en el uso de los splines con penalizaciones (P-splines). Estos se basan en: i) hacer una aproximación de la función f mediante una base de funciones y, ii) añadir una penalización a la hora de estimar el modelo de manera que se pueda controlar la variabilidad de la curva que se quiere estimar. Hay muchas maneras de representar una función a través de una base (un ejemplo sencillo de una base de funciones sería el caso de regresión polinómica en la que la base de funciones es decir, la matriz de regresión, sería una matriz cuyas columnas son las potencias de la variable explicativa: $[X : X^2 : \dots : X^p]$). Una de las mejores opciones es el uso de los B-splines (De Boor, 2001) por sus buenas propiedades numéricas. La penalización se añade a la función de verosimilitud y se construye a partir de la derivada de la curva que se quiere penalizar. Generalmente se utilizan penalizaciones de orden 2, lo que implica que se está penalizando todo aquello que no es lineal en la función; es decir, si la penalización es muy grande la curva estimada sería simplemente una línea recta. La penalización está controlada por un parámetro llamado **parámetro de suavizado**.

A la hora de ajustar este tipo de modelos que tomar dos decisiones importantes:

1.3. Aspectos metodológicos

13

- El tamaño de la base (el número de nodos del B-spline): generalmente se utiliza esta regla:

$$\text{número de nodos} = \min\{40, \text{valores únicos de } x/4\} \quad (1.3)$$

(por ejemplo si se tienen 100 observaciones, se elegirían $100/4=25$ nodos).

- El parámetro de suavizado: la selección del parámetro de suavizado se puede hacer mediante distintos métodos, validación cruzada, validación cruzada generalizada, etc. El método que da lugar a mejores estimaciones de las curvas es el método de máxima verosimilitud restringida (Henderson, 1953), abreviado como REML, y es el más recomendable. La Fig. 1.1 muestra el impacto que el parámetro de suavizado tiene en el ajuste final de la curva (los datos corresponden al dataset `fossil` del paquete `Semipar`).

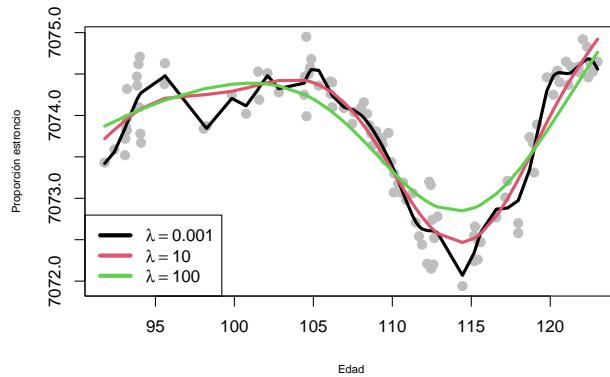


Figura 1.1: Regresión con P-splines para diferentes valores del parámetro de suavizado.

1.3. Aspectos metodológicos

Al igual que en el caso de los modelos lineales y los modelos GLM, en los modelos GAM es necesario conocer algunos aspectos metodológicos que son fundamentales para llevar a cabo un ajuste correcto de los modelos y entender los resultados obtenidos en el ajuste. A continuación se muestran los más relevantes:

1.3.1. Estimación de los parámetros del modelo

La estimación de los modelos GAM se lleva a cabo mediante máxima verosimilitud penalizada. Supóngase el caso de una sola variable explicativa y que se quiere ajustar el modelo:

$$Y = f(X) + \epsilon.$$

Como se comentó anteriormente, los modelos GAM tiene como punto de partida la aproximación de la función a estimar mediante una matriz formada por B-splines, es decir, se busca transformar el modelo lineal o lineal generalizado tradicional de tal forma que $f(X)$ sea el producto de una matriz multiplicada por unos coeficientes (esa matriz está formada por los B-splines); es decir, se elige una *base* (una matriz \mathbf{B}) que permita escribir la función $f(X)$ como una combinación lineal de sus elementos (los elementos de esta base son conocidos ya que se calculan a partir de las variables explicativas):

$$f(X) = \sum_{l=1}^k b_l(X)\theta_l,$$

donde $b_l(X)$ son las funciones B-spline que componen la base. En forma matricial:

$$f(X) = \mathbf{B}\boldsymbol{\theta}.$$

Los parámetros $\boldsymbol{\theta}$ se estiman minimizando la siguiente expresión (en el caso de datos Gaussianos los mínimos cuadrados penalizados son equivalentes a la verosimilitud penalizada):

$$(\mathbf{y} - \mathbf{B}\boldsymbol{\theta})'(\mathbf{y} - \mathbf{B}\boldsymbol{\theta}) + \lambda\boldsymbol{\theta}'\mathbf{P}\boldsymbol{\theta},$$

donde \mathbf{P} es la matriz de penalización y λ es el parámetro de suavizado. Dado un valor del parámetro de suavizado, las estimaciones de los parámetros vienen dadas por:

$$\hat{\boldsymbol{\theta}} = (\mathbf{B}^T\mathbf{B} + \lambda\mathbf{P})^{-1}\mathbf{B}'\mathbf{y}, \quad (1.4)$$

y las estimaciones de la variable respuesta se obtienen como: $\hat{\mathbf{y}} = \underbrace{\mathbf{B}(\mathbf{B}^T\mathbf{B} + \lambda\mathbf{P})^{-1}\mathbf{B}'\mathbf{y}}_{\mathbf{H}}$. La matriz \mathbf{H} juega un papel importante, ya que la suma de su diagonal da una idea de la complejidad de la curva ajustada (la curva más compleja sería la que interpola los datos), dicha suma es lo que se llaman los *grados de libertad efectivos* (que no se corresponden con el número de parámetros ajustados).

1.3.2. Inferencia sobre las funciones suaves

Para saber si la relación estimada entre Y y X es o no estadísticamente significativa, se debe proceder al contraste:

$$\begin{aligned} H_0 : f(X) &= 0 && \text{(no efecto)} \\ H_1 : f(X) &\neq 0 && \text{(efecto)}, \end{aligned}$$

dado que la función $f(X)$ depende de los coeficientes que acompañan a las bases de B-splines, el contraste anterior es equivalente al contraste:

$$\begin{aligned} H_0 : \boldsymbol{\theta} &= 0 \\ H_1 : \boldsymbol{\theta} &\neq 0. \end{aligned}$$

1.3. Aspectos metodológicos

15

La distribución del estadístico de contraste dependerá de si la variable respuesta sigue una distribución Normal o no: si los datos son Normales, el estadístico de contraste sigue un distribución F . En otro caso sigue una distribución χ^2 .

Comparación de modelos Cuando se trabaja con un modelo aditivo (1.2), en el que hay más de una variable explicativa, puede ser de interés comparar versiones de ese modelo que contengan distintos conjuntos de variables. La comparación dependerá de la relación entre los modelos a comparar:

1. **modelos anidados.** La comparación se basa, al igual que en los GLM, en la diferencia en la *deviance residual*. Si se quieren comparar dos modelos m_1 y m_2 (donde $m_1 \subset m_2$), entonces:

- En el caso de variable respuesta Normal:

$$\frac{(DR(m_1) - DR(m_2))/(df_2 - df_1)}{DR(m_2)/(n - df_2)} \approx F_{(df_2 - df_1), (n - df_2)},$$

donde DR es la deviance residual (suma de cuadrados residual) y df son los grados de libertad asociados con cada modelo. - En otro caso:

$$DR(m_1) - DR(m_2) \approx \chi^2_{df_2 - df_1}.$$

2. **Modelos no anidados.** En este caso los contrastes anteriores no son válidos y se utilizarán criterios basados en el AIC (Criterio de Información de Akaike).

1.3.3. Suavizado multidimensional y para datos no Gaussianos

Para el suavizado penalizado en 2 dimensiones (o más) también se necesita una base y una penalización. El modelo sería:

$$\mathbf{Y} = f(\mathbf{X}_1, \mathbf{X}_2) + \epsilon,$$

donde $f()$ es una función de las dos covariables \mathbf{X}_1 y \mathbf{X}_2 . Dicha función se aproxima mediante el producto tensorial de las bases de B-splines marginales para cada una de las covariables y la penalización dependerá de dos parámetros de suavizado. Los términos de suavizado multidimensional se pueden combinar con términos unidimensionales y términos lineales. En este caso la penalización dependería de dos parámetros de suavizado (uno para cada covariable).

La extensión de los modelos de suavizado al caso en el que la variable respuesta es no Gaussiana, se hace de forma similar al caso lineal, cuando se pasa de un modelo de regresión lineal a un GLM. Al igual que en el caso de los GLMs $g(\boldsymbol{\mu}) = \boldsymbol{\eta} = f(\mathbf{X}) = \mathbf{B}\boldsymbol{\theta}$, y se añade la penalización a la función de verosimilitud de la distribución correspondiente:

$$\ell_p(\boldsymbol{\theta}) = \ell(\boldsymbol{\theta}) + \lambda \boldsymbol{\theta}^T P \boldsymbol{\theta},$$

donde $\ell(\beta)$ es la log-verosimilitud.

1.4. Procedimiento con R: la función `gam()` del paquete `mgcv`

Aunque hay muchas librerías disponibles, la principal es `mgcv`, que implementa una gran variedad de modelos de suavizado a través de la función `gam()` (generalized additive models)¹.

```
gam(formula, method="", select="", family=gaussian())
```

- `formula` es el argumento principal de esta función; es la ecuación del modelo: por ejemplo, `y ~ x1+x2+s(x3)`.
 - Lo primero que se tiene que elegir es la base a utilizar para representar las funciones suaves, `s(x)` (ver `?s` o `?smooth.terms`), o `te(x1,x2)` en el caso de suavizado bidualimensional. Por defecto se usan los llamados thin plate splines. El tipo de base usada se puede modificar utilizando el argumento `bs` dentro de `s(x, bs = "ps")`, en este caso `ps` indica el uso de B-splines con penalizaciones. A continuación se describen otras alternativas:

<code>bs</code>	Descripción
<code>tp</code>	Thin Plate Regression Splines
<code>ts</code>	Thin Plate Regression Splines con regularización
<code>cr</code>	Spline cúbicos de regresión
<code>crs</code>	Spline cúbicos de regresión con regularización
<code>cc</code>	Spline cílicos
<code>ps</code>	P-splines

- `m` indica el orden de la penalización; por defecto es 2.
- `k` es el número de nodos para construir la base. El número por defecto suele ser demasiado bajo por lo que siempre se recomienda que el usuario elija el número utilizando la regla dada en (1.3).
- `by` se igualará a una variable numérica o factor de la misma dimensión de cada covariante para hacer interacciones entre curvas y variables.
- `id` se utiliza para forzar que diferentes términos suaves utilicen la misma base y la misma cantidad de suavizado.
- `method` selecciona método para estimar los parámetros de suavizado, se puede elegir entre: REML (máxima verosimilitud restringida), ML (máxima verosimilitud), GCV.Cp (validación cruzada generalizada), GACV.Cp (validación cruzada aproximada generalizada). En la práctica, como se indicó anteriormente, se prefiere REML.
- `family` permite elegir la distribución de la variable respuesta (binomial, Poisson, etc.); por defecto asume Gaussiana.
- `select=TRUE` contrasta si una variable debe entrar o no en el modelo.

¹La principal referencia para esta sección es el libro de Wood (2006).

1.5. Casos prácticos

En este apartado se verán una serie de aplicaciones que permiten mostrar los diferentes usos de este tipo de modelos.

1.5.1. Modelo unidimensional con `fossil`

Se empezará ilustrando el uso de la función `gam()` con el conjunto de datos `fossil` del paquete `SemiPar`. El objetivo es estimar la relación entre la edad de los fósiles y la proporción de isotopos de estroncio.

```
library("SemiPar")
data(fossil)
Y <- 10000*fossil$strontium.ratio
X <- fossil$age
plot(X,Y, xlab="Edad", ylab = "Proporción de estroncio")
```

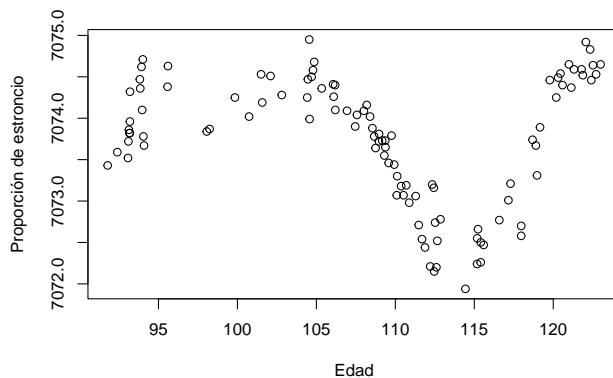


Figura 1.2: Edad de fósiles con respecto a la proporción de isótopos de estroncio

A la vista de la Fig. 1.2, es claro que se necesita ajustar una curva (y no una línea) para estimar la relación entre ambas variables. Para ello se utiliza la función `gam()` que devuelve un objeto de tipo "gam", y que se puede usar con las típicas funciones `print()`, `summary()`, `fitted()`, `plot()`, `residuals()`, etc.

```
library("mgcv")
fit_gam <- gam(Y ~ s(X,k=25,bs="ps"), method="REML", select=TRUE)
# se eligen 25 nodos ya que se la variable tiene 106 observaciones
summary(fit_gam)
#>
```

```
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> Y ~ s(X, k = 25, bs = "ps")
#>
#> Parametric coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 7.074e+03 2.435e-02 290504 <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>          edf Ref.df   F p-value
#> s(X) 10.22     24 35.89 <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.891  Deviance explained = 90.2%
#> -REML = 23.946  Scale est. = 0.062849 n = 106
```

Como se puede ver, se ha especificado la relación entre la variable respuesta (Y , proporción de estroncio) y la variable explicativa (X , edad) mediante un *spline*, `s()`, de tipo penalizado, `ps`, con 25 nodos, se ha especificado REML como método para estimar el parámetro de suavizado (los parámetros del spline se estiman también mediante REML, ya que da lugar a las mismas estimaciones que máxima verosimilitud).

En la primera parte de la salida anterior aparecen los términos que entran linealmente en el modelo (en este caso solo la ordenada en el origen), y en la parte de abajo los términos de suavizado. Como se indicó anteriormente, dado que se ha usado `select=TRUE`, se está contrastando si la variable `edad` debe entrar en el modelo o no. En este caso, es claro que ha de entrar ya que el p-valor de `s(x)` es pequeño y los grados de libertad asociados son aproximadamente 10, lo que indica que la relación entre Y y X está lejos de la linealidad.

La función `gam.check()` devuelve los gráficos de residuos usuales (residuos frente a valores ajustados, gráficos de cuantiles para comprobar la normalidad, etc.), pero además proporciona información sobre el proceso de ajuste del modelo.

```
gam.check(fit_gam, cex=1.2)
```

```
#>
#> Method: REML Optimizer: outer newton
#> full convergence after 5 iterations.
#> Gradient range [-4.557319e-06,5.900236e-06]
#> (score 23.94602 & scale 0.06284944).
#> Hessian positive definite, eigenvalue range [4.557347e-06,53.03185].
```

1.5. Casos prácticos

19

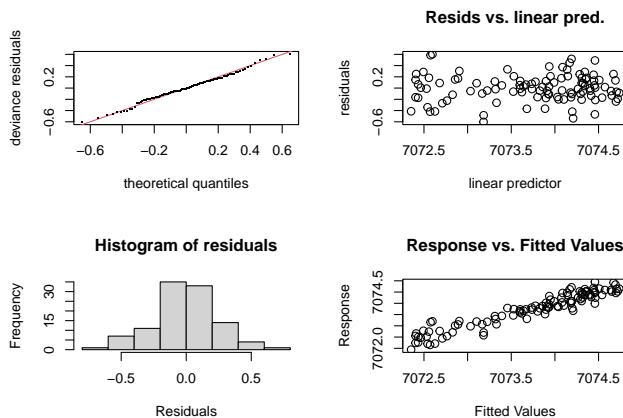


Figura 1.3: Gráficos de residuos obtenidos con ‘gam.check()’

```
#> Model rank = 25 / 25
#>
#> Basis dimension (k) checking results. Low p-value (k-index<1) may
#> indicate that k is too low, especially if edf is close to k'.
#>
#>      k'  edf k-index p-value
#> s(X) 24.0 10.2    1.03   0.61
```

El test que aparece en la parte de abajo está contrastando si el número de nodos elegido es suficiente. Si el valor de **k** está muy próximo a **edf** entonces se debería reajustar el modelo con más nodos.

El comando **plot()** permite dibujar la función suave que relaciona Y con X. La curva estimada que aparece en la Fig. 1.4 está centrada (la función **plot()** siempre lo hace de esta forma), el argumento **shade** hace que se sombre el intervalo de confianza, y **seWithMean** hace que la incertidumbre sobre la ordenada en el origen se incluya en el cálculo del intervalo de confianza.

```
plot(fit_gam, shade=TRUE, seWithMean=TRUE, pch=19, 1, cex=.55)
```

1.5.2. Modelo aditivo con airquality

En esta sección se analizan de nuevo los datos **airquality** (ver [airquality²](#)), que consisten en 154 medidas de calidad del aire en Nueva York, de Mayo a Septiembre 1973. El objetivo es establecer la relación entre las variables meteorológicas y la cantidad de ozono en la atmósfera.

²Conjunto de datos incluido con la instalación base de R.

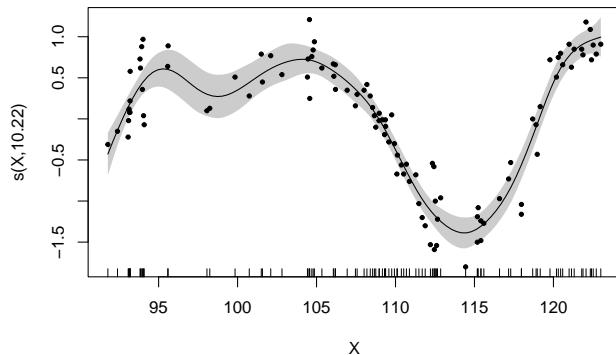


Figura 1.4: Curva ajustada e intervalo de confianza

Ya se ha analizado dicha relación en el Capítulo 16 de Modelos lineales, donde los ajustes lineales realizados eran satisfactorios, pero se encontraban problemas en los residuos del modelo, que no daban validez a dicha modelización. Allí se sugería que la relación entre la variable respuesta y alguna explicativa fuese no lineal. Además, se consideró la transformación logarítmica de la variable `Ozone`, y con dicha trasformación se obtenía una distribución más similar a la distribución Normal.

Por lo tanto se va a ajustar el modelo incluyendo las variables explicativas sin imponer linealidad; en particular, se van a incluir las variables `Wind`, `Temp` y `Solar.R`. Las variable `Wind` y `Temp` tienen sólo 31 y 40 valores únicos respectivamente, aunque el dataset tiene 154 valores; por eso se decide utilizar 10 nodos y no más, mientras que para la variable `Solar.R` se utilizan 20.

```
airq_gam=gam(log(Ozone)~s(Wind,bs="ps",k=10) +
  s(Temp,bs="ps",k=10)+s(Solar.R,bs="ps",k=20),
  method="REML",select=TRUE,data=airquality,na.action=na.omit)
summary(airq_gam)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(Ozone) ~ s(Wind, bs = "ps", k = 10) + s(Temp, bs = "ps",
#>       k = 10) + s(Solar.R, bs = "ps", k = 20)
#>
#> Parametric coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 3.41593    0.04586   74.49   <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

1.5. Casos prácticos

21

```
#>
#> Approximate significance of smooth terms:
#>          edf Ref.df   F p-value
#> s(Wind)    2.318     9 2.255 3.13e-05 ***
#> s(Temp)    1.852     9 6.128 < 2e-16 ***
#> s(Solar.R) 2.145    19 1.397 2.31e-06 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.689  Deviance explained = 70.7%
#> -REML = 86.106  Scale est. = 0.23342 n = 111
```

Los resultados indican que todas las variables son significativas (p-valores pequeños), estando la variable Temp próxima a la linealidad (los grados de libertad asociados a la variable son 1.8). El R^2 ajustado es 0.68, por lo que el modelo ajusta moderadamente bien los datos. La Fig. 1.5 muestra las tres curvas ajustadas, incluyendo los llamados *residuos parciales* que corresponden a, por ejemplo, en el caso del gráfico del viento, $\log(Ozone) - \hat{\beta}_0 - \hat{f}(Temp) - \hat{f}(Solar.R)$, es decir, lo que queda sin explicar después de haber ajustado los demás términos del modelo.

```
library("mgcv")
b <- getViz(airq_gam)
# getViz es otra opción para dibujar los términos de un modelo gam()
print(plot(b, allTerms = T, shade=T), pages = 1)
```

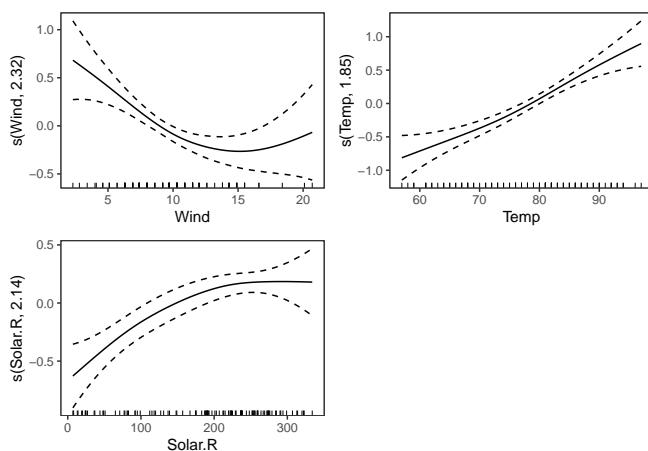


Figura 1.5: curvas estimadas para ‘Wind’, ‘Temp’ y ‘Solar’

1.5.3. Modelo semiparamétrico con onions

Es un caso particular del modelo aditivo, pues en este modelo todas las variables entran de forma lineal excepto una:

$$\mathbf{Y} = \beta_0 + \beta_1 \mathbf{X}_1 + \dots + \beta_{j-1} \mathbf{X}_{j-1} + f(\mathbf{X}_j) + \epsilon.$$

La forma de ajustar el modelo es exactamente igual a la anterior. Pero hay un caso que merece especial interés: cuando en la parte paramétrica se incluye una variable categórica con dos o más niveles. Al igual que en el caso de regresión lineal, se puede plantear si se quieren ajustar dos o más rectas paralelas (modelo aditivo) o no paralelas (modelo con interacción).

Para ilustrar este caso se acude al data.frame `onions` (librería `SemiPar`). Contiene 84 observaciones de un experimento sobre la producción de un tipo de cebolla en dos localidades (Purnong Landing y Virginia). El objetivo es relacionar el logaritmo de la producción de cebollas con la densidad de plantas por metro cuadrado. El modelo lineal básico sería:

$$\log(\text{yield}_i) = \beta_0 + \beta_1 \text{location}_{ij} + \beta_2 \text{dens}_i + \epsilon_i$$

donde

$$\text{location}_{ij} = \begin{cases} 0 & \text{si la observación } i \text{ es de Purnong Landing} \\ 1 & \text{si la observación } i \text{ es de Virginia} \end{cases}$$

Se comienza por ajustar el siguiente modelo:

$$\log(\text{yield}_i) = \beta_0 + \beta_1 \text{location}_{ij} + f(\text{dens}_i) + \epsilon_i$$

```
library("mgcv")
library("SemiPar")
data(onions)
#Se indica a R que la variable location es categórica
onions$location <- factor(onions$location)
#Se recodifica la variable
levels(onions$location) <- c("Purnong Landing", "Virginia")
fit1 <- gam(log(yield) ~ location + s(dens, k=20, bs="ps"),
            method="REML", select=TRUE, data=onions)
summary(fit1)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(yield) ~ location + s(dens, k = 20, bs = "ps")
#>
#> Parametric coefficients:
#>                               Estimate Std. Error t value Pr(>|t|)
#> (Intercept)        4.85011    0.01688 287.39  <2e-16 ***
#> locationVirginia -0.33284    0.02409 -13.82  <2e-16 ***
#> ---
```

1.5. Casos prácticos

23

```
#> Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>          edf Ref.df   F p-value
#> s(dens) 4.568     19 72.76 <2e-16 ***
#> ---
#> Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) = 0.946 Deviance explained = 94.9%
#> -REML = -54.242 Scale est. = 0.011737 n = 84
```

En este ejemplo se ve que en la parte lineal aparecen dos parámetros, ambos significativos, la ordenada en el origen y el parámetro que corresponde a la variable categórica `location`, concretamente el parámetro correspondiente a la categoría Virginia que es negativo indicando que la producción media para la otra localidad es mayor que la producción media en Virginia. El término de suavizado es significativo también.

En este caso, función `plot.gam()` sólo dibujará una curva, pues las curvas para las dos localizaciones son paralelas y la diferencia entre ellas es igual al valor del parámetro de la variable localización. Para dibujar las curvas para cada localización se utiliza la función `plot_smooth()` de la librería `tidymv`. Los argumentos son, primero el modelo, después la variable explicativa y por último la variable categórica.

```
library("tidymv")
library("ggplot2")
plot_smooths(fit1, dens, location) +
  theme(text = element_text(size = 12))
```

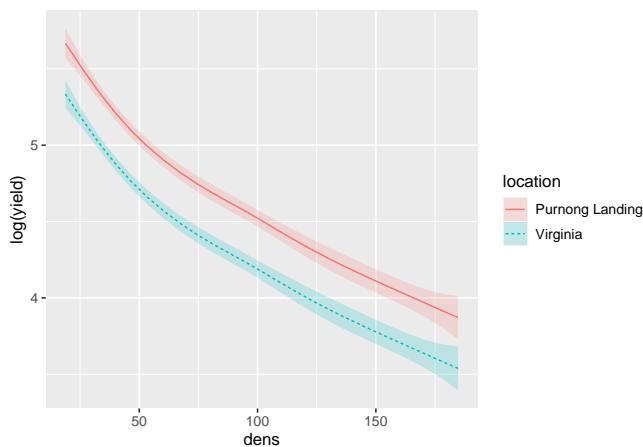


Figura 1.6: curvas ajustadas para ambas localidades

Asumir curvas paralelas para ambas localidades implica que el descenso en producción de cebollas a medida que aumenta la densidad de plantas es el mismo para ambas localidades, y esto no tiene por qué ser cierto. Para relajar esta hipótesis se puede ajustar un modelo con interacción (de manera similar a lo que se hace en el caso de regresión lineal):

$$\log(\text{yield}_i) = \beta_0 + \beta_1 \text{location}_{ij} + f(\text{dens}_i)_{L(j)} + \epsilon_i$$

donde

$$L(j) = \begin{cases} 0 & \text{si la } i\text{-ésima observación es de Purnong Landing} \\ 1 & \text{si la } i\text{-ésima observación es de Virginia} \end{cases}$$

Para hacerlo en R, se introduce el argumento `by=location` dentro de la curva

```
fit2 <- gam(log(yield) ~ location + s(dens, k=20, bs="ps", by=location),
             method="REML", data=onions)
summary(fit2)
#>
#> Family: gaussian
#> Link function: identity
#>
#> Formula:
#> log(yield) ~ location + s(dens, k = 20, bs = "ps", by = location)
#>
#> Parametric coefficients:
#>                               Estimate Std. Error t value Pr(>|t|)
#> (Intercept)           4.84415   0.01603 302.19  <2e-16 ***
#> locationVirginia -0.33018   0.02270 -14.54  <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Approximate significance of smooth terms:
#>                               edf Ref.df      F p-value
#> s(dens):locationPurnong Landing 3.096  3.834 176.9  <2e-16 ***
#> s(dens):locationVirginia       4.742  5.795 153.0  <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> R-sq.(adj) =  0.952  Deviance explained = 95.7%
#> -REML = -58.541  Scale est. = 0.010446 n = 84
```

Ahora aparecen dos términos suaves, uno para cada localidad, de modo que estas curvas no tienen por qué ser paralelas, sino que cada una se ajustará a la forma que tengan los datos. En este caso, la Fig. 1.7, generada de nuevo con `plot_smooths`, muestra como las curvas se van alejando a medida que aumenta la densidad de plantas.

Para finalizar se comparan ambos modelos con el criterio AIC.

```
AIC(fit1); AIC(fit2)
#> [1] -125.2307
#> [1] -131.2181
```

1.5. Casos prácticos

25

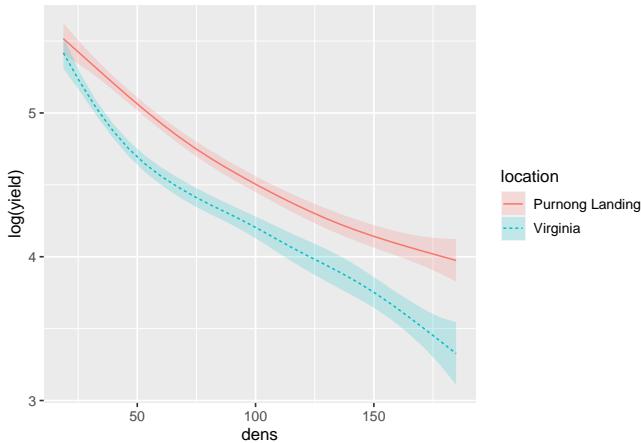


Figura 1.7: curvas ajustadas para ambas localidades permitiendo que no sean paralelas

Dado que el menor valor se alcanza en el segundo modelo, se escogería el modelo que incluye la interacción entre la variable densidad y la localidad.

1.5.4. Modelo aditivo generalizado y multidimensional con `smacker`

Se van a analizar los datos `smacker` del paquete `sm`. El objetivo es ver cómo influyen las condiciones del mar (temperatura de agua, etc.) en la ausencia o presencia de huevos de jurel en el mar cantábrico. Además, se utilizará la posición geográfica (latitud y longitud) como covariables para captar el efecto espacial.

```
library("sm")
data(smacker)
library("dplyr")
smacker <- smacker |>
  mutate(Presence = ifelse(Density>0, 1, 0),
        smack.long = -smack.long,
        ldepth = log(smack.depth))
library("maps")
par(pty="s")
Position <- cbind(smacker$smack.long, smacker$smack.lat)
plot(Position,col=NULL,xlim=c(-10,-1),ylim=c(43,48),cex=1.2,xlab="longitud",
     ylab="latitud")
map("world",add=TRUE,fill=TRUE,col="grey")
points(Position[smacker$Presence==1],pch=1,cex=.5,col=4)
points(Position[smacker$Presence==0],pch=16,cex=.5,col=2)
legend("topleft",c("Presencia ", "Ausencia"), col=c(4,2),pch=c(1,16),cex=.85)
```

Dado que la variable respuesta es dicotómica, se va a utilizar un modelo de regresión logística en el que se va a flexibilizar la relación lineal de las variables explicativas con la respuesta,

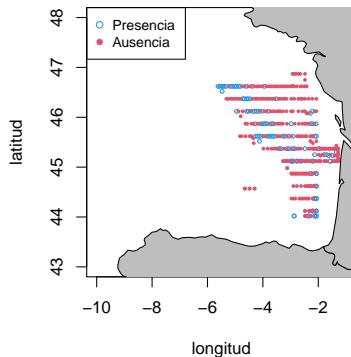


Figura 1.8: Localización donde se midió la ausencia/presecia de huevos de jurel

y además se usará una superficie para estimar el efecto de la localización como una función en dos dimensiones (en función de la posición geográfica: latitud y longitud). En este caso, en vez de usar el término `te()` se usa `s()` también para el caso de 2 dimensiones. La diferencia fundamental con `te()` es que `s()` asume un suavizado isotrópico, es decir, el mismo parámetro de suavizado para la latitud y longitud. No se debe usar `s()` para el suavizado en dos dimensiones si las covariables están medidas en unidades diferentes (en este caso como sí lo están, se puede usar el suavizado isotrópico):

```
logit1 <- gam(Presence~s(ldepth)+ s(Temperature)+ s(smack.long, smack.lat, k=60),
                family=binomial, select=TRUE, data=smacker)
b <- getViz(logit1)
print(plot(b, allTerms = T), pages = 1)
```

En la Fig. 1.9 se aprecia que la relación entre la temperatura y la probabilidad de presencia de huevos es no lineal, mientras que es lineal para la profundidad. El R^2 es 0,4 por lo que sería necesario recoger más variables para poder obtener buenas predicciones.

Si se quisieran obtener las probabilidades predichas se haría con la función `predict`.

```
prob=predict(logit1,type="response")
```

1.5. Casos prácticos

27

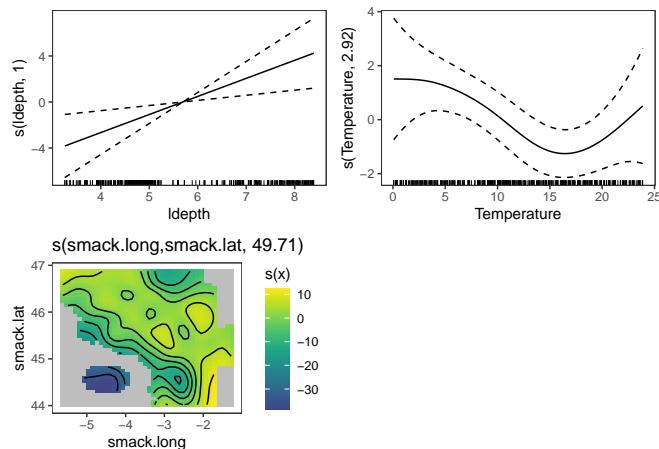


Figura 1.9: Efectos suaves estimados por el modelo para las variables. Efecto de la profundidad y temperatura en la fila superior y efecto espacial en la inferior.

Resumen

En este capítulo se introducen los modelos aditivos generalizados, en particular:

- Se presentan distintos aspectos metodológicos relacionados con la estimación e inferencia en modelos GAM.
- Se muestra el uso de R para el ajuste de este tipo de modelos.
- Se presentan diversos casos prácticos que ilustran la versatilidad de estos modelos para analizar datos complejos.

Capítulo 2

Modelos mixtos

María Durbán^a y Víctor Casero-Alonso^b

^aUniversidad Carlos III de Madrid ^bUniversidad de Castilla-La Mancha

2.1. Conceptos básicos

Los **modelos mixtos** (MMs) para variables de respuesta continuas son modelos estadísticos en los que los residuos siguen una distribución Normal pero puede que no sean independientes o no tengan varianza constante. Son necesarios en muchas situaciones, sobre todo en experimentos donde se realiza algún tipo de muestreo:

1. Estudios con datos agrupados, como por ejemplo, alumnos en una clase, individuos en una ciudad.
2. Estudios longitudinales o de medidas repetidas, donde un elemento o individuo es medido repetidamente a lo largo del tiempo o bajo condiciones distintas.

Este tipo de estudios se pueden encontrar en diferentes áreas como la medicina, biología, ciencias experimentales y sociales.

2.1.1. Tipo y estructura de los datos

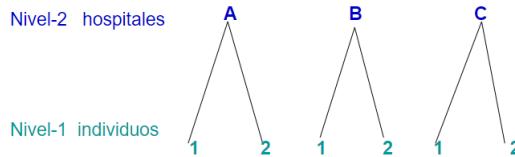
La estructura de los datos con la que se trabaja es el factor determinante para saber si se han de utilizar modelos mixtos, y en su caso, qué tipo de modelo.

2.1.1.1. Datos jerárquicos (o agrupados)

En este tipo de datos, la variable dependiente (de respuesta, de interés) se mide una sola en cada unidad de análisis (individuos, objetos, elementos ...), y los individuos¹ están agrupados (o anidados) en unidades mayores. Muchos tipos de datos tienen una estructura jerárquica: alumnos en escuelas, personas en municipios, pacientes en hospitales, plantas en una parcela...

Las jerarquías son una forma de representar la relación de dependencia que hay entre los individuos y los grupos a los que pertenecen. Por ejemplo, supóngase que se quiere hacer un estudio sobre el tiempo de recuperación en pacientes hospitalizados por COVID-19 en diferentes hospitales. Se tiene la siguiente estructura con dos niveles:

- Muchos individuos en el nivel 1 (pacientes).
- Agrupados en unas pocas unidades en el nivel 2 (hospitales).



Las estructuras multinivel pueden aparecer también como consecuencia del diseño del estudio que se está llevando a cabo. Por ejemplo, una encuesta sobre el estado de salud puede dar lugar a un diseño a tres niveles: primero se muestran regiones, luego distritos y después individuos.

En cada nivel de la jerarquía se pueden medir variables. Algunas estarán medidas en su nivel *natural*; por ejemplo, en el nivel del hospital se podría medir el tamaño, y al nivel de los pacientes se podría medir su situación socio-económica. Además, se pueden mover las variables de un nivel a otro mediante agregación o desagregación:

- **Agregación:** la variable al nivel más bajo se mueve a un nivel más alto; por ejemplo, se puede asociar a cada hospital la media del nivel socioeconómico de sus pacientes.
- **Desagregación:** mover las variables a un nivel más bajo; por ejemplo, asignarle a cada paciente una variable que indique el tamaño de su hospital de referencia.

2.1.1.2. Medidas repetidas y datos longitudinales

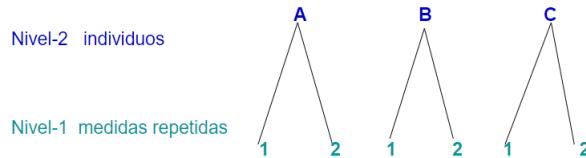
En este tipo de datos la variable dependiente se mide más de una vez en un mismo individuo ([Singer and Willet, 2003](#)). Por ejemplo, se miden los niveles de glucosa de un enfermo antes y después de haberle inyectado insulina. Este tipo de datos también puede ser considerado como datos multinivel (o jerárquicos) donde el Nivel 2 representa a los individuos y el Nivel 1 a las

¹En adelante nos referiremos a las unidades de análisis como individuos.

2.1. Conceptos básicos

31

diferentes medidas tomadas. Dado que las medidas se toman a un mismo individuo, es probable que no sean independientes, por lo que utilizar un modelo lineal ordinario no sería apropiado.



Por **datos longitudinales** se entienden datos en los que la variable dependiente se ha medido en distintos instantes de tiempo en cada una de las unidades de análisis. En algunos casos, cuando la variable dependiente se mide a lo largo del tiempo, puede ser difícil identificar si los datos son medidas repetidas o datos longitudinales. Desde el punto de vista del análisis de los datos mediante MMs esta distinción no es un elemento crítico. Lo importante es que en ambos tipos de datos la variable dependiente se ha medido repetidas veces en la misma unidad de análisis, y que, por tanto, las observaciones no serían independientes.

2.1.2. ¿Efectos fijos o aleatorios?

En un modelo mixto la clave se encuentra en la distinción entre efectos fijos y aleatorios ([Snijers, 2003](#)). Esto es importante porque la inferencia y el análisis de ambos efectos es distinta.

Los **efectos fijos** son variables en las cuales el investigador ha incluido sólo los niveles (o tratamientos) que son de su interés. Por ejemplo, en un experimento se puede estar interesados en comparar dos grupos, uno al que se le aplica un tratamiento y otro de control. En este caso, el estudio compara los grupos y no interesa generalizar los resultados a otros tratamientos que podrían haber sido incluidos. Otro ejemplo sería el caso en el que se hace una encuesta y se eligen 10 ciudades. Si sólo interesan los resultados para esas 10 ciudades y no se quieren generalizar al resto de ciudades que podrían haber sido seleccionadas, la variable ‘ciudad’ será un efecto fijo. Si se eligen las ciudades de forma aleatoria de una población grande de ciudades se consideraría la variable ‘ciudad’ como un **efecto aleatorio**.

Una cantidad se considera aleatoria cuando cambia sobre las unidades de una población. Cuando un efecto en un modelo estadístico es considerado aleatorio, se está asumiendo que se quieren extraer conclusiones sobre la población de la cual se han elegido las unidades observadas, y no se tiene interés en esas unidades en particular. En este contexto se habla de **intercambiabilidad**, en el sentido de que se podría cambiar una unidad de la muestra por otra de la población y sería indiferente. Este es el caso de los factores de agrupamiento o diseño, como las parcelas en un experimento agrícola, o los días cuando un experimento se lleva a cabo en días distintos, o el técnico de laboratorio cuando hay varios haciendo el experimento; también lo serían los sujetos en un diseño de medidas repetidas o las localizaciones donde se recogen muestras en un río, si el objetivo es generalizar a todo el río.

Los métodos estándar utilizados para construir tests e intervalos de confianza para los efectos fijos no son válidos para los efectos aleatorios, pues en este último caso los efectos observados son sólo una muestra de todos los posibles efectos.

La clave para distinguir, estadísticamente hablando, entre efectos fijos y aleatorios, es si los niveles de la variable se pueden interpretar como extraídos de una población con una cierta distribución de probabilidad. En el caso de un efecto fijo, normalmente interesaría comparar los resultados de la variable dependiente para los distintos niveles de la variable explicativa, es decir, interesaría la diferencia entre las medias. En el caso de efectos aleatorios, no interesa específicamente comparar si las medias son distintas, sino cómo el efecto aleatorio explica la variabilidad en la variable dependiente. Por lo tanto, para que un efecto pueda considerarse aleatorio, es necesario que la variable dependiente presente cierta variabilidad no explicada asociada con las unidades del efecto aleatorio.

La Fig. 2.1 puede ayudar a determinar si un efecto es fijo o aleatorio:

1. ¿Cuál es el número de niveles?

Pequeño	Fijo
Grande o infinito	Posiblemente aleatorio

2. ¿Son los niveles repetibles?

Sí	Fijo
No	Aleatorio

3. ¿Hay, conceptualmente, un número infinito de niveles?

No	Posiblemente fijo
Sí	Posiblemente aleatorio

4. ¿Se necesitan realizar inferencias para niveles no incluidos en el muestreo?

No	Posiblemente fijo
Sí	Posiblemente aleatorio

Figura 2.1: Cuestiones para determinar si un efecto es fijo o aleatorio

Por ejemplo, en un estudio sobre satisfacción en el trabajo (variable dependiente) de los empleados (unidades observadas) de un cierto número de empresas (efecto aleatorio), si el nivel de satisfacción de los empleados de unas empresas es mayor que el de otras y el investigador no lo tiene en cuenta, habrá una cierta variabilidad residual asociada con el efecto ‘empresa’. Si esta variabilidad fuera próxima a cero, no sería necesario incluir el efecto aleatorio asociado con la empresa.

¿Por qué hay que utilizar modelos mixtos?

Cuando las observaciones están agrupadas en niveles o siguen una cierta jerarquía, las unidades se ven afectadas por el grupo al que pertenecen. Las jerarquías (o niveles) permiten representar la relación de dependencia entre los individuos y los grupos a los que pertenecen. Los alumnos que están en una misma escuela se parecen más entre sí que si se hubieran seleccionado aleatoriamente de entre toda la población de alumnos. Los modelos mixtos permiten tener en cuenta que las observaciones no son independientes.

2.2. Formulación del modelo con efectos aleatorios o modelos mixtos

El nombre de *modelos mixtos lineales* viene del hecho de que estos modelos son lineales en los parámetros, y en las covariables, y pueden implicar efectos fijos o aleatorios. Son, por lo tanto, una extensión de los modelos lineales de regresión.

2.2.1. Formulación general

La formulación general de un modelo mixto tiene la siguiente forma:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{u} + \boldsymbol{\epsilon}, \quad \mathbf{u} \sim N(0, \mathbf{G}), \quad \boldsymbol{\epsilon} \sim N(0, \mathbf{R}), \quad (2.1)$$

donde:

- \mathbf{X} es una matriz $n \times k$ (k es el número de efectos fijos).
- \mathbf{Z} es una matriz $n \times p$ (p es el número de efectos aleatorios).
- $\boldsymbol{\beta}$ es el vector de efectos fijos y \mathbf{u} el de efectos aleatorios.
- \mathbf{G} es la matriz de varianzas-covarianzas de los efectos aleatorios, con dimensión $p \times p$.
- \mathbf{R} es la matriz de varianzas-covarianzas del error.

2.2.1.1. Estimación de $\boldsymbol{\beta}$ y \mathbf{u}

Se hace mediante las llamadas **ecuaciones de Henderson** (Henderson, 1953). Permiten obtener el mejor estimador lineal insesgado de $\boldsymbol{\beta}$ y el mejor predictor lineal insesgado de \mathbf{u} . Se obtienen maximizando la densidad conjunta de \mathbf{y} y \mathbf{u} :

$$f(\mathbf{y}, \mathbf{u}) = f(\mathbf{y}|\mathbf{u})f(\mathbf{u}), \quad \mathbf{y}|\mathbf{u} \sim N(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{u}, \mathbf{R}) \quad \mathbf{u} \sim N(0, \mathbf{G}). \quad (2.2)$$

Derivando con respecto a $\boldsymbol{\beta}$ y \mathbf{u} se obtienen las ecuaciones de Henderson cuya solución es:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{V}^{-1}\mathbf{X})^{-1}\mathbf{X}'\mathbf{V}^{-1}\mathbf{y} \quad (2.3)$$

$$\hat{\mathbf{u}} = \mathbf{G}\mathbf{Z}'\mathbf{V}^{-1}(\mathbf{y} - \mathbf{X}\hat{\boldsymbol{\beta}}), \quad (2.4)$$

donde $\mathbf{V} = \mathbf{Z}\mathbf{G}\mathbf{Z}' + \mathbf{R}$. Sin embargo, \mathbf{V} depende de los parámetros de la varianza en el modelo, que forman parte de \mathbf{G} y \mathbf{R} y que es necesario estimar, como se muestra a continuación.

2.2.1.2. Estimación de los componentes de la varianza

Los métodos más comunes para la estimación de los parámetros de las matrices de covarianzas son: Máxima verosimilitud (ML) y Máxima verosimilitud restringida (REML). No existe una solución cerrada para los estimadores, y se hace de forma numérica o mediante algoritmos iterativos. REML tiene en cuenta los grados de libertad utilizados para estimar los efectos fijos en el modelo. Si n es pequeño, REML dará mejores estimaciones que ML; si N es grande, no habrá prácticamente ninguna diferencia. El método preferido es REML.

2.2.2. Inferencia y selección del modelo

2.2.2.1. Contrastes de hipótesis para los efectos fijos, β

Utilizando la distribución aproximada:

$$\hat{\beta} \sim N \left(\beta, \underbrace{(\mathbf{X}' \hat{\mathbf{V}}^{-1} \mathbf{X})^{-1}}_{Var(\hat{\beta})} \right)$$

- Si se contrastan parámetros individuales se acudirá al t-test para un solo efecto.
- Si se contrasta un conjunto de parámetros se acudirá al F-test para más de un efecto.
- También se pueden comparar modelos usando el test de la razón de verosimilitud, LRT por sus siglas en inglés:

$$LRT = -2 [\ln(l_{H_0}) - \ln(l_{H_1})] \approx \chi^2_{df}. \quad (2.5)$$

Nota

En este caso hay que utilizar ML para estimar los parámetros de la varianza.

2.2.2.2. Contrastes de hipótesis para los parámetros de varianza

Al usar el test LRT en Eq. (2.5) se ha de tener en cuenta que la distribución asintótica del estadístico del test depende de si el valor del parámetro bajo la hipótesis nula (H_0) está en la frontera del espacio paramétrico²

- **Caso 1:** El valor de los parámetros de varianza bajo H_0 no está en la frontera del espacio paramétrico (por ejemplo, al contrastar si los parámetros de varianza de dos efectos aleatorios son iguales o no). En ese caso se utiliza el test normalmente.
- **Caso 2:** El valor de los parámetros de varianza bajo H_0 está en la frontera del espacio paramétrico (por ejemplo, si se quiere contrastar si la varianza del efecto aleatorio es cero o no). La distribución asintótica del estadístico del test es una mixtura entre χ^2_p y χ^2_{p-1} , concretamente $0,5\chi^2_p + 0,5\chi^2_{p-1}$, donde p es el número de parámetros de la varianza que se hacen cero bajo la H_0 .

²Es espacio paramétrico es el conjunto de posibles valores del parámetro. Los valores que están en la frontera son los valores que están en el límite inferior (el mínimo) o el superior (máximo) del conjunto de valores posible. Dado que la varianza es positiva, si se contrasta si el valor es cero, estaría tomando un valor en la frontera.

2.2.3. Diagnosis del modelo

En el caso de modelos mixtos se ha de verificar la hipótesis de normalidad tanto para los residuos al nivel más bajo como para los efectos aleatorios, y también las de independencia.

En este tipo de modelos, se utilizan los residuos condicionales, que son la diferencia entre los valores observados y el valor predicho condicional:

$$\hat{\epsilon} = y - X\hat{\beta} - Z\hat{u}.$$

Estos residuos tienden a estar correlados y sus varianzas pueden cambiar de un grupo a otro, aunque en el verdadero modelo los residuos sean incorrelados y con varianza constante. Para solucionar este problema se pueden escalar los residuos por sus desviaciones estándar (o las estimaciones de éstas), dando lugar a los *residuos estandarizados* (si las desviaciones estándar son conocidas), o a los *residuos studentizados* (si son desconocidas y se utilizan estimaciones de las mismas). Con estos residuos se hace un análisis similar al caso de modelos de regresión lineal.

Además se tendrá que comprobar la hipótesis de normalidad de los efectos aleatorios.

2.3. Procedimiento con **R** para ajustar modelos mixtos

Hay varios paquetes de **R** para el ajuste de modelos mixtos. Los más usados son **nlme** y **lme4**. El segundo es una versión del primero que incluye modelos más generales y mejora los gráficos. A continuación se describe la función principal del paquete **lme4**.

2.3.1. La función **lmer()**

Esta función permite el uso de efectos aleatorios anidados y de errores correlados y/o heterocedásticos dentro de los grupos. En general, para definir un modelo mixto se necesita especificar la estructura de la media y de la parte aleatoria del modelo, incluidos los factores de agrupamiento, así como la estructura de correlación (si la hay).

También se puede especificar el método de estimación: “REML” o “ML”.

La parte aleatoria del modelo se incluye entre paréntesis en la ecuación y “|” separa las variables de agrupamiento de las predictoras, si no hay variables predictoras para la parte aleatoria se pondría un 1.

La función **VarCorr()** aplicada a un objeto **lmer** proporciona información sobre la estructura de componentes de varianza.

2.4. Caso práctico

En esta sección se comenzará viendo cómo construir diferentes modelos con efectos aleatorios según a qué nivel estén medidas las variables explicativas y se terminará dando una guía de

construcción de estos modelos en la práctica. Los datos con los que se va a trabajar se encuentran en el dataframe `Hsb82` del paquete `mlmRev` y provienen de un estudio titulado *High School and Beyond*. Los datos corresponden a 7185 estudiantes repartidos en 160 escuelas, el número de alumnos por escuela varía entre 14 y 67. La variable de interés, `mAch`, es el nivel estandarizado alcanzado en matemáticas. Una cuestión inicial que se puede plantear es si el nivel socioeconómico (`cse`) del alumno predice las diferencias en el nivel de matemáticas. Para ello se ajustaría el modelo:

$$y_j = \beta_0 + \beta_1 x_j + \epsilon_j,$$

que ignora que los alumnos provienen de distintos centros (por eso solo aparece el subíndice j , que es el que representa a las unidades de nivel más bajo, en este caso a los alumnos).

```
library("mlmRev")
Hsb82$school = factor(Hsb82$school, ordered=F)
multi0 <- lm(mAch ~ cses, data = Hsb82)
summary(multi0)

#>
#> Call:
#> lm(formula = mAch ~ cses, data = Hsb82)
#>
#> Residuals:
#>      Min       1Q   Median       3Q      Max
#> -17.8660  -5.1165   0.2966   5.3880  14.8705
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 12.74785   0.07933 160.69 <2e-16 ***
#> cses         2.19117   0.12010   18.24 <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 6.725 on 7183 degrees of freedom
#> Multiple R-squared:  0.04429,    Adjusted R-squared:  0.04415
#> F-statistic: 332.8 on 1 and 7183 DF,  p-value: < 2.2e-16
```

La ordenada en el origen es 12.75 y la pendiente 2.19, lo que indica que por cada unidad que aumenta el nivel socio-económico, la puntuación del test aumenta en 2.19 unidades, además se puede ver que el coeficiente es significativo.

Supóngase que ocurre la situación mostrada en la Fig. 2.2:

Los alumnos de la escuela A (rombos negros) sacan, en promedio, mejores notas que las que le asignaría el modelo ajustado; con la escuela B (círculos azules) ocurre lo contrario. El gráfico indica que la ordenada en el origen no debería ser la misma para todos los centros, sino que debería ser distinta para distintos centros. Es decir, el valor predicho debe ajustarse hacia arriba o abajo. Eso se puede conseguir permitiendo que cada escuela tenga su propia ordenada en el origen:

$$y_{ij} = \beta_{0i} + \beta_1 x_{ij} + \epsilon_{ij}$$

2.4. Caso práctico

37

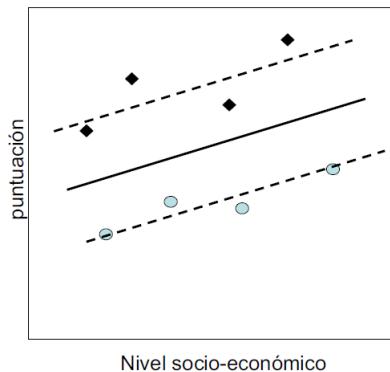


Figura 2.2: Ilustración de posibles escenarios para dos escuelas

Este modelo es similar al anterior añadiendo el subíndice i para identificar el centro al que pertenece cada alumno. En realidad, se utiliza una variable categórica con tantas categorías como escuelas.

```
multi1 <- lm(mAch ~ cses + school, data = Hsb82)
```

Se están considerando las escuelas como un efecto fijo y no aleatorio, es decir, implícitamente se está suponiendo que solo interesan estas escuelas en particular.

La situación se pueden complicar más: es posible que el efecto del nivel socio-económico sea distinto para cada centro, es decir, que un aumento de una unidad en ese nivel pueda dar lugar a un aumento distinto en la nota del test en cada centro. En la Fig. 2.3 se ve como la pendiente de la recta para la escuela C es distinta a la dos anteriores.

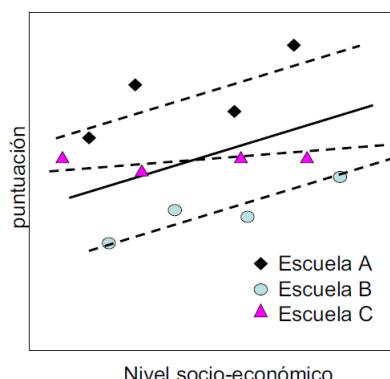


Figura 2.3: Ilustración de posibles escenarios para tres escuelas

El modelo que permite tener en cuenta esta situación es:

$$y_{ij} = \beta_{0i} + \beta_{1i}x_{ij} + \epsilon_{ij}$$

donde aparece ahora el sub-índice i también en la pendiente, lo que indica que cada centro tiene una pendiente diferente.

```
multi2 <- lm(mAch ~ cses*school, data = Hsb82)
```

El código anterior generaría 159 coeficientes más (uno por cada escuela), que son los que se incluirían con la interacción. Pero no interesan estas escuelas en concreto, sino la población de la que estas escuelas son una muestra.

Con un modelo con efectos aleatorios, sin embargo, se pueden contestar a preguntas como: ¿Cuáles son las causas de esta variabilidad? ¿Qué variables pueden explicarla?

2.4.1. Modelo con ordenada en el origen aleatoria

Es el modelo mixto más sencillo. Se considera que los datos tienen una estructura con dos niveles, los alumnos están en el nivel 1 y están agrupados en escuelas, nivel 2. Se va a empezar suponiendo que no se dispone de ninguna variable explicativa, y que por lo tanto el único interés es la diferencia entre las notas medias del test de matemáticas entre los distintos centros.

Los dos niveles del modelo son:

$$\text{Nivel 1: } y_{ij} = \beta_{0i} + \epsilon_{ij}$$

- El subíndice j corresponde a individuos y el i a escuelas, si se considera a las escuelas como un efecto aleatorio,
- β_{0i} (la media de cada escuela) vendría dada por:

$$\text{Nivel 2: } \beta_{0i} = \beta_0 + u_i,$$

- β_0 es la media de todos los alumnos,
- u_i es la desviación de la media de la escuela i respecto de la media de todas las escuelas.

Poniendo las dos ecuaciones juntas:

$$y_{ij} = \beta_0 + u_i + \epsilon_{ij}, \quad i = 1, \dots, m, \quad j = 1, \dots, n_m \quad (2.6)$$

- La media de y para el grupo i es $\beta_0 + u_i$,
- Los residuos a nivel individual ϵ_{ij} son la diferencia entre el valor de la variable respuesta del individuo j y la media del grupo al que pertenece,
- $u_i \sim N(0, \sigma_u^2)$, $\epsilon_{ij} \sim N(0, \sigma_e^2)$, y ambos son independientes, es decir, las observaciones que provienen de distintas escuelas son independientes.

En el ejemplo de las escuelas:

2.4. Caso práctico

39

```
library("lme4")
Modelo0 <- lmer(mAch ~ 1+(1 | school), data = Hsb82)
Modelo0
#> Linear mixed model fit by REML ['lmerMod']
#> Formula: mAch ~ 1 + (1 | school)
#> Data: Hsb82
#> REML criterion at convergence: 47116.79
#> Random effects:
#> Groups   Name        Std.Dev.
#> school   (Intercept) 2.935
#> Residual            6.257
#> Number of obs: 7185, groups: school, 160
#> Fixed Effects:
#> (Intercept)
#>          12.64
```

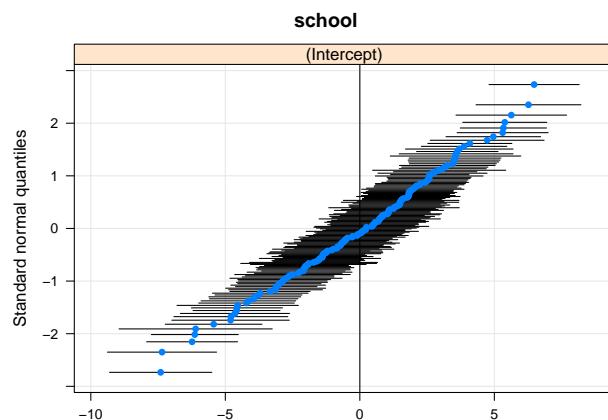
- La media total estimada es 12.64,
- La media estimada para la escuela i es: $12.64 + \hat{u}_i$, donde \hat{u}_i es el efecto aleatorio estimado de dicha escuela.

Para obtener los valores predichos de los efectos aleatorios se utiliza la función `ranef()`.

El siguiente gráfico permite ver los efectos aleatorios junto con sus intervalos de confianza (las escuelas han sido ordenadas atendiendo a su media para apreciar mejor la variabilidad entre las mismas).

Se dibujan los efectos aleatorios para ver si siguen una distribución Normal; para eso se ha de ajustar el modelo con la función `lmer()`:

```
library("lattice")
qqmath(ranef(Modelo0, condVar = TRUE))$school
```



Una primera aproximación para contrastar si hay o no diferencias entre los grupos sería calcular el intervalo de confianza para σ_u :

```
confint(Modelo0)
#>              2.5%    97.5%
#> .sig01      2.594729  3.315880
#> .sigma       6.154803  6.361786
#> (Intercept) 12.156289 13.117121
```

pudiéndose apreciar que el intervalo no contiene al cero. Sin embargo, la forma más correcta de hacerlo sería utilizando el test LRT en Eq. (2.5) para:

$$\begin{aligned} H_0 : \quad \sigma_u^2 = 0 &\Rightarrow y_{ij} = \beta_0 + \epsilon_{ij} \\ H_1 : \quad \sigma_u^2 \neq 0 &\Rightarrow y_{ij} = \beta_0 + u_i + \epsilon_{ij}. \end{aligned}$$

El resultado del test en este caso se compara con el valor de una mixtura de distribuciones Chi-cuadrado $0,5\chi_0^2 + 0,5\chi_1^2$.

```
Modelo_NULL <- lm(mAch ~ 1, data = Hsb82)
test = -2*logLik(Modelo_NULL, REML = T) + 2*logLik(Modelo0, REML = T)
mean(pchisq(test, df = c(0, 1), lower.tail = F))
#> [1] 9.320673e-217
```

Conclusión: el efecto aleatorio es necesario en el modelo.

El siguiente paso sería introducir las variables explicativas (en este caso solo hay una), ya estén al nivel 1 o al 2.

2.4.1.1. Variables explicativas en el Nivel 1 (individuos)

Como la variable explicativa está medida al Nivel 1, se introduce en la ecuación del Nivel 1:

- Nivel 1: $y_{ij} = \beta_0 + \beta_1 x_{ij} + \epsilon_{ij}$
- Nivel 2: $\mu_i = \beta_0 + u_i$

Si x es una variable continua, este modelo asume que la pendiente de la recta es la misma para todas las escuelas (por eso β_1 no lleva el subíndice i). Poniendo las dos ecuaciones juntas:

$$y_{ij} = \underbrace{\beta_0 + \beta_1 x_{ij}}_{\text{efectos fijos}} + \underbrace{u_i + \epsilon_{ij}}_{\text{efectos aleatorios}}$$

En este modelo, la relación global entre Y y X viene representada por la línea recta con ordenada en el origen β_0 y pendiente β_1 . Sin embargo, la ordenada en el origen para una determinada escuela i viene dada por $\beta_0 + u_i$. Será mayor o menor que la ordenada en el origen global β_0 .

2.4. Caso práctico

41

en una cantidad u_i . Aunque la ordenada en el origen varía de grupo a grupo, la pendiente es la misma para todos los grupos. Todas las líneas rectas ajustadas para cada grupo son paralelas.

En el ejemplo de las escuelas, se introduce como variable explicativa **cses** (nivel socioeconómico centrado en su media):

```
Modelo1 <- lmer(mAch ~ cses + (1 | school), data = Hsb82)
Modelo1
#> Linear mixed model fit by REML ['lmerMod']
#> Formula: mAch ~ cses + (1 | school)
#>   Data: Hsb82
#> REML criterion at convergence: 46724
#> Random effects:
#> Groups   Name        Std.Dev.
#> school   (Intercept) 2.945
#> Residual           6.084
#> Number of obs: 7185, groups: school, 160
#> Fixed Effects:
#> (Intercept)      cses
#>          12.636     2.191
```

Ahora se tienen dos efectos fijos:

$$\hat{\beta}_0 = 12,64 \\ \hat{\beta}_1 = 2,19$$

$\hat{\beta}_0$ es la nota media para alumnos con nivel socioeconómico medio (la variable está centrada). La recta media vendría dada por:

$$E[y|cses] = 12,64 + 2,19 cses$$

Para comprobar si la variable **cses** es significativa se utiliza el test LRT (2.5). Primero se tienen que ajustar de nuevo los modelos que se quieren comparar usando máxima verosimilitud (en vez de REML). Si se utiliza la función **lmer()** para ajustar el modelo no es necesario reajustar con ML pues la función **anova** lo hará automáticamente, mientras que si se usa la función **lme()** sí será necesario hacerlo.

```
anova(Modelo0, Modelo1)
#> Data: Hsb82
#> Models:
#> Model0: mAch ~ 1 + (1 | school)
#> Model1: mAch ~ cses + (1 | school)
#>       npar  AIC  BIC logLik deviance Chisq Df Pr(>Chisq)
#> Model0    3 47122 47142 -23558     47116
#> Model1    4 46728 46756 -23360     46720 395.4  1  < 2.2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Por lo tanto, el nivel socioeconómico afecta a los resultados escolares. Comparado con el modelo sin la variable explicativa (Modelo0), la inclusión del nivel socio-económico (Modelo1) ha reducido la variabilidad a nivel del alumno en un 2.8% ($(6.257 - 6.084)/6.257 = 0.028$).

2.4.1.2. Variables explicativas en el Nivel 2 (grupos)

Si las variables explicativas son medidas al Nivel 2, entonces:

$$\begin{aligned} \text{Nivel 1: } y_{ij} &= \mu_i + \epsilon_{ij} \\ \text{Nivel 2: } \mu_i &= \beta_0 + \beta_2 s_i + u_i \end{aligned}$$

$$y_{ij} = \underbrace{\beta_0 + \beta_2 s_i}_{\text{efectos fijos}} + \underbrace{u_i + \epsilon_{ij}}_{\text{efectos aleatorios}}.$$

En nuestro ejemplo, la variable utilizada es `sector` (público o privado):

$$mAch = \beta_0 + \beta_2 \text{sector} + u_i + \epsilon_{ij}$$

Se ajusta el modelo usando la función `lmer()`:

```
Modelo2 <- lmer(mAch ~ sector + (1 | school), data = Hsb82)
Modelo2
#> Linear mixed model fit by REML ['lmerMod']
#> Formula: mAch ~ sector + (1 | school)
#>   Data: Hsb82
#> REML criterion at convergence: 47080.13
#> Random effects:
#> Groups   Name        Std.Dev.
#> school   (Intercept) 2.584
#> Residual           6.257
#> Number of obs: 7185, groups: school, 160
#> Fixed Effects:
#>   (Intercept) sectorCatholic
#>           11.393            2.805
```

$$E[y|sector] = 11,39 + 2,8 \text{sector}$$

o equivalentemente

$$\begin{aligned} E[y|sector = 0] &= 11,39 \\ E[y|sector = 1] &= 11,39 + 2,8 = 14,19 \end{aligned}$$

La nota de un alumno en una escuela privada se espera que sea 2.8 unidades mayor que la de un alumno en una escuela pública (se puede generalizar, pues se asume que las escuelas son un efecto aleatorio). La varianza del efecto aleatorio de nivel 2, σ_u^2 , ha descendido: $(2,935^2 - 2,584^2)/2,935^2 = 0,22$, es decir, se ha reducido en un 22% la variabilidad no explicada entre los centros al introducir la variable sector.

Para contrastar si la variable sector es significativa se usa de nuevo el test LRT:

2.4. Caso práctico

43

```
anova(Modelo0, Modelo2)
#> Data: Hsb82
#> Models:
#> Modelo0: mAch ~ 1 + (1 | school)
#> Modelo2: mAch ~ sector + (1 | school)
#>      npar   AIC   BIC logLik deviance Chisq Df Pr(>Chisq)
#> Modelo0     3 47122 47142 -23558     47116
#> Modelo2     4 47087 47115 -23540     47079 36.705  1  1.374e-09 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Por lo tanto, el hecho de que la escuela sea pública o privada influye en el resultado académico de los alumnos.

2.4.2. Modelo con pendiente aleatoria

En este tipo de modelos se supone que la relación entre la variable respuesta y las variables explicativas va a ser distinta para las distintas unidades de nivel 2, es decir, la relación puede cambiar de un centro a otro. Por ejemplo, el efecto del nivel socioeconómico en las notas puede ser distinto en distintos centros, de modo que se puede relajar el modelo anterior, en el que la pendiente era la misma para todos los grupos, permitiendo que la pendiente varíe aleatoriamente entre los grupos.

$$\begin{aligned} \text{Nivel 1: } & y_{ij} = \mu_i + \beta_{1i}x_{ij} + \epsilon_{ij} \\ \text{Nivel 2: } & \mu_i = \beta_0 + u_i \\ & \beta_{1i} = \beta_1 + v_i \end{aligned}$$

Poniendo las dos ecuaciones juntas:

$$y_{ij} = \underbrace{\beta_0 + \beta_1 x_{ij}}_{\text{efectos fijos}} + \underbrace{u_i + v_i x_{ij} + \epsilon_{ij}}_{\text{efectos aleatorios}}, \quad \begin{pmatrix} u_i \\ v_i \end{pmatrix} \sim N(0, \mathbf{G}_i) \quad \mathbf{G}_i = \begin{pmatrix} \sigma_u^2 & \\ \sigma_{uv} & \sigma_v^2 \end{pmatrix},$$

donde σ_{uv} es la covarianza entre las ordenadas en el origen de los grupos y las pendientes. Un valor positivo de la covarianza implica que los grupos con un valor del efecto de grupo u_i elevado tienden a tener valores elevados de v_i , o equivalentemente, centros con ordenada en el origen alta, tienen pendiente alta.

El modelo en R sería:

```
Modelo3 <- lmer( mAch ~ cses + (cses | school), data = Hsb82)
Modelo3
#> Linear mixed model fit by REML ['lmerMod']
#> Formula: mAch ~ cses + (cses | school)
#>   Data: Hsb82
#> REML criterion at convergence: 46714.23
#> Random effects:
#> Groups   Name        Std.Dev. Corr
```

```
#> school  (Intercept) 2.9464
#>           cses          0.8331   0.02
#> Residual           6.0581
#> Number of obs: 7185, groups: school, 160
#> Fixed Effects:
#> (Intercept)      cses
#>       12.636      2.193
```

El efecto del nivel socioeconómico en la escuela i se estima como $2,19 + \hat{u}_i$, y la varianza de las pendientes entre escuelas es $0,833^2 = 0,694$. Para la *escuela promedio* se predice un aumento de 2,19 en la puntuación cuando el nivel socioeconómico aumenta en una unidad.

Ahora se tienen las siguientes estimaciones de los parámetros de varianza:

$$\hat{\sigma}_u^2 = 8,68 \quad \hat{\sigma}_v^2 = 0,694 \quad \hat{\sigma}_{uv} = \rho\sigma_u\sigma_v = 0,051 \quad \hat{\sigma}^2 = 36,7$$

La varianza de la ordenada en el origen estimada, 8,68, se interpreta como la variabilidad entre las escuelas para un nivel socioeconómico medio.

El parámetro de covarianza estimado es $\sigma_{uv} = 0,051$, por lo que se puede plantear si es necesario o no.

Para comprobarlo, el contraste de hipótesis sería en este caso:

$$H_0 : \sigma_{uv} = 0 \quad \text{y} \quad H_1 : \sigma_{uv} \neq 0$$

```
Modelo3.1 <- lmer(ses ~ cses + (cses || school), data = Hsb82)
```

Cuando se quiere que haya un efecto aleatorio para la ordenada en el origen y para la pendiente pero que estén incorrelados, en la función solo hay que poner doble barra en vez de simple.

En este caso no es necesario utilizar la mixtura de distribuciones Chi-cuadrado para contrastar $H_0 : \sigma_{uv} = 0$, pues σ_{uv} puede tomar cualquier valor.

```
anova(Modelo3.1, Modelo3)
#> Data: Hsb82
#> Models:
#> Modelo3.1: ses ~ cses + ((1 / school) + (0 + cses / school))
#> Modelo3: mAch ~ cses + (cses / school)
#>           npar    AIC    BIC logLik deviance Chisq Df Pr(>Chisq)
#> Modelo3.1     5 -226164 -226129 113087   -226174
#> Modelo3       6   46723  46764 -23355   46711      0  1          1
```

Por lo tanto, se puede suponer que la covarianza es 0.

El siguiente paso sería contrastar si es necesario que las rectas tengan pendientes diferentes, es decir, $H_0: \sigma_v^2 = 0$, $H_1: \sigma_v^2 > 0$. En este caso sí se necesita la aproximación:

2.4. Caso práctico

45

```
test <- -2 * logLik(Modelo1, REML = T) +
      2 * logLik(Modelo3.1, REML = T)
mean(pchisq(test, df = c(0, 1), lower.tail = F))
#> [1] 0
```

Por lo tanto, la pendiente es diferente en las distintas escuelas.

Además, se puede usar algún criterio de información para comparar los modelos:

```
AIC(logLik(Modelo3))
#> [1] 46726.23
AIC(logLik(Modelo3.1))
#> [1] -226113.6
AIC(logLik(Modelo1))
#> [1] 46732
```

A veces la covariante medida a nivel 2 (a nivel de grupo, en este caso escuelas) puede explicar tanto la variabilidad de la ordenada en el origen como la pendiente:

$$\begin{aligned} \text{Nivel 1: } y_{ij} &= \mu_i + \beta_1 x_{ij} + \epsilon_{ij} \\ \text{Nivel 2: } \mu_i &= \beta_0 + \beta_2 s_i + u_i \\ &\quad \beta_1 = \beta_1 + \beta_3 s_i + v_i \end{aligned}$$

$$y_{ij} = \underbrace{\beta_0 + \beta_1 x_{ij} \beta_2 s_i + \beta_3 x_{ij}}_{\text{efectos fijos}} : s_i + \underbrace{u_i + v_i x_{ij} + \epsilon_{ij}}_{\text{efectos aleatorios}} .$$

Al introducir la variable medida al nivel 2, la parte fija se modifica (con respecto al Modelo 3), pero no la parte aleatoria:

- β_2 indica si los centros privados son diferentes de los públicos en cuanto a su nota media.
- β_3 indica si los centros privados difieren de los públicos en cuanto a la relación entre el nivel socio-económico y la puntuación.

```
Modelo4 <- lmer(mAch ~ cses*sector + (cses || school),
                 data = Hsb82)
Modelo4
#> Linear mixed model fit by REML ['lmerMod']
#> Formula: mAch ~ cses * sector + ((1 | school) + (0 + cses | school))
#>   Data: Hsb82
#> REML criterion at convergence: 46648.85
#> Random effects:
#> Groups   Name        Std.Dev.
#> school   (Intercept) 2.5971
#> school.1 cses        0.5182
#> Residual            6.0580
#> Number of obs: 7185, groups: school, 160
#> Fixed Effects:
```

```
#>      (Intercept)      cses    sectorCatholic
#>      11.393          2.784        2.805
#> cses:sectorCatholic
#>      -1.346
```

Los centros privados tienen una nota media más alta que los públicos (2.81), y tienen una pendiente más suave que la de los centros públicos (-1.35). En un colegio privado la mejora de la nota con respecto al nivel socio-económico es más lenta que un colegio público.

2.4.3. ¿Cómo construir el modelo en la práctica?

1. Se ajusta el modelo con todos los efectos fijos y aleatorios posibles.

```
Modelo5 <- lmer(mAch ~ cses * sector+(cses | school), data = Hsb82)
```

2. Se contrasta qué efectos aleatorios son significativos, sin mover los efectos fijos.

Primero se contrasta si la covarianza entre efectos fijos y aleatorios es cero o no:

```
#Se ajusta el modelo con covarianza = 0
Modelo5.1 <- lmer(ses ~ cses * sector+(cses||school),
                   data = Hsb82)
anova(Modelo5.1, Modelo5)
## Data: Hsb82
## Models:
## > Modelo5.1: ses ~ cses * sector + ((1 / school) + (0 + cses / school))
## > Modelo5: mAch ~ cses * sector + (cses / school)
##>      npar      AIC      BIC logLik deviance Chisq Df Pr(>Chisq)
##> Modelo5.1    7 -220069 -220021 110042   -220083
##> Modelo5     8  46650  46705 -23317    46634      0  1       1
```

Como lo es, no se tendría que contrastar nada más. Los efectos aleatorios son los que se han incluido en el Modelo5. Si no hubiera sido significativa, se continuaría contrastando si la pendiente aleatoria es significativa y si la ordenada en el origen lo es.

3. Una vez elegidos los efectos aleatorios que se mantienen en el modelo, se eligen los efectos fijos:

```
Modelo6 = update(Modelo5, . ~ .-cses:sector)
anova(Modelo6, Modelo5)
## Data: Hsb82
## Models:
## > Modelo6: mAch ~ cses + sector + (cses / school)
```

2.4. Caso práctico

47

```
#> Modelo5: mAch ~ cses * sector + (cses | school)
#>      npar   AIC   BIC logLik deviance Chisq Df Pr(>Chisq)
#> Modelo6    7 46678 46726 -23332     46664
#> Modelo5    8 46650 46705 -23317     46634 29.983  1  4.358e-08 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Como la interacción es significativa ya no se tendría que hacer ningún test más y este sería el modelo final.

Resumen

En este capítulo se introducen los modelos mixtos o modelos con efectos aleatorios, en particular:

- Se dan las claves para distinguir entre efectos fijos y aleatorios.
- Se presenta la formulación del modelo y se han dado las claves para llevar a cabo la estimación del mismo.
- Se explican las etapas del proceso a seguir para el ajuste de este tipo de modelos.
- Se muestra el uso de R para ajustar estos modelos.
- Se ilustra el análisis de modelos multinivel como caso particular de un modelo con efectos aleatorios.

Capítulo 3

Modelos *sparse* y métodos penalizados de regresión

María Durbán

Universidad Carlos III de Madrid

3.1. Introducción

El modelo de regresión lineal múltiple: $y = \beta_0 + \beta_1 X_1 + \dots + \beta_p X_p + \varepsilon$, visto en el Capítulo 16, a pesar de su simplicidad, tiene importantes ventajas como la **interpretabilidad** y su buen poder **predictivo** en muchas situaciones.

En este Capítulo se va a ver cómo se puede hacer el modelo aún más interpretable y mejor predictor, y para conseguirlo se reemplazará el método de mínimos cuadrados (utilizado hasta ahora para la estimación de los parámetros) por un método alternativo.

Por lo tanto, el objetivo de este Capítulo es aprender técnicas para mejorar:

- **Precisión de la predicción:** en particular cuando el número de variables es mayor que el número de observaciones: $p > n$ (algo que ocurre con mucha frecuencia hoy en día). En este caso no se pueden utilizar mínimos cuadrados ya que la matriz de diseño no es de rango completo, y por lo tanto, no se puede encontrar una solución al problema de minimización. Por ello, se necesita reducir el número de variables, que además, evitará que se sobreajusten los datos.
- **Interpretabilidad del modelo:** al eliminar las variables irrelevantes (es decir, haciendo cero los correspondientes coeficientes) se obtendrá un modelo que es más fácil de interpretar. Por lo tanto, se presentarán varios métodos para llevar a cabo de forma automática la *selección de variables*.

Los métodos para reducir el número de variables en el modelo son:

- **Selección del mejor subconjunto:** su objetivo es identificar un subconjunto de entre los p predictores que se considera que son los que están relacionados con la variable respuesta.
- **Shrinkage:** en este caso no se quieren seleccionar variables explícitamente, sino que se añade una penalización que penaliza el número de coeficientes o su tamaño.
- **Reducción de la dimensión:** el objetivo es proyectar los p -predictores en un subespacio de dimensión más pequeña (mediante el uso de combinaciones lineales de variables, las cuales se usarán como predictores). Dichas combinaciones lineales se llaman **Componentes principales** y a su análisis se dedica el Cap. ??.

Por lo tanto, en este Capítulo se ven los dos primeros métodos.

3.2. Selección del mejor subconjunto

Supóngase que se tiene acceso a p variables predictoras, pero se quiere un modelo más simple que involucre solo a un subconjunto de esos p predictores. La forma lógica de conseguirlo es considerar todos los posibles subconjuntos de los p predictores y elegir el mejor modelo de entre todos los modelos construidos con cada uno de los subconjuntos de variables. Los pasos a seguir serían:

1. Se crea el modelo **nulo**, M_0 , que es aquel que solo contiene la ordenada en el origen y ningún predictor. Este modelo simplemente predice la media muestral para cada observación.
2. Para cada valor de $k = 1, 2, \dots, p$ se calculan los $\binom{p}{k}$ modelos que contienen k predictores. Es decir, los p modelos que contienen 1 predictor, los $p \times (p - 1)/2$ modelos que contienen 2 predictores, etc.
3. Para cada valor de k , se elige el mejor entre los $\binom{p}{k} = \frac{p!}{(p-k)!k!}$ posibles modelos y se denota por M_k . Es decir, M_1 sería el mejor modelo entre los p modelos con una sola variable, M_2 sería el mejor modelo entre los modelos con dos variables, etc. En este caso, el **mejor** modelo sería aquel cuyo RSS (suma de residuos al cuadrado) sea menor, o equivalentemente, aquel cuyo R^2 sea mayor.
4. Elegir entre los modelos: M_1, \dots, M_p aquel que es mejor utilizando un criterio como AIC, BIC o R^2 ajustado.

Este método se puede usar también en el caso de GLMs, es decir se usará el *deviance* en vez de RSS .

3.2.1. Procedimiento con R: la función `regsubset()`

Se va a aplicar el método descrito al conjunto de datos `Hitters` del paquete `ISLR2`. El objetivo es predecir el sueldo, `Salary`, de jugadores de béisbol a partir de varias variables asociadas con su rendimiento el año anterior.

La variable `Salary` no está disponible para algunos de los jugadores que se pueden identificar utilizando la función `is.na()`. Y la función `sum()` permite ver cuántos hay. Se utilizará `na.omit()` para eliminarlas.

```
library("ISLR2")
Hitters <- na.omit(Hitters)
```

La función `regsubsets()` del paquete `leaps` lleva a cabo la selección del mejor subconjunto de variables identificando el mejor modelo que contiene un número dado de variables (1,2,3, etc.) atendiendo a *RSS*. La sintaxis usada es similar a la de la función `lm()`.

```
library("leaps")
regfit_full <- regsubsets(Salary ~ ., Hitters)
```

Los resultados se pueden ver usando `summary()`, donde se muestra el mejor modelo para cada subconjunto de variables. Con un asterisco indica las variables incluidas en cada modelo. Por ejemplo, el mejor modelo con dos variables incluye `Hits` y `CRBI`. Por defecto, `regsubsets()` solo muestra los resultados de los modelos que contienen hasta ocho variables. La opción `nvmax` se puede usar para incrementar esta cantidad, por ejemplo hasta 19 variables (que es el número de variables predictoras en el conjunto de datos):

```
regfit_full <- regsubsets(Salary ~ .,
  data = Hitters,
  nvmax = 19
)
reg_summary <- summary(regfit_full)
```

La función `summary()` devuelve diferentes medias de bondad de ajuste: R^2 , *RSS*, R^2 ajustado, C_p y *BIC*. Se utiliza esta información para elegir el *mejor* de entre todos los modelos.

```
names(reg_summary)
#> [1] "which"    "rsq"      "rss"      "adjr2"    "cp"       "bic"      "outmat"   "obj"
reg_summary$adjr2
#> [1] 0.3188503 0.4208024 0.4450753 0.4672734 0.4808971 0.4972001 0.5007849
#> [8] 0.5137083 0.5180572 0.5222606 0.5225706 0.5217245 0.5206736 0.5195431
#> [15] 0.5178661 0.5162219 0.5144464 0.5126097 0.5106270
```

Por ejemplo el R^2 ajustado mayor corresponde al modelo con 11 variables. Se pueden también visualizar los resultados y dibujar simultáneamente, por ejemplo, los valores de *RSS* y R^2 ajustado de todos los modelos.

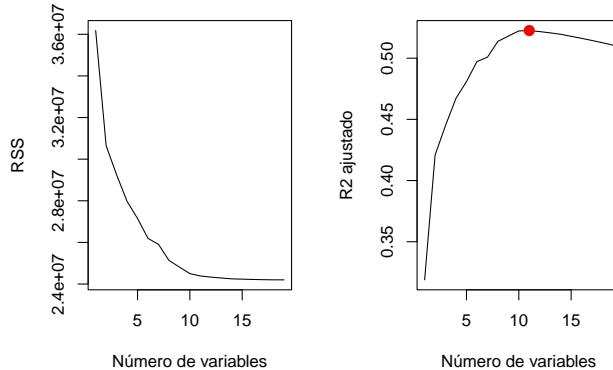


Figura 3.1: valores de R^2 y R^2 ajustados correspondientes a modelos con distinto número de variables

Otra manera de visualizar los resultados es:

```
plot(regfit_full, scale = "adjr2")
```

La primera fila tiene un cuadrado negro en cada una de las variables elegidas de acuerdo al modelo con mayor R^2 ajustado (en este caso, sería similar para los otros criterios).

Varios modelos tienen un valor de R^2 ajustado próximo a 0,52, pero es el modelo con 11 variables el que alcanza el mayor valor. La función `coef()` permite ver los coeficientes estimados de este modelo.

```
coef(regfit_full, 11)
#> (Intercept)      AtBat       Hits       Walks      CATBat      CRUNS
#> 135.7512195 -2.1277482  6.9236994  5.6202755 -0.1389914  1.4553310
#> CRBI          CWalks     LeagueN DivisionW PutOuts      Assists
#>  0.7852528 -0.8228559 43.1116152 -111.1460252   0.2894087  0.2688277
```

3.2.2. Selección *stepwise*

Cuando el número de variables predictoras, p , es grande, el método anterior es computacionalmente muy costoso ya que el número de posibles combinaciones de variables crece de una manera alarmante. En general, la función `regsubset()` puede lidar con hasta 30-40 variables predictoras. Además, otro problema es el sobreajuste. Si se tienen 40 variables, se estarían ajustando millones de modelos, y puede que el modelo elegido funcione muy bien en los datos utilizados para su construcción, pero no tan bien en un nuevo conjunto de datos. Una alternativa es el método **stepwise**. La idea detrás de este método es similar a la anterior, pero solo se mira un conjunto mucho más pequeño de modelos.

3.2. Selección del mejor subconjunto

53

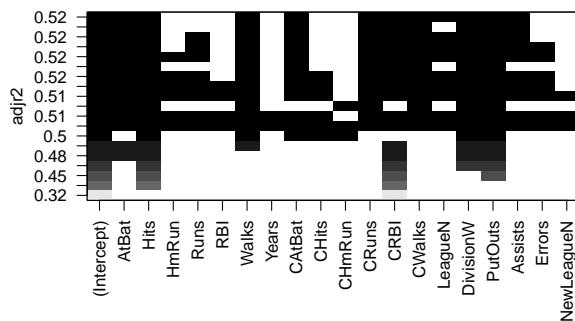


Figura 3.2: Variables seleccionadas en cada uno de los modelos y su correspondiente valor de R^2 ajustado.

Hay dos posibilidades de hacer stepwise: **forward** y **backward**. Ambas son bastante similares; la principal diferencia es el modelo del que se parte: del modelo sin ninguna variable predictora o del modelo con todas ellas.

3.2.2.1. Forward stepwise

En este caso se comienza con el modelo *nulo*, M_0 , y se van añadiendo variables secuencialmente. En particular, en cada paso (*step*) la variable que proporciona la mayor mejora al ajuste es la que se añade al modelo. Los pasos a seguir serían:

1. Se crea el modelo **nulo**, M_0 .
2. Para cada valor de $k = 0, 1, 2, \dots, p$:
 - i. Se consideran todos los $p - k$ modelos que surgen de aumentar el modelo M_k con un predictor.
 - ii. Se elige el **mejor** de esos $p - k$ modelos, que se denotará M_{k+1} . El término **mejor** significa tener el *RSS* más bajo o el R^2 más alto.
3. Se elige entre los modelos M_0, \dots, M_p aquel que es mejor utilizando un criterio como AIC, BIC o R^2 ajustado.

Este enfoque tiene ventajas computacionales claras, ya que el número de modelos ajustados es mucho menor, pero no garantiza que el modelo elegido sea el mejor modelo posible, especialmente si existe correlación entre las variables predictoras.

3.2.2.2. Backward stepwise

En este caso se comienza con el modelo que incluye todas (p) las variables predictoras y se van eliminando de forma iterativa hasta llegar al modelo nulo (M_0). Los pasos serían:

1. Se ajusta el modelo M_p , que contiene todas (p) las variable predictoras.
2. Para cada valor de $k = p, p-1, \dots, 1$:
 - I. Se consideran todos los k modelos que surgen de reducir en el modelo M_k un predictor, es decir, modelos con $k-1$ variables predictoras.
 - II. Se elige el **mejor** de esos k modelos, que se denotará M_{k-1} . El término **mejor** significa tener el RSS más bajo o el R^2 más alto
3. Se elige entre los modelos M_0, \dots, M_p aquel que es mejor utilizando una criterio como AIC, BIC o R^2 ajustado.

Tanto en el caso de forward como backward stepwise, se busca el mejor modelo sólo entre $1+p(p+1)/2$ modelos, lo que permite su uso cuando p es demasiado grande para seleccionarlos mediante la búsqueda del mejor subconjunto.

El método backward necesita que el número de observaciones n sea mayor que el de variables predictoras p (ya que se necesita ajustar el modelo con todas las variables). Por el contrario, el método forward se puede usar incluso cuando $n < p$.

3.2.2.3. Procedimiento con R: la función `regsubset()`

La función `regsubset()` permite utilizar el método backward y forward, usando el argumento `method = "forward"` o `method = "backward"`:

```
regfit_fwd <- regsubsets(Salary ~ .,
  data = Hitters,
  nvmax = 19, method = "forward"
)
regfit_bwd <- regsubsets(Salary ~ .,
  data = Hitters,
  nvmax = 19, method = "backward"
)
```

A continuación se ve como, por ejemplo, para el caso del mejor modelo con 2 variables los tres métodos: mejor subconjunto, forward y backward dan lugar conjuntos de variables diferentes:

```
coef(regfit_full, 2)
#> (Intercept)      Hits       CRBI
#> -47.9559022   3.3008446  0.6898994
coef(regfit_fwd, 2)
#> (Intercept)      Hits       CRBI
```

3.2. Selección del mejor subconjunto

55

```
#> -47.9559022  3.3008446  0.6898994
coef(regfit_bwd, 2)
#> (Intercept)      Hits       CRuns
#> -50.8174029   3.2257212  0.6614168
```

Hay que decidir un criterio para elegir el mejor modelo: R^2 ajustado, BIC , etc. Si se usa R^2 ajustado, en ambos casos el mejor modelo es el que tiene 11 variables:

```
which.max(summary(regfit_fwd)$adjr2)
#> [1] 11
which.max(summary(regfit_bwd)$adjr2)
#> [1] 11
```

Con BIC , el modelo elegido no tiene el mismo número de variables:

```
which.min(summary(regfit_fwd)$bic)
#> [1] 6
which.min(summary(regfit_bwd)$bic)
#> [1] 8
```

Otra posibilidad es utilizar como criterio el error de predicción, y para ello se puede utilizar algún esquema de validación cruzada. A continuación se ilustra el caso en el que se divide la muestra en dos subconjuntos: *training* y *testing*, pero se puede utilizar cualquier otro método (*k-fold*, etc.).

```
set.seed(1)
entreno <- sample(c(TRUE, FALSE), nrow(Hitters), replace = TRUE)
test <- (!entreno)
```

Se utiliza `regsubsets()` en la muestra de entrenamiento para obtener los modelos con distinto número de variables predictoras:

```
regfit_best <- regsubsets(Salary ~ ., data = Hitters[entreno, ], nvmax = 19)
```

Para calcular el error de predicción, dado que la función `regsubset()` no tiene asociada una función `predict()`, se han de calcular “manualmente” los valores predichos para la muestra de test. Para eso se necesita la matriz de diseño del modelo.

```
test.mat <- model.matrix(Salary ~ ., data = Hitters[test, ])
```

Ahora, para cada modelo de tamaño k , se extraen los coeficientes de `regfit_best` para el mejor modelo de ese tamaño, se multiplica el vector de coeficientes por la matriz de diseño y se obtienen las predicciones; a continuación se calcula el error cuadrático medio (MSE).

```
val_errors <- rep(NA, 19)
for (i in 1:19) {
  coefi <- coef(regfit_best, id = i)
  pred <- test.mat[, names(coefi)] %*% coefi
  val_errors[i] <- mean((Hitters$Salary[test] - pred)^2)
}
```

El mejor modelo es el que contiene 7 variables:

```
val_errors
#> [1] 164377.3 144405.5 152175.7 145198.4 137902.1 139175.7 126849.0 136191.4
#> [9] 132889.6 135434.9 136963.3 140694.9 140690.9 141951.2 141508.2 142164.4
#> [17] 141767.4 142339.6 142238.2
```

3.3. Métodos *shrinkage*

Los métodos anteriores se basan en el ajuste de modelos mediante mínimos cuadrados. Ahora se trabajará con un método diferente: ***shrinkage***. Este método se basa en una modificación de mínimos cuadrados añadiendo una penalización que *encoje* los coeficientes del modelo típicamente hacia 0. Una de las ventajas de este método es que reduce la varianza de los coeficientes estimados.

3.3.1. Regresión *ridge*

Se recuerda que el ajuste por mínimos cuadrados estima $\beta_0, \beta_1, \dots, \beta_p$ mediante los valores que minimizan:

$$RSS = \sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2.$$

La **regresión ridge** añade un término de penalización controlado por un parámetro (que habrá que elegir) que penalizará los coeficientes que se hacen demasiado grandes. Cuanto más grande es el coeficiente mayor es la penalización:

$$\sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^p \beta_j^2 = RSS + \lambda \sum_{j=1}^p \beta_j^2. \quad (3.1)$$

En realidad lo que se está haciendo es hacer pagar al modelo un precio por el hecho de que los coeficientes no sean cero, y el precio será tanto mayor cuanto mayor sea la magnitud del coeficiente. A esta penalización se le llama **penalización shrinkage** porque “anima” a los coeficientes a que se *contraigan* hacia 0. La magnitud de dicha contracción está gobernada por lambda, el parametro de afinado o regulación (también conocido en la jerga como de tuneado). Si $\lambda = 0$ se está en el caso de mínimos cuadrados, y cuanto mayor sea λ , mayor será el precio a pagar

3.3. Métodos shrinkage

57

para que esos coeficientes sean distintos de 0. Si λ es extremadamente grande los coeficientes estarán muy próximos a 0 para poder hacer el segundo término pequeño (recuérdese que se quiere minimizar RSS más la penalización). Aunque valores más grandes de los coeficientes den un mejor ajuste (y por lo tanto un menor RSS), el término de penalización se hará grande y no se alcanzará el mínimo. Por lo tanto λ sirve como equilibrio entre un buen ajuste del modelo y el tamaño de los coeficientes (y por lo tanto el número de coeficientes distintos de cero).

Elegir un buen valor de λ es crítico. Se utilizará la validación cruzada para elegirlo.

3.3.1.1. Escalado de variables predictoras

Un punto importante en regresión ridge es si las variables predictoras están escaladas o no.

En el caso de mínimos cuadrados, el método es *invariante a la escala* (*scale-invariant*), es decir, que si se multiplica una variable predictora X_j por una constante c , esto solo implica que el coeficiente estimado se ve multiplicado por $1/c$, pero $X_j\hat{\beta}_j$ no cambia. Sin embargo, en el caso de la regresión *ridge* los coeficientes estimados pueden cambiar sustancialmente si se multiplica una variable predictora por una constante, ya que aparecen todos juntos en el término de penalización. Por lo tanto, antes de utilizar la regresión *ridge* (o cualquier método de regularización) es importante **estandarizar las variables predictoras**, dividiendo cada variable por su desviación estándar, de forma que todas tengan desviación estándar igual a 1.

$$\tilde{x}_{ij} = \frac{x_{ij}}{\sqrt{\frac{1}{N} \sum_{i=1}^N (x_{ij} - \bar{x}_{ij})^2}}$$

Con esto se consigue que los coeficientes sean comparables.

La regresión *ridge* en muchas ocasiones da lugar a un menor MSE que el obtenido con mínimos cuadrados ordinarios. Sin embargo, por muy grande que sea λ los coeficientes no serán 0, sino que estarán próximos a cero, por lo que este método no es realmente un método de selección de variables.

La regresión ridge puede ser muy útil cuando hay variables predictoras altamente correlacionadas, pero se desea mantener todas en el modelo. En estos casos, la regresión ridge soluciona los problemas de multicolinealidad.

3.3.1.2. Procedimiento con R: la función `glmnet()`

El paquete que se va a usar para llevar a cabo la regresión *ridge* (y para otros métodos de regresión *shrinkage*) es `glmnet`. La función principal en este paquete se llama también `glmnet()`. Esta función tiene una sintaxis un poco diferente a las funciones usuales para el ajuste de distintos modelos en *R*. Es necesario pasarle la matriz X de variables predictoras (sin la columna correspondiente a la ordenada en el origen), y el vector Y con la variable respuesta. Para ilustrar su uso se utilizarán los datos anteriores sobre béisbol.

```
x <- model.matrix(Salary ~ ., Hitters)[, -1]
y <- Hitters$Salary
```

La función `glmnet()` tiene un argumento, `alpha`, que determina el tipo de penalización que se añade en el modelo. En el caso de regresión ridge `alpha=0`.

Por defecto, la función `glmnet()` elige de forma automática el rango de valores de λ . Sin embargo, a modo ilustrativo, se va a elegir la rejilla de valores, desde $\lambda = 10^{10}$ hasta $\lambda = 10^{-2}$, cubriendo de esta forma una gran gama de escenarios, desde el modelo nulo (solo la ordenada en el origen), hasta el caso de mínimos cuadrados. Se verá más adelante que se puede llevar a cabo el ajuste del modelo para un valor determinado de λ que no esté entre los de la rejilla inicial.

```
library("glmnet")
grid <- 10^seq(10, -2, length = 100)
ridge_mod <- glmnet(x, y, alpha = 0, lambda = grid)
```

Por defecto, la función `glmnet()` estandariza las variables predictoras para que estén en la misma escala. Si por alguna razón no se quisiera hacer, se usaría `standardize = FALSE`.

Asociado con cada valor de λ hay un vector de coeficientes estimados mediante regresión ridge almacenados en un matriz accesible utilizando `coef()`. En este caso, el tamaño de la matriz es 20×100 , las 20 filas corresponden a cada uno de los predictores más la ordenada en el origen, y las 100 columnas a cada valor de λ . Lo esperable es que los coeficientes estimados sean más pequeños cuanto mayor sea el valor de λ . A continuación se muestra el valor de los coeficientes cuando $\lambda = 11,498$, así como la suma de sus cuadrados, $\sum_{j=1}^p \beta_j^2$:

```
ridge_mod$lambda[50]
#> [1] 11497.57
sum(coef(ridge_mod)[-1, 50]^2)
#> [1] 40.45739
```

Por el contrario, si λ es más pequeño, 705, el valor es mucho mayor.

```
ridge_mod$lambda[60]
#> [1] 705.4802
sum(coef(ridge_mod)[-1, 60]^2)
#> [1] 3261.554
```

A continuación se dibuja el efecto de λ en los coeficientes (véase Fig. 3.3):

```
plot(ridge_mod, xvar = "lambda", label = TRUE)
```

El lado izquierdo de la Fig 3.3 corresponde a un valor de λ muy pequeño, y por lo tanto no existen restricciones sobre los coeficientes. Conforme aumenta el valor de λ los coeficientes se

3.3. Métodos shrinkage

59

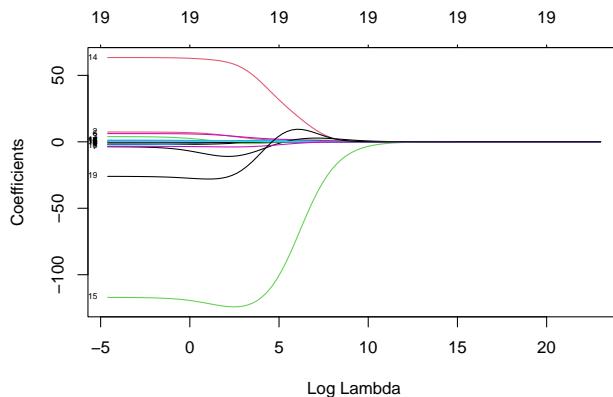


Figura 3.3: Coeficientes estimados para distintos valores del parámetro de regularización (en la escala logarítmica).

aproximan a cero, ya que el precio a pagar por ser distinto de cero es cada vez mayor. Pero no todos se aproximan a cero de la misma manera: hay un conjunto de variables cuyo coeficiente es prácticamente cero para cualquier valor de λ , mientras que para un valor de $\log(\lambda) = 3$ parece que hay solo 4 coeficientes distintos de 0.

La función `predict()` se puede utilizar con diferentes propósitos. Por ejemplo, se pueden obtener los coeficientes de la regresión ridge para un valor específico de λ , por ejemplo $\lambda = 50$:

```
predict(ridge_mod, s = 50, type = "coefficients")[1:20, ]
```

Ahora se van a dividir los datos en una muestra de entrenamiento y otra de test para estimar el error de predicción de la regresión *ridge*.

```
set.seed(1)
entreno <- sample(1:nrow(x), nrow(x) / 2)
test <- (-entreno)
y_test <- y[test]
```

Se ajusta la regresión ridge a la muestra de entrenamiento usando un valor de λ (por ejemplo $\lambda = 4$), y se evalúa su *MSE* en la muestra de test. Para ello se usará la función `predict()`. En este caso, para obtener las predicciones para la muestra de test, se reemplaza `type = "coefficients"` por el argumento `newx`.

```
ridge_mod <- glmnet(x[entreno, ], y[entreno], alpha = 0, lambda = grid)
ridge_pred <- predict(ridge_mod, s = 4, newx = x[test, ])
mean((ridge_pred - y_test)^2)
#> [1] 142226.5
```

El *MSE* es 142,199. Si se usa un valor muy alto de λ , por ejemplo 10^{10} (esto sería equivalente a ajustar un modelo solo con la ordenada en el origen), el resultado es muy distinto:

```
ridge_pred <- predict(ridge_mod, s = 1e10, newx = x[test, ])
mean((ridge_pred - y_test)^2)
#> [1] 224669.8
```

Por lo tanto, en este caso, ajustar un modelo de regresión ridge con $\lambda = 4$ da un *MSE* mucho menor que el obtenido cuando el modelo sólo contiene la ordenada en el origen.

A continuación se compara el resultado para $\lambda = 4$ con el obtenido utilizando mínimos cuadrados ($\lambda = 0$).¹

```
ridge_pred <- predict(ridge_mod, s = 0, newx = x[test, ], exact = T,
                      x = x[entreno, ], y = y[entreno])
mean((ridge_pred - y_test)^2)
#> [1] 167018.2
```

Se observa que el error es menor cuando se usa regresión ridge (con $\lambda = 4$) que cuando se usan mínimos cuadrados.

Hasta ahora se ha elegido el valor $\lambda = 4$ de forma arbitraria, en la siguiente Sección se ve cómo seleccionar el valor de dicho parámetro de una forma automática.

3.3.2. Selección del parámetro de regularización

Se ha visto que el valor de λ tiene un gran impacto en los resultados obtenidos cuando se utiliza un modelo con penalización.

Una buena manera de elegir λ es usar validación cruzada (*cross-validation*), por ejemplo, se puede usar *k-fold cross-validation*:

- Se dividen los datos en k grupos, se ajusta el modelo ridge a $k - 1$ de esos grupos (para una rejilla de valores de λ) y se calcula el error de predicción para el otro grupo.
- La acción anterior se repite tomando como muestra de test cada uno de los k grupos y se suman los errores de predicción.
- Al final se dispondrá de una curva con los errores para cada valor de λ y se elegirá el que dé el mínimo error.

En la práctica, el procedimiento anterior se puede hacer con la función `cv.glmnet()`. Por defecto, esta función usa un *10-fold cross-validation*, pero se puede cambiar usando el argumento `nfolds`.

En el ejemplo del béisbol:

¹Además se ha de añadir ‘exact = T’ en la función ‘predict()’

3.3. Métodos shrinkage

61

```
set.seed(1)
cv_out <- cv.glmnet(x[entreno, ], y[entreno], alpha = 0)
plot(cv_out)
```

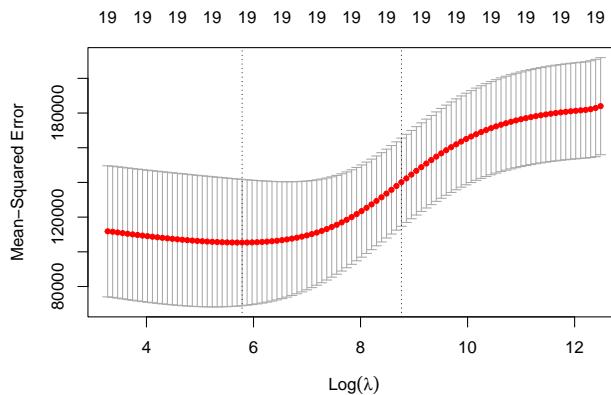


Figura 3.4: Valor del error cuadrático medio y su intervalo de cofianza (calculado sobre los 10-fold) para distintos valores del parámetro de regularización.

```
mejorlam <- cv_out$lambda.min
mejorlam
#> [1] 326.0828
```

En la Fig. 3.4, los puntos rojos corresponden a la media del MSE para los k -folds y las barras superior e inferior corresponden a esa cantidad más/menos una desviación estándar (el ancho será tanto menor cuanto mayor sea k en el k -fold). La primera línea vertical corresponde al valor de λ que hace mínimo el MSE y la segunda es el valor que está a una distancia de una desviación típica del λ mínimo (usar esta valor podría ser una buena opción para evitar el sobre-ajuste, es decir dejar demasiadas variables en el modelo).

Se calcula el valor mínimo del MSE :

```
ridge_pred <- predict(ridge_mod, s = mejorlam, newx = x[test, ])
mean((ridge_pred - y_test)^2)
#> [1] 139833.6
```

Como se puede apreciar, hay una mejora sobre el error de predicción que se había obtenido cuando $\lambda = 4$.

3.3.3. Regresión *lasso*

Uno de los puntos débiles de la regresión ridge es que no hace selección de variables (los coeficientes pueden ser próximos a cero pero no exactamente cero). En el modelo final se incluyen todos los coeficientes y, por lo tanto, la regresión *ridge* sólo es útil cuando la mayoría de las variables predictoras son tienen un impacto significativo en la respuesta.

La regresión *lasso*, introducida por Tibshirani (1996), es una alternativa a la regresión *ridge* cuyo objetivo es precisamente eliminar esa desventaja de esta técnica, y es útil cuando la mayoría de las variables predictoras no son relevantes en el modelo. Los coeficientes *lasso*, $\hat{\beta}^L$, minimizan la siguiente cantidad:

$$\sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^p |\beta_j| = RSS + \lambda \sum_{j=1}^p |\beta_j|$$

Ahora los coeficientes se *contraen* hacia cero utilizando el valor absoluto en vez de la suma de cuadrados. A esta norma se le llama l_1 , $\|\beta\|_1 = \sum_{j=1}^p |\beta_j|$. El cambio que supone es sutil pero importante. En ambos casos los coeficientes se contraen hacia 0, pero en el caso de la regresión *lasso* cuando λ es suficientemente grande los coeficientes serán 0, de modo que se estará haciendo una selección de variables. Es decir, hará los coeficientes exactamente igual a 0 si esas variables no son importantes y λ es suficientemente grande. En este sentido *lasso* es lo que se llama un **modelo sparse** (un modelo con un número *sparse, o escaso, de parámetros).

¿Por qué *lasso* hace que los coeficiente se contraigan exactamente hacia cero? Para entenderlo se va a ver una formulación equivalente a la de los mínimos cuadrados penalizados en el caso de la regresión *lasso*:

$$\sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 \quad \text{sujeto a} \quad \sum_{j=1}^p |\beta_j| < s$$

Se está utilizando mínimos cuadrados con una restricción, o lo que es lo mismo, con un *presupuesto* en la norma l_1 sobre los coeficientes. Las dos formulaciones son equivalentes en el sentido de que si se tiene un *presupuesto* s , habrá un λ en la formulación previa que corresponda al mismo problema y viceversa. Supóngase que se hacen mínimos cuadrados y se obtienen unas ciertas estimaciones de los parámetros (coeficientes) tal que la suma de sus valores absolutos es 10, pero alguien dice que nuestro *presupuesto* es 5 (la suma de los valores absolutos de los coeficientes no puede ser mayor que esa cantidad). Ahora, hay que resolver el problema de mínimos cuadrados pero coeficientes no pueden tomar cualquier valor, ya que se tiene una restricción sobre los mismos. Cuanto más pequeño sea el *presupuesto*, más próximos a cero serán los coeficientes. Si el *presupuesto* es 0, todos los coeficientes serán también 0. Si el presupuesto es muy alto, hay libertad para que los coeficientes tomen el valor que quieran, y se estaría en el caso de mínimos cuadrados. El *presupuesto* impone que haya un equilibrio entre el ajuste a los datos y el tamaño de los coeficientes.

La Fig. 3.5 (tomada de James et al. (2013)) muestra por qué lasso es *sparse*:

El gráfico corresponde a un modelo de regresión con dos variables predictoras. El punto donde está el vector de coeficientes, $\hat{\beta}$, es donde se alcanzaría el valor mínimo de los mínimos cuadrados

3.3. Métodos shrinkage

63

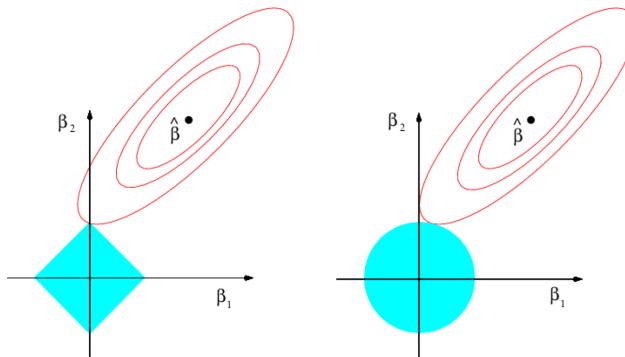


Figura 3.5: Contornos (rojo) de RSS y regiones de restricción (en azul) para lasso (izquierda) y ridge (derecha).

(RSS), los contornos serían combinaciones de valores de β_1 y β_2 que dan lugar al mismo valor de RSS pero que ya no sería el mínimo. Las regiones de restricción son $|\beta_1| + |\beta_2| < s$ (*lasso*) y $\beta_1^2 + \beta_2^2 < s$ (*ridge*). En el caso de *ridge*, el *presupuesto* sería el radio del círculo, y la regresión *ridge* busca el primer lugar en el que el contorno toca a la región de restricción, pero al ser un círculo, difícilmente uno u otro coeficiente va a ser 0. En el caso de *lasso* la región es un diamante, y por lo tanto tiene vértices, en la Fig. 3.5 el contorno toca a la región de restricción en el caso en que $\beta_1 = 0$.

Se vuelve al ejemplo del béisbol para mostrar la regresión lasso; en este caso el argumento α toma valor 1:

```
lasso_mod <- glmnet(x[entreno, ], y[entreno], alpha = 1, lambda = grid)
plot(lasso_mod)
```

En la Fig. 3.6 Se puede ver, que dependiendo del valor del parámetro de regularización, algunos de los coeficientes se hacen exactamente 0. Para elegir el valor de dicho parámetro y calcular el error de predicción resultante se procede como sigue:

```
set.seed(1)
cv_out <- cv.glmnet(x[entreno, ], y[entreno], alpha = 1)
plot(cv_out)

mejorlab <- cv_out$lambda.min
lasso.pred <- predict(lasso_mod, s = mejorlab, newx = x[test, ])
mean((lasso.pred - y_test)^2)
#> [1] 143673.6
```

Este valor es bastante más bajo que MSE en la muestra de test en el caso de mínimos cuadrados (224669,8), y bastante parecido al obtenido con regresión *ridge* (cuando el parámetro de regularización se elige mediante validación cruzada, 139856,6).

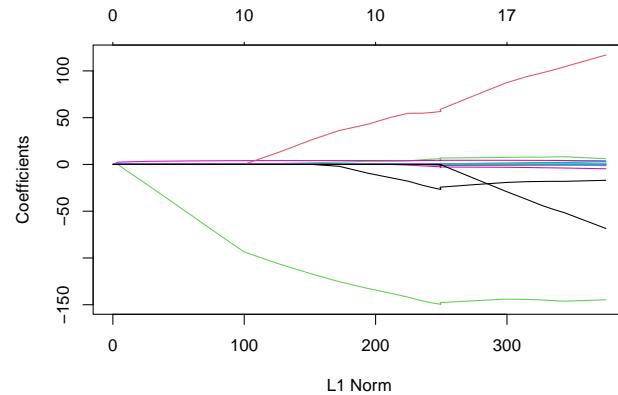


Figura 3.6: Valor de los parámetros estimados para distintos valores de la penalización (que depende del parámetros de regularización.)

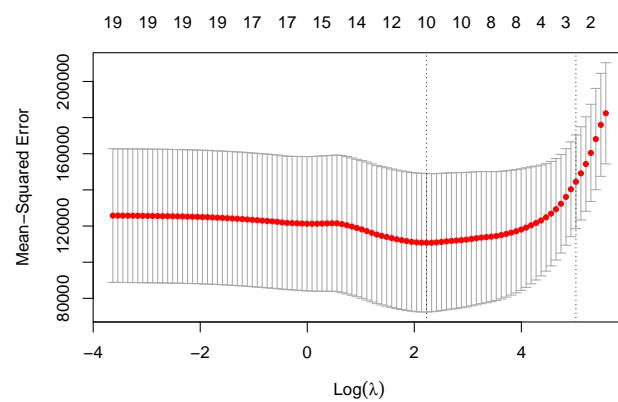


Figura 3.7: Valor del error cuadrático medio y su intervalo de confianza para distintos valores del parámetro de regularización.

3.3. Métodos shrinkage

65

Sin embargo, *lasso* tiene una ventaja importante con respecto a la regresión *ridge* ya que los coeficientes estimados son *sparse*. En los resultados que se muestran a continuación, se puede observar que 10 de los 20 coeficientes estimados son 0. Por lo tanto, el modelo *lasso* con λ elegido mediante validación cruzada contiene solo nueve variables predictoras.

```
out <- glmnet(x, y, alpha = 1)
lasso_coef <- predict(out, type = "coefficients", s = mejorlab)[1:20, ]
lasso_coef[lasso_coef != 0]
#>   (Intercept)      Hits       Walks      CHmRun      CRuns
#> -3.04787656  2.02551572  2.26853781  0.01647106  0.21177390
#>     CRBI      LeagueN    DivisionW     PutOuts     Errors
#>  0.41944632 20.48456551 -116.59062083  0.23718459 -0.94739923
```

3.3.4. *Elastic net*

Uno de los problemas de la regresión *lasso* es cuando hay variables predictoras correladas entre sí, pues elegirá una de ellas (y los coeficientes de las demás los hará cero) sin un criterio objetivo. Además, supóngase que se está en una situación en la que el número de variables p es mayor que el número de observaciones N , en este caso *lasso* elegiría como mucho N variables; mientras que la regresión *ridge* las utilizaría todas, aunque disminuye la complejidad del modelo (esto en algunos casos puede ser lo deseable o no). *Elastic net* (Zou and Hastie, 2005) es una generalización de los métodos anteriores que combina la penalización *ridge* y la *lasso*:

$$\sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 + \lambda_1 \sum_{j=1}^p \beta_j^2 + \lambda_2 \sum_{j=1}^p |\beta_j|.$$

También aparece en muchas ocasiones de esta otra forma:

$$\sum_{i=1}^n \left(y_i - \beta_0 - \sum_{j=1}^p \beta_j x_{ij} \right)^2 + \lambda \left[\frac{1}{2}(1-\alpha) \sum_{j=1}^p \beta_j^2 + \alpha \sum_{j=1}^p |\beta_j| \right]$$

donde $\alpha \in [0, 1]$. Se puede ver α como el parámetro que controla la mezcla entre las dos penalizaciones y λ como el que controla la cantidad de penalización. Si $\alpha = 0$ se está en el caso de regresión *ridge*, y si $\alpha = 1$ en el caso de regresión *lasso*.

La función `glmnet()` también sirve para ajustar *elastic net*, pero el parámetro α hay que elegirlo a priori. Otra opción es utilizar el paquete `caret` para hacer validación cruzada sobre α y λ simultáneamente:

```
set.seed(1)
library("caret")
cv_glmnet <- train(
  x = x[entreno, ],
  y = y[entreno],
  method = "glmnet",
```

```

trControl = trainControl(method = "cv", number = 10),
tuneLength = 10
)
# modelo con el MSE más pequeño
cv_glmnet$bestTune
#>   alpha    lambda
#> 9    0.1 99.12337
ggplot(cv_glmnet)

```

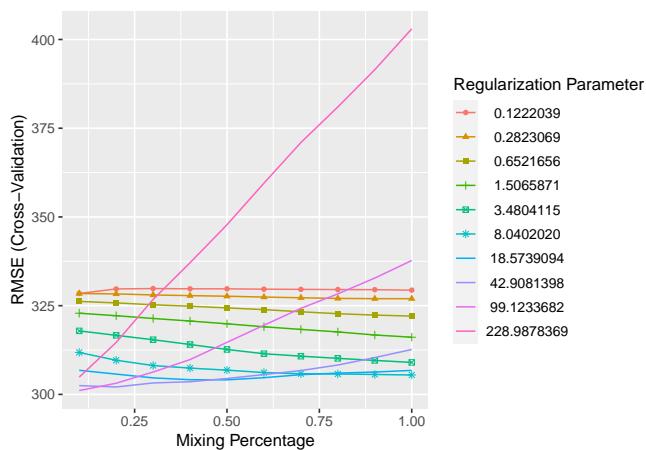


Figura 3.8: Valor de la raíz cuadrada del error cuadrático medio para distintas combinaciones de α y λ .

La Fig. 3.8 muestra como la combinación de α y λ da lugar a diferentes MSE (aquí aparece el $RMSE$, o sea, su raíz cuadrada). Cada línea corresponde a un valor de λ distinto, y en el eje x se representan los valores de α .

Se calcula el error de predicción para estos dos valores de los parámetros de regularización:

```

elastic_mod <- glmnet(x[entreno, ], y[entreno], alpha = cv_glmnet$bestTune$alpha)
elastic_pred <- predict(elastic_mod, newx = x[test, ], s = cv_glmnet$bestTune$lambda)
mean((elastic_pred - y_test)^2)
#> [1] 141626.1

```

Se puede comprobar que es peor que el de la regresión *ridge*, pero mejor que el de *lasso*. Si no se quiere hacer ningún tipo de selección, en este caso se elegiría *ridge*, pero si se quiere reducir al máximo el número de variables predictoras se usaría *lasso* (a costa de que el error de predicción aumente) y el equilibrio vendría con el uso de elastic-net que hace selección de variables pero no aumenta el error de predicción.

Existen otros métodos de regularización que se derivan de estos, como el *group lasso*, *sparse group-lasso*, etc. Se puede encontrar información de estos métodos en (Hastie and Tibshirani, 2015).

Resumen

En este capítulo se introducen una serie de técnicas para mejorar la predicción y la interpretabilidad de los modelos de regresión, en particular:

- Se muestra el uso de la técnica de selección del mejor subconjunto de variables en el modelo, así como los métodos *stepwise*.
- Se presentan 3 métodos tipo *shrinkage*: regresión *ridge*, *lasso* y *elastic net*, bien para la selección de variables, o para solventar problemas de multicolinealidad en el modelo.
- Se muestra cómo seleccionar los parámetros de regularización que controlan la regresión penalizada.
- Se ilustra el uso de todas las metodologías propuestas en el capítulo mediante el análisis de un caso práctico.

Bibliografía

- De Boor, C. (2001). *A practical guide to splines*. Applied Mathematical Sciences. Springer-Verlag, New York.
- Hastie, T. and Tibshirani, R. (2015). *Statistical Learning with Sparsity: The Lasso and Generalizations*. Monographs on Statistics & Applied Probability. Chapman and Hall/CRC.
- Henderson, C. (1953). Estimation of variance and covariance components. *Biometrics*, 9:226–252.
- James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An introduction to statistical learning*, volume 112. Springer. <https://www.statlearning.com/>.
- Singer, J. and Willet, J. (2003). *Applied Longitudinal Data Analysis*. Oxford University Press.
- Snijders, T. (2003). Fixed and random effects. 2:664–665.
- Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B*, 58:267–288.
- Wood, S. N. (2006). *Generalized Additive Models - An introduction with R*. Texts in Statistical Science. Chapman & Hall.
- Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B*, 67:301–320.

Índice alfabético

- agregación, 30
- base, 14
- componentes principales, 50
- datos
 - jerárquicos, 30
 - longitudinales, 31
- desagregación, 30
- deviance, 15, 50
- ecuaciones de Henderson, 33
- efectos
 - aleatorios, 31
 - fijos, 31
- elastic net, 65
- estandarización, 57
- grados de libertad efectivos, 14
- intercambiabilidad, 31
- k-fold cross-validation, 60
- modelo
 - aditivo, 12
 - sparse, 62
- modelos mixtos lineales, 33
- parámetro de suavizado, 12
- penalización shrinkage, 56
- regresión lasso, 62
- regresión ridge, 56
- residuos
 - estandarizados, 35
 - parciales, 21
 - studentizados, 35
- selección de variables, 49
- selección stepwise, 52
 - backward, 53
 - forward, 53
- Shrinkage, 50
- shrinkage, 56
- spline, 18
- subconjunto
 - testing, 55
 - training, 55