Facultad de Matemáticas

Departamento de Estadística e Investigación Operativa

Trabajo Fin de Grado en Matemáticas

TÉCNICAS DE REGULARIZACIÓN EN REGRESIÓN: IMPLEMENTACIÓN Y APLICACIONES

María Carrasco Carrasco

19 de junio de 2016

Dirigido por:
Dr. Juan Manuel Muñoz Pichardo

Índice general

Introducción

En Estadística, los métodos de regularización son utilizados para la selección del modelo y para evitar el sobreajuste en las técnicas predictivas. A la hora de estimar los coeficientes de regresión por mínimos cuadrados, nos podemos encontrar con el problema de colinealidad. Este problema impide obtener estimaciones y predicciones fiables a través de mínimos cuadrados, por lo que se ha de recurrir a los métodos de regresión regularizada como son Ridge, Lasso y Elastic Net, que son los tres métodos que enfocaremos en este trabajo. Estos métodos también se utilizan cuando el número de predictores es muy grande, ya que utilizando algunos de ellos podemos descartar algunas variables y crear nuestro modelo más simple e interpretable.
En este trabajo se recoge la descripción teórica de las técnicas citadas, se ilustran las mismas aplicando el paquete "glmnet" de R-Program.

Capítulo 1

Regresión lineal

Este primer capítulo estará dedicado a estudiar de forma breve en que consiste un modelo de regresión. Se dividirá en diferentes secciones de la siguiente manera: en la Sección 1.1 comenzaremos introduciendo de forma breve el modelo de regresión lineal. A continuación en la Sección 1.2 abordaremos el problema de la estimación de los parámetros del modelo mediante el método de mínimos cuadrados. Seguidamente en la Sección 1.3 motivaremos la necesidad de la selección de variables en un modelo de regresión. En la sección 1.4 se introduce el concepto de regresión sesgada y por último en la Sección 1.5 introduciremos el problema de colinealidad que nos encontramos en un análisis de regresión y cómo detectarla.

1.1. Modelo de regresión lineal

En la regresión lineal aleatoria se intenta describir y predecir el comportamiento de una variable $y \in \mathbb{R}$ a partir de un vector de variables explicativas o predictoras $\underline{x}$.
El modelo expresa la esperanza condicionada de la variable objetivo como combinación lineal de las variables explicativas. Es decir, el modelo se define con las hipótesis distribucional y estructural siguientes:

- **Distribucional:** $Y|_{\underline{X}=\underline{x}} \sim N(\mu(\underline{x}), \sigma^2)$
- **Estructural:** $\mu(\underline{x}) = \beta_0 + \beta_1 x_1 + ... + \beta_p x_p$

El modelo muestral en este caso, teniendo y como la variable respuesta u objetivo, y $\underline{x}$ como el vector con las variables explicativas sería:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_p x_p + \varepsilon, \quad \varepsilon \sim N(0, \sigma^2)$$

Así, para una muestra aleatoria $(y_i, \underline{x}_i)$ $i = 1, ..., n$ o conjunto de datos que contenga n casos de la población bajo estudio extraídos de forma independiente y aleatoria, se tiene:

$$y_i = \beta_0 + \beta_1 x_{1i} + ... + \beta_p x_{pi} + \varepsilon_i, \quad \varepsilon_i \sim N(0, \sigma^2) \text{ independientes}$$

y en forma matricial

$$\underline{Y} = X\underline{\beta} + \underline{\varepsilon}$$

A partir de ahora se tiene en cuenta que β es un vector p-dimensional.

Los coeficientes de regresión $\beta_0, ..., \beta_p$ son desconocidos, luego deben ser estimados. Para realizar un análisis de regresión lineal múltiple se realizan las siguientes consideraciones sobre los datos:

- Linealidad
- Homocedasticidad
- Independencia
- Normalidad
- Las variables explicativas se obtienen sin errores de medida

Si los datos cumplen estas hipótesis entonces pasamos a estimar los coeficientes por el método de los mínimos cuadrados, MCO, ya que los estimadores de los parámetros son insesgados y tienen mínima varianza. Mas adelante nos centraremos en el caso de que no sea lo acertado calcular la estimación de los coeficientes por el método de los mínimos cuadrados.

Supuestos en la regresión lineal

Además de suponer que $\underline{Y} = X\beta + \underline{\varepsilon}$ y que la matriz X es no aleatoria, requeriremos lo siguiente:

1. $E[\underline{\varepsilon}] = \underline{0}$
2. $E[\underline{\varepsilon}\underline{\varepsilon}'] = \sigma^2 I_n$
3. $rango(X) = p < N$

El supuesto 1) no implica pérdida de generalidad ni supone ninguna restricción, al menos en el caso en que X tiene entre sus columnas una cuyos valores sean constantes (y ésto suele suceder; típicamente, la primera columna está formada por "unos").

El supuesto 2), bastante más restrictivo, dado que requiere que las perturbaciones sean incorreladas (covarianzas cero) y homoscedásticas (de idéntica varianza).

El supuesto 3) simplemente fuerza la independencia lineal entre las (p) columnas de X. El requerimiento $N > p$ excluye de nuestra consideración el caso $N = p$, pues entonces $\underline{y} = X\hat{\beta}$ es un sistema de ecuaciones lineales determinado, y tiene siempre solución para algún vector $\hat{\beta}$ que hace los residuos nulos. Las estimaciones del vector β se obtendrían entonces resolviendo dicho sistema.

1.2. Estimación por mínimos cuadrados

El método MCO tiene como objetivo minimizar la suma de los cuadrados de los residuos:

$$RSS(\beta) = \|Y - X\beta\|_2^2 = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j X_{ij} \right)^2$$

El estimador de β por MCO viene dado por:

$$\hat{\beta}^{MCO} = (X'X)^{-1}X'\underline{y} \quad \varepsilon \sim N(0, \sigma^2 I_n) \quad , \quad \hat{\beta}^{MCO} \sim N_p(\beta, \sigma^2(X'X)^{-1})$$

Suponemos que rango(X) = p, entonces $(X'X)$ es de rango completo y, por tanto, posee inversa. Este estimador es único en el caso que las columnas de X formen un conjunto linealmente independiente, y bajo el supuesto anterior de normalidad($\varepsilon \sim N(0, \sigma^2 I_n)$), $\hat{\beta}^{MCO}$ es el estimador de mínima varianza de β en la clase de estimadores lineales e insesgados (teorema de Gauss-Markov).

Problemas con MCO

Hay dos razones por las que el método de mínimos cuadrados podría no ser adecuado para estimar modelos con variables no relevantes, es decir, con

poca capacidad predictora. Estas son:

- **Baja precisión en las predicciones**. El estimador a menudo presenta poco sesgo pero gran varianza, lo cual se traduce en un pobre poder predictivo sobre nuevas observaciones. La existencia de al menos, una variable que pudiese ser expresada como combinación lineal del resto, conduce a que el determinante de $(X'X)$ sea nulo y por tanto no exista su inversa.
  Si no hay variables que sean combinación lineal de otras pero están fuertemente correlacionadas, provoca inestabilidad en la solución del estimador dado que el determinante de $(X'X)$ será muy cercano a cero.

- **Falta de interpretabilidad**. Si se utiliza un gran número de predictores (necesario para tener bajo sesgo ante un problema más o menos complejo), sería deseable determinar un pequeño subconjunto de éstos con fuerte poder explicativo y predictivo, ya que con $p \gg n$ el estimador no estará bien definido.
  Esta desventaja de MCO también está vinculada a la existencia de predictores fuertemente correlacionadas.

Al estimar los parámetros de regresión por el método de mínimos cuadrados ordinarios, puede que alguna de estas estimaciones sea "casi" cero y por tanto la variable correspondiente a dicho coeficiente tendría muy poca influencia en el modelo, sin embargo, es poco común que estas estimaciones lleguen a tomar exactamente el valor cero. Por tanto, no nos sirve como método de selección de variables. De este modo, necesitaremos de otros métodos para seleccionar variables, algunos de los cuales se recogen en el siguiente apartado.

1.3. Selección de variables

Uno de las cuestiones más importantes a la hora de encontrar el modelo de ajuste más adecuado para explicar la variabilidad de una característica cuantitativa es la correcta especificación del llamado modelo teórico. En muchas situaciones se dispone de un conjunto grande de posibles variables explicativas, una posible pregunta sería saber si todas las variables deben entrar en el modelo de regresión y, en caso negativo, saber qué variables deben entrar y cuáles no.
En otras palabras, debemos seleccionar un subconjunto de variables entre todas las variables candidatas a ser explicativas de la variable dependiente,

subconjunto que resulte suficientemente explicativo.
En general, si se incluyen cada vez más variables en un modelo de regresión, el ajuste a los datos mejora, aumenta la cantidad de parámetros a estimar pero disminuye su precisión individual (mayor varianza) y, por tanto, la de la función de regresión estimada, produciéndose un sobreajuste. Por el contrario, si se incluyen menos variables de las necesarias en el modelo, las varianzas se reducen, pero los sesgos aumentarán obteniéndose una mala descripción de los datos.
Por otra parte, algunas variables predictoras pueden perjudicar la confiabilidad del modelo, especialmente si están correlacionadas con otras.
De esta manera, el objetivo de los métodos de selección de variables es buscar un modelo que se ajuste bien a los datos y que, a la vez, sea posible buscar un equilibrio entre bondad de ajuste y sencillez.
En la práctica, no obstante, la selección del subconjunto de variables explicativas de los modelos de regresión se deja en manos de procedimientos más o menos automáticos. Los procedimientos más usuales son los siguientes:

- **MÉTODOS DE MÍNIMOS CUADRADOS PENALIZADOS**
  Se basan en los mínimos cuadrados ordinarios pero añadiendo una penalización en la función objetivo, para forzar que alguna componente del vector de parámetros $\beta = (\beta_1, ..., \beta_p)$ sea cero y de esta manera conseguir estimación de los parámetros y selección de variables conjuntamente.
  De estos métodos ya profundizaremos más adelante ya que es el tema de este trabajo. Estudiamos tres de ellos como son el Ridge, Lasso y Elastic Net.

- **ALGORITMOS**
  Procedimientos secuenciales de selección de variables basados en criterios que conjugan la optimalidad del ajuste y la reducción de la dimensión del espacio predictor. El procedimiento termina cuando se satisface una regla de parada establecida.
  Entre los algoritmos de selección de variables podemos señalar tres de los más usuales:

  - **Selección hacia adelante**:
    Procedimiento de selección de variables en el que las variables se introducen secuencialmente en el modelo. La primera variable que

se considerará introducir en la ecuación será la que tenga mayor correlación, positiva o negativa, con la variable dependiente. Dicha variable se introducirá en la ecuación sólo si cumple el criterio de entrada. Si se introduce la primera variable, a continuación se considerará la variable independiente cuya correlación parcial sea la mayor y que no esté en la ecuación. El procedimiento termina cuando ya no quedan variables que cumplan el criterio de entrada.

- **Selección hacia atrás**:
  Procedimiento de selección de variables en el que se introducen todas las variables en la ecuación y después se van excluyendo una tras otra. Aquella variable que tenga la menor correlación parcial con la variable dependiente será la primera en ser considerada para su eliminación. Si satisface el criterio de eliminación, se eliminará. Tras haber excluido la primera variable, se pondrá a prueba aquella variable, de las que queden en la ecuación, que presente una correlación parcial más pequeña. El procedimiento termina cuando ya no quedan en la ecuación variables que satisfagan el criterio de eliminación.

- **Regresión paso a paso**:
  Este procedimiento es una combinación de los dos anteriores.
  Comienza como el de introducción progresiva, pero en cada etapa se plantea si todas las variables introducidas deben de permanecer en el modelo.

Cuando se aplica este tipo de procedimientos tenemos que tener en cuenta cuál será la condición para suprimir o incluir un término. Para ello podemos considerar dos criterios:

- **Criterios de significación**: En un método de eliminación hacia atrás se suprimirá el término que resulte menos significativo, y en un método de selección hacia adelante se añadirá el término que al añadirlo al modelo resulte más significativo. Un criterio de significación puede ser la significación de cada coeficiente.

- **Criterios globales**: Una medida global de cada modelo, de modo que tenga en cuenta el ajuste y el exceso de parámetros. Como criterios destacamos el Criterio de Información de Akaike, AIC (Akaike, [1]) definido por

$$AIC = -2ln(L) + k$$

donde k es el número de parámetros en el modelo (en nuestro caso, k=p+2) y L es el máximo valor de la función de verosimilitud en el modelo ajustado y el Criterio de Información de Bayes, BIC (Schwarz, [2]) definido por

$$BIC = -2lnL + kln(n)$$

Se trata de buscar un modelo cuyo AIC o BIC sea pequeño, ya que en ese caso habría una verosimilitud muy grande y pocos parámetros.

* **Consecuencias de los métodos de selección de variables**

Dos inconvenientes de estos métodos son que realizan un proceso discreto de exploración del espacio de modelos(cada variable es seleccionada o descartada) y trae consigo una fuerte inestabilidad, pequeños cambios en el conjunto de datos pueden producir grandes modificaciones en los resultados y que resultan inaplicables cuando el número de variables p es similar o incluso superior al número de observaciones n.
Para ello utilizamos los métodos de mínimos cuadrados penalizados citados anteriormente.

1.4. Regresión sesgada

Como ya vimos antes, los estimadores MCO son los de mínima varianza en la clase de los estimadores lineales insesgados. Cualesquiera otros que consideremos, si son lineales y de varianza menor, habrán de ser sesgados. Si consideramos adecuado como criterio en la elección de un estimador $\hat{c}$ su error cuadrático medio, $ECM \stackrel{def}{=} E[\hat{c} - c]^2$, y reparamos en que:

$$\begin{aligned} E[\hat{c} - c]^2 &= E[\hat{c} - E[\hat{c}] + E[\hat{c}] - c]^2 \\ &= E[\hat{c} - E[\hat{c}]]^2 + E[E[\hat{c}] - c]^2 + 2\underbrace{E[\hat{c} - E[\hat{c}]][E[\hat{c}] - c]}_{=0} \\ &= var(\hat{c}) + (sesgo(\hat{c}))^2 \end{aligned}$$

Podemos plantearnos la siguiente pregunta: ¿Es posible reducir el ECM en la estimación tolerando un sesgo? Si la respuesta fuera afirmativa, podríamos preferir el estimador resultante que, aunque sesgado, tendría un ECM menor, producido por una disminución en la varianza capaz de compensar el segundo sumando.

Los métodos de regresión sesgada se contemplan a veces como alternativas a los métodos de selección de variables en situaciones de acusada multicolinealidad. Nos ocuparemos de procedimientos "ad-hoc" para reducir la varianza de los estimadores.

No se profundiza más sobre la regresión sesgada en este trabajo, se puede obtener más información acerca de este tema en el capítulo 10 del trabajo de F. Tusell [3].

1.5. Problema de colinealidad

La colinealidad es uno de los mayores inconvenientes que nos podemos encontrar en un análisis de regresión. Si en un modelo de regresión lineal múltiple alguna variable independiente es combinación lineal de otras, el método MCO es irresoluble, debido a que, en ese caso, la matriz $(X'X)$ es singular, es decir, su determinante es cero y no se puede invertir. A este fenómeno se le denomina colinealidad.
Otro modo, por tanto, de definir la colinealidad es cuando alguno de los coeficientes de correlación simple o múltiple entre algunas de las variables independientes es 1, es decir, cuando algunas variables independientes están correlacionadas entre sí.
En la práctica, esta colinealidad exacta raras veces ocurre, pero sí surge con cierta frecuencia la llamada "casi" colinealidad, o por extensión, simplemente colinealidad en que alguna variable es "casi" combinación lineal de otra u otras, o dicho de otro modo, algunos coeficientes de correlación simple o múltiple entre las variables independientes están cercanos a 1, aunque no llegan a dicho valor.
En este caso la matriz $(X'X)$ es "casi" singular, es decir su determinante no es cero pero es muy pequeño. Como para invertir una matriz hay que dividir por su determinante, en esta situación surgen problemas de precisión en la estimación de los coeficientes, ya que los algoritmos de inversión de matrices pierden precisión al tener que dividir por un número muy pequeño, siendo además inestables.
Además, como la matriz de varianzas de los estimadores es proporcional a $(X'X)$, resulta que en presencia de colinealidad los errores estándar de los coeficientes son grandes (no hay precisión en sentido estadístico).
Es importante señalar que el problema de multicolinealidad, en mayor o menor grado, se plantea porque no existe información suficiente para conseguir una estimación precisa de los parámetros del modelo.

Detección de la colinealidad

A la hora de plantear modelos de regresión lineal múltiple conviene estudiar previamente la existencia de colinealidad.

- **Factor de inflación de la varianza**

  Como medida de la misma hay varios estadísticos propuestos, los más sencillos son los coeficientes de determinación o cuadrados de los coeficientes de correlación múltiple de cada variable independiente con todas las demás, es decir

  $$R_i^2 = R^2_{X_i|X_1,...,X_{i-1},X_{i+1},...,X_n}, \qquad i = 1, ..., n$$

  y, relacionados con ellos, el factor de inflación de la varianza (FIV) y la tolerancia (T), definidos como

  $$FIV_i = \frac{1}{1 - R_i^2}, \qquad T_i = \frac{1}{FIV_i} = 1 - R_i^2$$

  Una regla empírica, citada por Kleinbaum y otros [4], consiste en considerar que existen problemas de colinealidad si algún FIV es superior a 10, que corresponde a algún $R_i^2 > 0,9$ y $T_i < 0,1$.
  Aunque puede existir colinealidad con FIV bajos, además puede haber colinealidades que no impliquen a todas las variables independientes y que, por tanto, no son bien detectadas por el FIV.

- **Número de condición**

  Este procedimiento de detección de la multicolinealidad es el más adecuado entre los actualmente disponibles, según afirman Judge et al. [5]. El número de condición,$k(X)$, es igual a la raíz cuadrada de la razón entre la raíz característica más grande (λ_{max}) y la raíz característica más pequeña (λ_{min}) de la matriz $(X'X)$, es decir,

  $$k(X) = \sqrt{\frac{\lambda_{max}}{\lambda_{min}}}$$

  Si la matriz $(X'X)$ es de dimensión $n \times n$ se obtienen n raíces características, pudiéndose calcular para cada una de ellas un índice de condición definido de la siguiente forma:

$$ic(\lambda_i) = \sqrt{\frac{\lambda_{max}}{\lambda_i}}$$

El número de condición mide la sensibilidad de las estimaciones mínimo-cuadráticas ante pequeños cambios en los datos.
De acuerdo con los estudios realizados, tanto con datos observados como con datos simulados, el problema de la multicolinealidad es grave cuando el número de condición toma un valor entre 20 y 30. Naturalmente, si este indicador superase el valor de 30, el problema sería ya manifiestamente grave. Estos valores vienen generalmente referidos a regresores medidos con escala de longitud unidad (es decir, con los regresores divididos por la raíz cuadrada de la suma de los valores de las observaciones), pero no centrados. Parece que no es conveniente centrar los datos (es decir, restarles sus correspondientes medias), ya que esta operación oscurece cualquier dependencia lineal que implique al término independiente.

Capítulo 2

Técnicas de regularización

Una formulación genérica de las técnicas de regularización en el contexto de modelos lineales puede realizarse de la siguiente manera:

$$\hat{\beta} = \underset{\beta}{arg\ min} \left\{ \sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{p} \beta_j X_{ij} \right)^2 + \phi_\lambda(\beta) \right\}$$

$$donde\ \beta = (\beta_1, ..., \beta_p), \quad \lambda \geq 0 \quad y \quad \phi_\lambda(\beta) = \lambda \sum_{j=1}^{p} \phi_j(|\beta_j|)$$

es la función creciente de penalización sobre el "tamaño" de β, que depende de λ.
Una familia de funciones de penalización muy utilizada es la correspondiente a la norma-Lq, dada por

$$\phi_\lambda(\beta) = \lambda(\|\beta\|_q)^q = \lambda \sum_{j=1}^{p} |\beta_j|^q, \quad q > 0$$

Los estimadores resultantes en estos casos son conocidos como estimadores Bridge(Fu,[6]). En este capítulo, se aborda en primer lugar la regresión Ridge, que tiene por función la anterior con norma L2; seguidamente la regresión Lasso, con norma L1; luego la regresión Elastic Net, que será la que estudiemos con más detalle ya que las anteriores son casos particulares de esta (si $\alpha = 0$ estaríamos en el caso de regresión Ridge y $\alpha = 1$ regresión Lasso), y por último se verá de qué manera se elige el parámetro lambda en esos tres métodos anteriores, se explica y se ve el algoritmo utilizado.

2.1. Métodos de penalización ó métodos de mínimos cuadrados penalizados

2.1.1. Regresión Ridge

Esta técnica fue propuesta originalmente por Hoerl y Kennard [7] como un método para eludir los efectos adversos del problema de colinealidad en un modelo lineal estimado por mínimos cuadrados, en el contexto $p < n$. Regresión Ridge es muy similar a los mínimos cuadrados, a excepción de que los coeficientes se estiman minimizando una cantidad diferente.
Los coeficientes estimados por Ridge, $\hat{\beta}^{ridge}$, son los valores que minimizan

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

donde $\lambda \geq 0$ es el parámetro de contracción que se determinará por separado.

El método Ridge tiende a contraer los coeficientes de regresión al incluir el término de penalización en la función objetivo: cuanto mayor sea λ, mayor penalización y, por tanto, mayor contracción de los coeficientes.

Una forma equivalente de escribir el problema Ridge es:

$$\hat{\beta}^{ridge} = \underset{\beta}{arg\,min} \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} X_{ij} \beta_j \right)^2 \quad s.a. \ \sum_{j=1}^{p} \beta_j^2 \leq t,$$

donde t es el parámetro de penalización por complejidad.

Sabemos que $\hat{\beta}^{MCO} = (X'X)^{-1}X'Y$ es la estimación por mínimos cuadrados de β. Se planteó que la poca estabilidad de $\hat{\beta}^{MCO}$ podría ser aliviada agregando una pequeña constante, $\lambda \geq 0$, a cada término de la diagonal de $(X'X)$ antes de invertir la matriz. Nos encontramos con:

$$\hat{\beta}^{ridge}(\lambda) = (X'X + \lambda I_p)^{-1}X'Y,$$

donde I es la matriz $p \times p$. Hay que tener en cuenta que esta penalización se le aplica a los coeficientes $\beta_1, ..., \beta_p$ pero no a β_0. Ajustamos el modelo sin término independiente, estimando este mediante $\overline{y} = \sum_{i=1}^{n} \frac{y_i}{n}$. Es excluido de la penalización para evitar que el resultado dependa del origen en la variable Y.

A diferencia del método de mínimos cuadrados, regresión Ridge produce un conjunto diferente para cada valor de λ y no un único vector de coeficientes estimados.
si $\lambda = 0$ estamos en el caso de mínimos cuadrados ordinarios. En otro caso, $\lambda \to \infty$, $\hat{\beta} \to 0$ y estamos ante un estimador sesgado de β. En resumen, introducimos sesgo pero reducimos la varianza.
Para evitar que la penalización varíe frente a cambios de escalas de las variables, habitualmente estas son estandarizadas previamente(media 0 y varianza 1). También Y se supone centrada, por lo que la matriz X tendrá p columnas y no $p + 1$.
En general, regresión Ridge produce predicciones más precisas que MCO y selección de variables.

Una vez que tengamos estimado nuestros coeficientes hay que buscar el valor de λ, $0 < \lambda < \infty$, con el propósito de minimizar una estimación del error de predicción esperado. Tanto para éste método como para los siguientes utilizaremos para conocer ése parámetro el método de validación cruzada que se explicará mas adelante.

Uno de los inconvenientes de este método es que contrae todos los coeficientes hacia cero, pero sin conseguir la nulidad de ninguno de ellos. Por tanto, no se produce selección de variables, permaneciendo en el modelo todas las variables. Este hecho resulta un inconveniente en aquellos estudios que tienen un elevado número p de variables explicativas o predictores.
Para eludir este inconveniente se propuso la regresión Lasso, incluida en el siguiente apartado.

2.1.2. Regresión Lasso

Motivado por el objetivo de encontrar una técnica de regresión lineal que, mediante la contracción de los coeficientes, lograra estabilizar las estimaciones y predicciones y que realizase selección de variables, Tibshirani [8], propuso la técnica Lasso (least absolute shrinkage and selection operator).
Es una técnica de regresión lineal regularizada, como Ridge, con una leve diferencia en la penalización que trae consecuencias importantes.
En especial, a partir de cierto valor del parámetro de complejidad el estimador de Lasso produce estimaciones nulas para algunos coeficientes y no nulas para otros, con lo cual Lasso realiza una especie de selección de variables en forma continua, debido a la norma L1. Lasso reduce la variabilidad de las estimaciones por la reducción de los coeficientes y al mismo tiempo produce modelos interpretables por la reducción de algunos coeficientes a cero.

El auge en los últimos años en la investigación y aplicación de técnicas Lasso se debe, principalmente, a la existencia de problemas donde $p \gg n$ y al desarrollo paralelo de algoritmos eficientes (Tibshirani, [9]).

Lasso resuelve el problema de mínimos cuadrados con restricción sobre la norma L1 del vector de coeficientes:

$$\hat{\beta}^{lasso} = \min_{\beta} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j X_{ij} \right)^2 \right\} \quad s.a. \ \sum_{j=1}^{p} |\beta_j| \leq s$$

o de forma equivalente, minimizando

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

siendo s y $\lambda \geq 0$ los respectivos parámetros de penalización por complejidad. De nuevo reparametrizamos la constante β_0 mediante la estandarización de los predictores, y por tanto, la solución $\hat{\beta}_0 = \overline{y}$.

En general, los modelos generalizados Lasso son mucho mas fáciles de interpretar que los obtenido mediante Ridge. Al igual que antes, queda por buscar el mejor valor del parámetro λ por el método de validación cruzada.

Utilizando Lasso tendríamos un modelo con buena precisión e interpretable, pero este método también tiene varias limitaciones como son:

- En el caso $p > n$, Lasso selecciona a lo sumo n variables antes de saturarse, debido a la naturaleza del problema de optimización convexa. Esto parece ser una limitación para un método de selección de variables. Además, Lasso no está bien definido a menos que el límite de la norma L1 de los coeficientes sea menor que un cierto valor.

- Si hay un grupo de variables entre las cuales las correlaciones por parejas son muy altas, entonces Lasso tiende a seleccionar sólo una variable del grupo, sin importarle cuál de ellas selecciona.

- Para el caso $n > p$, si hay una alta correlación entre predictores, se ha observado que, en general, la predicción a través de regresión Ridge resulta más optima que la obtenida a través de Lasso.

Algunos estudios comparativos entre Lasso y Ridge llegan a las siguientes conclusiones:

1. Está claro que Lasso al hacer selección de variables tiene una gran ventaja sobre la regresion Ridge, ya que produce modelos más simples y más interpretables que implica un único subconjunto de los predictores. Sin embargo, no hay un método que siempre domine al otro.

2. En general, se podría esperar que Lasso fuese mejor en un entorno en el que un número relativamente pequeño de predictores tiene coeficientes sustanciales, y los restantes predictores tienen coeficientes que son muy pequeños o iguales a cero.

3. Regresión Ridge obtiene mejores resultados cuando la respuesta es una función de muchos factores predictivos, todos con coeficientes de aproximadamente el mismo tamaño.
   Sin embargo, el número de predictores que se relaciona con la respuesta no se conoce a priori para los conjuntos de datos reales.

4. Una técnica tal como la validación cruzada se puede utilizar con el fin de determinar qué enfoque es mejor en un conjunto de datos particular.

5. Al igual que la regresión Ridge, cuando las estimaciones de mínimos cuadrados tiene excesivamente alta varianza, la solución Lasso puede producir una reducción de la varianza a expensas de un pequeño aumento de sesgo, y por tanto, puede generar predicciones mas exactas.

2.1.3. Elastic Net

Zou y Hastie [10], proponen una nueva técnica de regularización y selección de variables conocido como Elastic Net, la cual retiene las ventajas de Lasso, hace automáticamente selección de variables y contracción continua, y al mismo tiempo supera algunas de sus limitaciones. Con este nuevo método se puede seleccionar grupos de variables correlacionadas.
Este método es particularmente útil cuando el número de predictores (P) es mucho más grande que el número de observaciones (n).
En primer lugar, los autores definen "Naive Elastic Net" (red elástica simple), que es un método de mínimos cuadrados penalizado utilizando una penalización nueva de Elastic Net.

Naive Elastic Net

Para cualesquiera λ_1 y λ_2 constantes fijas no negativas, se define el criterio "Elastic Net" simple por:

$$L(\lambda_1, \lambda_2, \beta) = |y - X\beta|^2 + \lambda_2|\beta|^2 + \lambda_1|\beta|_1$$

$$\text{donde } |\beta|^2 = \sum_{j=1}^{p} \beta_j^2 \quad , \quad |\beta|_1 = \sum_{j=1}^{p} |\beta_j| \quad \text{y} \quad |y - X\beta|^2 = RSS$$

El estimador obtenido por esta técnica, es decir, el vector que minimiza la ecuación anterior

$$\hat{\beta}^{ene} = arg \underset{\beta}{min}\ L(\lambda_1, \lambda_2, \beta)$$

se denomina estimador "Elastic Net" simple.

Considerando $\alpha = \frac{\lambda_2}{\lambda_1 + \lambda_2}$, tenemos el siguiente problema de optimización:

$$\hat{\beta}^{ene} = arg \underset{\beta}{min} |y - X\beta|^2$$

$$s.a : \alpha|\beta|_1 + (1 - \alpha)|\beta|^2 \leq t \quad \text{para algún t}$$

Llamamos a la función $\alpha|\beta|_1 + (1-\alpha)|\beta|^2$ la penalización Elastic Net, que es una combinación convexa de las penalizaciones Lasso y Ridge.
A continuación se desarrolla un método para resolver el problema de Elastic Net de manera eficiente, basado en el hecho de que minimizar la ecuación $L(\lambda_1, \lambda_2, \beta)$ es equivalente a un problema de optimización tipo Lasso.

Lema 1. *Dado un conjunto de datos (y, X) y los parámetros (λ_1, λ_2) no negativos, se considera el conjunto de datos artificial (y^*, X^*):*

$$X^*_{(n+p)xp} = (1 + \lambda_2)^{-1/2} \begin{pmatrix} X \\ \sqrt{\lambda_2} I \end{pmatrix}, \quad y^*_{(n+p)} = \begin{pmatrix} y \\ 0 \end{pmatrix}$$

Denotamos $\gamma = \frac{\lambda_1}{\sqrt{1+\lambda_2}}$ y $\beta^* = \sqrt{(1 + \lambda_2)}\beta$.

Entonces, el criterio de Elastic Net simple puede ser escrito como:

$$L(\gamma, \beta) = L(\gamma, \beta^*) = |y^* - X^*\beta^*|^2 + \gamma|\beta^*|_1$$

Considerando $\hat{\beta}^* = arg \underset{\beta^*}{min}\ L(\gamma, \beta^*)$ tenemos entonces $\hat{\beta} = \frac{1}{\sqrt{1+\lambda_2}}\hat{\beta}^*$

El lema 1 muestra que se puede transformar el problema Elastic Net simple en un problema Lasso equivalente con datos aumentados o transformados. El tamaño de la muestra en el problema con datos aumentados es $(n+p)$ y X^* tiene rango p, lo que significa que este método puede seleccionar todos los p predictores en todas las situaciones. Esta importante propiedad supera la primera de las limitaciones de Lasso.
El lema 1 también muestra que este método realiza selección de variables agrupadas, una propiedad que no es compartida por el método Lasso.
En el caso $p \gg n$, la situación de las variables "agrupadas" tienen gran importancia. Una ilustración de esta afirmación es el estudio detallado de Segal y otros [11] que motiva fuertemente el uso de un procedimiento de regresión regularizado para encontrar los genes agrupados.
Consideramos el método de penalización genérica:

$$\hat{\beta}^{nen} = \underset{\beta}{arg\ min}\ |y - X\beta|^2 + \lambda J(\beta), \tag{2.1}$$

donde $J(.)$ es positivo para $\beta \neq 0$

Cualitativamente hablando, un método de regresión presenta el efecto de agrupamiento si los coeficientes de regresión de un grupo de variables altamente correlacionadas tienden a ser igual (hasta un cambio de signo si hay correlación negativa).
En particular, en la situación extrema en la que algunas variables son exactamente idénticas, el método de regresión debe asignar coeficientes idénticos a las variables idénticas.

Lema 2. *Asumiendo que las variables x_i, x_j son iguales, $x_i = x_j$, $i, j \in (1, ..., p)$*

1. *Si $J(.)$ es estrictamente convexa, entonces $\hat{\beta}_i = \hat{\beta}_j,\ \forall \lambda > 0$.*
2. *Si $J(\beta) = |\beta|_1$, entonces $\hat{\beta}_i \hat{\beta}_j \geq 0$ y $\hat{\beta}^*$ es otro minimizador de la ecuación (2.1), donde:*

$$\hat{\beta}_k^* = \begin{cases} \hat{\beta}_k, & si\ k \neq i\ y\ k \neq j \\ (\hat{\beta}_i + \hat{\beta}_j)s, & si\ k = i \\ (\hat{\beta}_i + \hat{\beta}_j)(1-s), & si\ k = j \end{cases} \qquad para\ cualquier\ s \in [0,1]$$

El **lema 2** muestra una clara distinción entre las funciones de penalización estrictamente convexa y la penalización Lasso. La convexidad estricta

garantiza el efecto de agrupación en la situación extrema con predictores idénticos. Por el contrario, Lasso ni siquiera tiene una única solución.
La penalización de Elastic Net con $\lambda_2 > 0$ es estrictamente convexa, así:

Teorema 1. *Considérense los datos (y, X) y los parámetros no negativos (λ_1, λ_2), con la respuesta y centrada y los predictores X estandarizados. Sea $\hat{\beta}(\lambda_1, \lambda_2)$ el estimador del criterio Elastic Net simple. Supóngase que $\hat{\beta}_i(\lambda_1, \lambda_2)\hat{\beta}_j(\lambda_1, \lambda_2) > 0$, y sea*

$$D_{\lambda_1,\lambda_2}(i,j) = \frac{1}{|y|_1}|\hat{\beta}_i(\lambda_1, \lambda_2) - \hat{\beta}_j(\lambda_1, \lambda_2)|$$

entonces:

$$D_{\lambda_1,\lambda_2}(i,j) \leq \frac{1}{\lambda_2}\sqrt{2(1-\rho)}$$

donde $\rho = \underline{x}_i'\underline{x}_j$ es el coeficiente de correlación muestral entre las variables x_i y x_j.

Así, si ambas variables están altamente correladas ($\rho \approx 1$), la diferencia entre los estimadores de los coeficientes de regresión asociados es casi nula.
Como un método de selección de variable, este último método supera las dos primeras limitaciones de Lasso citadas anteriormente, pero tiene una deficiencia, y es que no se lleva a cabo de manera satisfactoria a menos que sea muy cercano a la regresión Ridge o a Lasso. Por este motivo se le denomina Elastic Net simple.
Un método de penalización precisa logra un buen rendimiento de predicción a través de la compensación entre sesgo y varianza. El estimador Elastic Net simple es un procedimiento de dos etapas: fijamos λ_2 y encontramos en primer lugar los coeficientes de regresión Ridge, y luego se hace la contracción de tipo Lasso. Por tanto, el procedimiento aplica una doble contracción que no ayuda a reducir las varianzas e introduce sesgo adicional innecesario, en comparación con el Lasso o contracción Ridge. En la siguiente sección se mejora la predicción del rendimiento de este método mediante la corrección de esta doble contracción.

Elastic Net "corregido"

Dado un conjunto de datos (y, X), los parámetros de penalización (λ_1, λ_2) y los datos aumentados (y^*, X^*), Elastic Net simple resuelve el siguiente problema tipo Lasso:

$$\hat{\beta}^* = \underset{\beta^*}{arg\ min}\ |y^* - X^*\beta^*|^2 + \frac{\lambda_1}{\sqrt{1+\lambda_2}}|\beta^*|_1$$

Zou y Hastie (2005) proponen la siguiente estimación del parámetro β según el criterio Elastic Net corregido:

$$\hat{\beta}^{enet} = \sqrt{(1+\lambda_2)}\hat{\beta}^{*}$$

Recordemos que

$$\hat{\beta}^{nen} = \frac{1}{\sqrt{1+\lambda_2}}\hat{\beta}^{*}$$

entonces se tiene:

$$\hat{\beta}^{enet} = (1+\lambda_2)\hat{\beta}^{nen}$$

Así, $\hat{\beta}^{enet}$ es un estimador Naive Elastic Net "reescalado". Esta transformación de escala preserva la propiedad de selección de variables del Elastic Net simple y es un camino más sencillo para deshacer la contracción y, por tanto, disminuir el sesgo. El siguiente teorema da otra presentación de Elastic Net, en la que el argumento de descorrelación es más explícito.

Teorema 2. *Considerando los datos* (y, X) *y* (λ_1, λ_2)*, entonces el estimador* $\hat{\beta}$ *viene dado por:*

$$\hat{\beta} = \underset{\beta}{arg\ min}\ \beta' \left(\frac{X'X + \lambda_2 I}{1+\lambda_2} \right) \beta - 2y'X\beta + \lambda_1 |\beta|_1$$

El teorema interpreta Elastic Net como una versión estabilizada de Lasso, dado que el estimador del criterio Lasso es:

$$\hat{\beta}^{lasso} = \underset{\beta}{arg\ min}\ \beta'(X'X)\beta - 2y'X\beta + \lambda_1 |\beta|_1$$

donde $(X^T X)$ es una versión muestral de la matriz de correlaciones de las variables explicativas y

$$\frac{1}{(1+\lambda_2)}[X'X + \lambda_2 I] = (1-\gamma)(X'X) + \gamma I$$

con $\gamma = \frac{\lambda_2}{1+\lambda_2}$ es una contracción de $(X'X)$ hacia la matriz identidad.

Computación

Zou y Hastie (2005) proponen un algoritmo llamado LARS-EN para resolver el método Elastic Net de manera eficiente, basado en el algoritmo LARS de Efron et al. [12], propuesto para obtener la solución del problema Lasso.

Por el lema 1, para cada λ_2 fijo, el problema Elastic Net es equivalente a un problema Lasso en el conjunto de datos aumentados o transformados. Así, el algoritmo LARS se puede utilizar directamente para calcular el estimador Elastic Net de manera eficiente con los esfuerzos computacionales de un ajuste por MCO.
Se ha de señalar que para $p \gg n$, el conjunto de datos aumentados tiene $p+n$ "observaciones" y p variables, lo que puede ralentizar considerablemente el cálculo.
Con objeto de facilitar los cálculos, Zou y Hastie (2005) proponen algunas modificaciones que, con más detalles, se pueden encontrar en el apartado 3.4 del citado trabajo.

2.2. Elección del parámetro λ

Como puede observarse todas estas técnicas de mínimos cuadrados penalizados dependen de un parámetro de penalización λ, que controla la importancia dada a la penalización en el proceso de optimización. Cuanto mayor es λ mayor es la penalización en los coeficientes de regresión y más son contraídos éstos hacia cero.
La elección de éste parámetro involucra un balance entre los componentes de sesgo y varianza del ECM al estimar β.
Una propuesta inicial y que continúa siendo sugerida por algunos autores es la utilización de una traza Ridge para determinar λ. Consiste en graficar simultáneamente los coeficientes de regresión estimados en función de λ, y elegir el valor más pequeño del parámetro para el cuál se estabilizan dichos coeficientes.
Un método más automático, pero intensivo computacionalmente, consiste en estimar λ mediante validación cruzada (en general se recomiendan utilizar ambos métodos y comparar resultados).
El método de validación cruzada consiste en dividir el modelo en un set de entrenamiento (training set) para ajustar un modelo y un set de prueba (test set) para evaluar su capacidad predictiva, mediante el error de predicción u otra medida.
La forma en que se aplica la validación cruzada es mediante la división del conjunto de datos disponibles de manera aleatoria en k subconjuntos o pliegues de igual tamaño y mutuamente excluyentes.
Uno de los subconjuntos se utiliza como datos de prueba y el resto (K-1) como datos de entrenamiento. El proceso de validación cruzada es repetido durante k iteraciones, con cada uno de los posibles subconjuntos de datos de prueba. Finalmente se realiza la media aritmética de los resultados de cada

iteración para obtener un único resultado. Este método es muy preciso puesto que evaluamos a partir de K combinaciones de datos de entrenamiento y de prueba, pero aun así tiene una desventaja, y es que es lento desde el punto de vista computacional.
La validación cruzada con $k = 10$ es una de las más utilizadas, pero hay que tener en cuenta el número de observaciones del que disponemos.
Nuestro valor del parámetro será el que nos de el mínimo error.
A continuación, se ve el algoritmo que se lleva a cabo para obtener ese valor del parámetro, explicado anteriormente:

Validación cruzada con K-pliegues

1. Se divide el conjunto de datos D en K subconjuntos de igual tamaño (partición) $D_1, ..., D_K$. Generalmente, K toma los valores 5 o 10.

2. Para cada $k = 1, ..., K$

   - Se ajusta el modelo $\hat{f}_{-k}^{(\lambda)}(\underline{z})$ con el conjunto de entrenamiento $D - D_k$ (exclusión del k-ésimo pliegue).
   - Se calcula el error por validación cruzada,

$$(CVError)_k^{(\lambda)} = \frac{1}{|D_k|} \sum_{\underline{z} \in D_k} [y - \hat{f}_{-k}^{(\lambda)}(\underline{z})]^2$$

3. Criterio: minimizar el error global de validación cruzada:

$$\lambda^* = \underset{\lambda}{arg\,min}\ (CVError)^{(\lambda)} = \underset{\lambda}{arg\,min}\ \left[\frac{1}{K}\sum_{k=1}^{K}(CVError)_k^{(\lambda)}\right]$$

Leave-one-out CV (validación cruzada dejando uno fuera) es un caso especial con $k = n$.

Capítulo 3

Regularización en R

R, desarrollado inicialmente por Robert Gentleman y Ross Ihaka en 1993, es un lenguaje y entorno de programación para el análisis estadístico y gráfico. Probablemente, uno de los más utilizados por la comunidad estadística por su versatilidad y rendimiento en el análisis de grandes masas de datos y sus capacidades computacionales. Posee una gran variedad de bibliotecas o paquetes sobre diversas finalidades.
En este capítulo se analiza el paquete "glmnet", desarrollado bajo el título "Lasso and Elastic Net Regularized Linear Models", por J. Friedman, T. Hastie, N. Simon y R. Tibshirani, en 2008.

En este capítulo se explicará de forma breve el paquete glmnet, que es el usado para obtener los estimadores de los parámetros tanto para el método Ridge, Lasso o Elastic Net. Posteriormente, se realizará un ejemplo práctico utilizando dicho paquete, para ilustrar lo explicado anteriormente en teoría. Se decidirá, de manera personal, el mejor método según los criterios elegidos.

3.1. Breve descripción del paquete glmnet

glmnet es un paquete de R que realiza procedimientos extremadamente eficientes para el montaje de toda la regularización de Ridge, Lasso o Elastic Net para la regresión lineal, regresión logística y los modelos multinomiales, regresión de Poisson y el modelo de Cox.
Los procedimientos y algoritmos implementados en dicho paquete están descritos en el artículo de Frieman y otros [13].
A continuación se explicarán las funciones más importantes del paquete y el procedimiento para realizar el análisis de regresión con el mismo, en nuestro caso lo aplicaremos a la regresión lineal, y luego se realizará un ejemplo

práctico para ilustrar su manejo y funcionalidad y explicar las salidas que nos proporcionan las funciones.

La principal función de este paquete es **glmnet** que ajusta el modelo lineal generalizado a través de la máxima verosimilitud penalizada:

glmnet(x, y, alpha, nlambda, lambda.min.ratio, lambda, ...),

donde x es la matriz de entrada, de dimensión (num. obs)×(num. var); y es la variable respuesta; *alpha* es el parámetro que determina el método de penalización que estamos utilizando, $0 \leq \alpha \leq 1$; y por último *lambda*, que se podría o bien dar un secuencia de valores o que el programa calcule su propia secuencia basado en nlambda (número de valor lambda, por defecto 100) y lambda.min.ratio (el valor más pequeño de lambda). Si se da esa secuencia, los dos atributos anteriores no hay que especificarlos en la función glmnet.
En el paquete se optimiza el criterio

$$|y - X\beta|^2 + \lambda \left[\frac{1-\alpha}{2} |\beta|^2 + \alpha |\beta|_1 \right]$$

Así, para $\alpha = 0$, la función criterio coincide con la función de la regresión Ridge y para $\alpha = 1$ coincide con la regresión Lasso.

Otra función de este paquete es **plot**, que produce un gráfico de la trayectoria de los coeficientes cuando se ajustan mediante la función glmnet:

plot(x, xvar = c("norm", "lambda", "dev"), label = FALSE, ...)

Solo nombraremos el atributo x que sería el modelo glmnet ya creado anteriormente y del que queremos que nos dibuje la trayectoria de sus coeficientes, y *xvar* que dependiendo de la opción que elijamos en este atributo nos representará en el eje x un valor u otro. Por ejemplo si elegimos $xvar =$ "$norm$", nos representa en el eje x la norma L1 de los coeficientes. Los demás atributos se pueden dejar con los valores asignados por defecto.

La función **predict** predice valores ajustados, logit, coeficientes y más objetos ajustados:

predict(object, newx, s=NULL, type = c("link", "response", "coefficients", "nonzero", "class"), exact = FALSE, ...)

object vuelve a ser el modelo en el que queremos predecir;
newx es una matriz de nuevos valores para x en la que se realiza las predicciones. Este último argumento no se utiliza para type = c("coefficients",

"nonzero").
s es el valor del parámetro lambda en el que se requieren las predicciones, que por defecto, s será la secuencia entera que se usa para crear el modelo. El atributo *type* es el tipo de predicción requerida, en nuestro ejemplo utilizaremos sólo dos de ellas como son coefficients (realiza los coeficientes pedidos para el valor s) y nonzero (regresa una lista con los coeficientes distintos de cero para cada valor de s).
Por último, si *exact* = TRUE, y las predicciones se hacen a valores de s no incluidos en el ajuste original, estos valores de s se fusionan con object$lambda y el modelo hay que volver a ajustarlo antes de hacer las predicciones. Si es false (por defecto), entonces la función predict utiliza la interpolación lineal para hacer predicciones para los valores de s que no coinciden con los utilizados en el algoritmo de ajuste.

La función **deviance** extrae la secuencia de la desviación para un objeto glmnet:

deviance(object,...)

donde *object* es el objeto ajustado por glmnet.

Un objeto glmnet tiene componentes dev.ratio y nulldev. La primera es la fracción de la desviación explicada. Los cálculos de desviación incorporan pesos si están presente en el modelo. La desviación se define como $2 * (loglike\_sat - loglike)$, donde $loglike\_sat$ es el logaritmo de la verosimilitud para el modelo saturado (un modelo con parámetros libres para cada observación y loglike es el logaritmo de la verosimilitud alcanzada por el modelo). Desviación nula se define como $2 * (loglike\_sat - loglike(nulo))$, donde el modelo nulo es aquel que solo contiene el parámetro intercept, es decir, modelo constante para todas las observaciones. Por lo tanto $dev.ratio = 1 - desviacion/nulldev$.

La última función de este paquete es **cv.glmnet** que hace validación cruzada en k pliegues, produce un gráfico, y devuelve el valor de lambda que hace mínimo el error:

cv.glmnet(x, y, lambda, alpha, nfols, ...),

donde *x*, *y* y *alpha* son los mismos que en la función glmnet (alpha debe de darse porque de lo contrario toma el valor 1 por defecto y estaríamos en el caso del Lasso), *lambda* es opcional, podemos dar una secuencia o dejar que glmnet seleccione la suya propia y *nfolds* indica el número de pliegues sobre

los que queremos que se realice la validación cruzada (por defecto sería 10).

Estas son las funciones mas importantes de este paquete. También posee la función **plot.cv** que da un gráfico de la curva de validación cruzada y las curvas de desviación estándar superior e inferior , como una función de los valores de lambda utilizados y **predict.cv** que hace las predicciones de un modelo de validación cruzada, utilizando el modelo ajustado anteriormente y el valor óptimo elegido para lambda.

3.2. Ilustración

Consideramos el conjunto de datos HITTERS (Baseball data), incluidos en el paquete de R "ISLR", que contiene los conjuntos de los principales datos de la liga de béisbol de las temporadas de 1986 y 1987 usados en el libro de Games y otros [14].
Este conjunto está formado por 322 observaciones de los principales jugadores de la liga en las siguientes 20 variables:

AtBat Número de veces al bate en 1986
Hits Número de golpes 1986
HmRun Número de home runs 1986, se da cuando el bateador hace contacto con la pelota de una manera que le permita recorrer las bases y anotar una carrera (junto con todos los corredores en base) en la misma jugada, sin que se registre ningún out ni error de la defensa.
Runs Número de carreras 1986
RBI Número de carreras bateadas 1986
Walks Número de veces en 1986 que se otorga la primera base a un bateador cuando el pitcher ha lanzado cuatro bolas malas, el turno del bateador no es computado en sus estadísticas de turnos al bate
Years Número de años en las grandes ligas
CatBat Número de veces al bate durante su carrera
CHits Número de golpes durante su carrera
CHmRun Número de home runs durante su carrera
CRuns Número de carreras durante su carrera
CRBI Número de carreras bateadas durante su carrera
CWalks Número de walks durante su carrera
League Un factor con niveles A y N indicando la liga del jugador a finales de 1986
Division Un factor con niveles E, W indicando la división del jugador a finales de 1986

PutOuts Se anota esta estadística en favor del jugador a la defensiva que concluye una jugada para poner fuera a un corredor en las bases o a quien ha realizado su turno en el bate. Esta variable muestra el número de PutOuts en 1986
Assists Se anota como asistencia a cualquier jugador de la defensa que atrape o toque la pelota previo a la puesta en out del bateador-corredor u otro corredor que corre las bases, incluso si el contacto no es intencional, pero suficiente para .ªyudar.en el out. Esta variable muestra el número de asistencias en 1986
Errors Una estadística en la cual el anotador oficial establece que un jugador a la defensiva permitió a un rival alcanzar una base, o alargar un turno al bate por un mal manejo de la pelota. En esta variable se tiene el número de errores en 1986
Salary Salario en miles de dolares en la jornada inaugural del año 1987
NewLeague Un factor con niveles A y N indicando la liga del jugador a principios de 1987

Con este ejemplo vamos a predecir el salario (salary) de un jugador de béisbol sobre la base de diversas variables asociadas con el rendimiento en el año anterior .
Nuestro objetivo es buscar, dependiendo de los tres métodos de regularización explicados en teoría, el mejor modelo basándonos en el error de predicción y en el número de variables que nos quedaría en el modelo una vez estimados.

Comenzamos descargando los paquetes glmnet y ISLR que incluye los conjuntos de datos asociados con este ejemplo.

```
> install.packages("glmnet")
> install.packages("ISLR")
> library(glmnet)
> library(ISLR)
```

Lo que diferencia un método de regularización de otro a la hora de hacerlo en R es el valor del parámetro α que utilicemos. Como se indicaba anteriormente, la penalización está definida como:

$$(1-\alpha)/2\|\beta\|_2^2 + \alpha\|\beta\|_1$$

Si $\alpha = 1$ estamos en el caso de una penalización tipo Lasso, $\alpha = 0$ penalización Ridge y $0 < \alpha < 1$ penalización de Elastic Net.

Empezamos cargando los datos:

```
> head(Hitters)
                  AtBat Hits HmRun Runs RBI Walks Years CAtBat
-Andy Allanson      293   66     1   30  29    14     1    293
-Alan Ashby         315   81     7   24  38    39    14   3449
-Alvin Davis        479  130    18   66  72    76     3   1624
-Andre Dawson       496  141    20   65  78    37    11   5628
-Andres Galarraga   321   87    10   39  42    30     2    396
-Alfredo Griffin    594  169     4   74  51    35    11   4408

                  Chits CHmRun CRuns CRBI CWalks Leag Div
-Andy Allanson       66      1    30   29     14    A   E
-Alan Ashby         835     69   321  414    375    N   W
-Alvin Davis        457     63   224  266    263    A   W
-Andre Dawson      1575    225   828  838    354    N   E
-Andres Galarraga   101     12    48   46     33    N   E
-Alfredo Griffin   1133     19   501  336    194    A   W

                  PutOuts Assists Err Salary NewLeag
-Andy Allanson        446      33  20     NA       A
-Alan Ashby           632      43  10  475.0       N
-Alvin Davis          880      82  14  480.0       A
-Andre Dawson         200      11   3  500.0       N
-Andres Galarraga     805      40   4   91.5       N
-Alfredo Griffin      282     421  25  750.0       A
```

En esta muestra de los datos vemos por filas los nombres de los jugadores de béisbol y por columnas las variables a estudiar de cada uno de los ellos, que en nuestro caso la que nos interesa es el salario.
Con las siguientes órdenes tendríamos los nombres de las variables y la dimensión:

```
> names(Hitters)
 [1] "AtBat"    "Hits"     "HmRun"    "Runs"    "RBI"
 [6] "Walks"    "Years"    "CAtBat"   "CHits"   "CHmRun"
[11] "CRuns"    "CRBI"     "CWalks"   "Leag"    "Div"
[16] "PutOuts"  "Assists"  "Err"      "Salary"  "NewLeag"
> dim(Hitters)
 [1] 263  20
```

Antes de continuar tenemos que asegurarnos que no faltan valores de ninguna variable, y en el caso de que falten eliminarlas de los datos. Para ello utilizamos la función is.na() que identifica las observaciones que faltan y devuelve un vector de la misma longitud que el vector de entrada, con un "cierto" para los que faltan, y un "falso" para los que no. La función sum() puede entonces ser utilizada para contar todos los elementos que faltan.
En la muestra anterior vemos que el primer jugador no tiene su salario, asique se estudia cuantos jugadores están en su misma situación y pasan a ser eliminados.

```
> sum(is.na(Hitters$Salary))
  [1] 59
```

Por lo tanto vemos que no se dispone de la variable salario para 59 jugadores. La función na.omit() elimina todas las filas donde falten valores de algunas variables.

```
> Hitters =na.omit(Hitters)
> sum(is.na(Hitters))
  [1] 0
```

Una vez que ya tenemos todos los datos en todas las variables de todos los jugadores definimos nuestras variables:

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

Como la primera columna de la matriz x no aporta información la eliminamos. x es una matriz de dimensión (263×19), en la que vemos el nombre de los jugadores por filas y todas las variables quitando el salario por filas. y es el vector del salario de todos los jugadores.
Aunque la función glmnet() selecciona un rango de valores de lambda, en este caso damos ya el valor de ese parámetro que lo obtenemos mediante validación cruzada. Por defecto, la función estandariza las variables, por lo que todas están en la misma escala.

Para obtener el mejor valor de λ tenemos que dividir las muestras en un conjunto de pruebas y otro de entrenamiento con el fin de estimar el test de error de la regresión. Hay dos caminos para hacer esta división. En la que nos vamos a centrar hay que establecer una semilla aleatoria para que los resultados obtenidos sean reproducibles:

```
> set.seed(1)
> train=sample(1:nrow(x),nrow(x)/2)
> test=(-train)
> y.test=y[test]
```

Esto será para los tres métodos igual, ahora nos centramos en cada uno de los métodos para saber cual de ellos funciona mejor según el criterio que queramos escoger.

3.2.1. Ridge

Como ya vimos en teoría, cuando aplicamos el método Ridge tomamos el valor de alpha = 0.
Creamos la función principal que sería:

```
> ridge.mod = glmnet(x[train,],y[train],alpha=0)
```

A continuación vemos que valor de lambda por validación cruzada nos da menor error de predicción:

```
> set.seed(1)
> cv.out=cv.glmnet(x[train,],y[train],alpha=0)
> bestlam=cv.out$lambda.min
> bestlam
  [1] 211.7416
```

donde lambda.min es el valor de lambda que da el mínimo cvm, y cvm es la media del error de validación cruzada.

Por lo tanto el valor de lambda que da el menor error por validación cruzada es 211.7416.
A continuación vemos cuál es el error asociado a este valor del parámetro.

```
> ridge.pred=predict(ridge.mod,s=bestlam,newx=x[test,])
> mean((ridge.pred-y.test)^2)
  [1] 95982.96
```

Finalmente, reajustamos nuestro modelo de regresión ridge en el conjunto de datos, usando el valor de lambda elegido por validación cruzada, y examinamos los coeficientes estimados. Se ha redondeado cada valor de cada coeficiente para tener sólo cuatro decimales por simplificar un poco, ya que da números con muchos decimales.

```
> out=glmnet(x,y,alpha=0)
> ridge.coef=predict(out,type="coefficients",
                     s=bestlam)[1:20,]
> round(head(ridge.coef),4)
(Inter)   AtBat    Hits   HmRun    Runs     RBI
 9.8849  0.0314  1.0088  0.1393  1.1132  0.8732
> sum(ridge.coef!=0)
  [1] 20
```

Como era de esperar ningún coeficiente es cero, ya que regresión Ridge no hace selección de variables.

3.2.2. Lasso

Volvemos a hacer los mismos pasos que para el método anterior cambiando unicamente el valor de alpha, que en este caso será 1.

```
> lasso.mod = glmnet(x[train,],y[train],alpha=1)
```

Obtenemos el valor de lambda que nos da el menor error por validación cruzada:

```
> set.seed(1)
> cv.out1=cv.glmnet(x[train,],y[train],alpha=1)
> bestlam1=cv.out1$lambda.min
> bestlam1
  [1] 16.78016
```

Vemos cuál es el error asociado a este valor del parámetro:

```
> lasso.pred=predict(lasso.mod,s=bestlam1,newx=x[test,])
> mean((lasso.pred-y.test)^2)
  [1] 100838.2
```

Por ultimo, nos interesa ver qué coeficientes llegan a ser cero, ya que las variables de esos coeficientes se descartarían del modelo y obtendríamos un modelo mas simple para trabajar con el.

```
> out1=glmnet(x,y,alpha=1)
> lasso.coef=predict(out1,type="coefficients",
                     s=bestlam1)[1:20,]
> round(head(lasso.coef),4)
(Inter)   AtBat    Hits   HmRun    Runs     RBI
19.5224  0.0000  1.8702  0.0000  0.0000  0.0000
> sum(lasso.coef!=0)
  [1] 8
```

Los coeficientes estimados que no son ceros son los siguientes:

```
> round(lasso.coef[lasso.coef!=0],4)
(Inter)   Hits  Walks  CRuns   CRBI    Leag       Div PutOuts
19.5224 1.8702 2.2188 0.2073 0.4128  1.7592 -103.5051  0.2207
```

Como era de esperar, Lasso "contrae" los coeficientes a cero y, por tanto, hace selección de variables. En este ejemplo vemos que 12 de los 20 coeficientes estimados son exactamente nulos, luego las variables asociadas no estarían en el modelo.

3.2.3. Elastic net

El método de regularización Elastic Net toma valores de alpha entre 0 y 1. Vamos a ir variando los valores que toma para ver con que valor da un mejor modelo.

- **CASO ALPHA = 0.1**

```
> ene.mod =glmnet(x[train,],y[train],alpha =0.1)
```

Calculamos el valor del parámetro lambda mediante la validación cruzada

```
> set.seed(1)
> cv.out2=cv.glmnet(x[train,],y[train],alpha=0.1)
> bestlam2=cv.out2$lambda.min
> bestlam2
  [1] 87.49189
```

Vemos cuál es el error producido por este valor de lambda y los coeficientes estimados por este método:

```
> ene.pred=predict(ene.mod,s=bestlam2,newx=x[test,])
> mean((ene.pred-y.test)^2)
  [1] 98362.68
> out2=glmnet(x,y,alpha=0.1)
> ene.coef=predict(out2,type="coefficients",
                   s=bestlam2)[1:20,]
> round(head(ene.coef),4)
(Inter)   AtBat    Hits   HmRun    Runs     RBI
14.8692  0.0000  1.3359  0.0000  0.7881  0.4398
> sum(ene.coef!=0)
  [1] 14
```

Vemos cuáles son los coeficientes estimados que no son nulos:

```
> round(ene.coef[ene.coef!=0],4)
(Inter)   Hits   Runs    RBI  Walks CAtBat  CHits CHmRun
14.8692 1.3359 0.7881 0.4398 2.0526 0.0033 0.0771 0.4693

 CRuns   CRBI    Leag         Div  PutOuts      Err
0.1561 0.1723 20.7166  -100.4414   0.2128  -0.4686
```

- **CASO ALPHA = 0.25**

```
> ene.mod1 =glmnet(x[train,],y[train],alpha =0.25)
```

Calculamos el valor del parámetro lambda mediante la validación cruzada

```
> set.seed(1)
> cv.out3=cv.glmnet(x[train,],y[train],alpha=0.25)
> bestlam3=cv.out3$lambda.min
> bestlam3
  [1] 55.72473
```

Vemos cuál es el error producido por este valor de lambda y los coeficientes estimados por este método:

```
> ene.pred1=predict(ene.mod1,s=bestlam3,newx=x[test,])
> mean((ene.pred1-y.test)^2)
  [1] 99520.35
> out3=glmnet(x,y,alpha=0.25)
> ene.coef1=predict(out3,type="coefficients",
                    s=bestlam3)[1:20,]
> round(head(ene.coef1),4)
(Inter)    AtBat    Hits   HmRun    Runs     RBI
23.0936  0.0000  1.5623  0.0000  0.4090  0.1438
> sum(ene.coef1!=0)
  [1] 12
```

Vemos cuáles son los coeficientes estimados que no son ceros:

```
> round(ene.coef1[ene.coef1!=0],4)
(Inter)    Hits    Runs     RBI   Walks  CHits   CHmRun
23.0936  1.5623  0.4090  0.1438  2.1521 0.0743   0.4394

 CRuns    CRBI    Leag       Div  PutOuts
0.1652  0.2001  9.3289  -99.6627   0.2153
```

- **CASO ALPHA = 0.5**

```
> ene.mod2 =glmnet(x[train,],y[train],alpha =0.5)
```

Calculamos el valor del parámetro lambda mediante la validación cruzada

```
> set.seed(1)
> cv.out4=cv.glmnet(x[train,],y[train],alpha=0.5)
> bestlam4=cv.out4$lambda.min
> bestlam4
  [1] 30.57891
```

Vemos cuál es el error producido por este valor de lambda y los coeficientes estimados por este método:

```
> ene.pred2=predict(ene.mod2,s=bestlam4,newx=x[test,])
> mean((ene.pred2-y.test)^2)
  [1] 100322.3
> out4=glmnet(x,y,alpha=0.5)
> ene.coef2=predict(out4,type="coefficients",
                    s=bestlam4)[1:20,]
> round(head(ene.coef2),4)
(Inter)   AtBat    Hits   HmRun    Runs     RBI
19.7715  0.0000  1.8380  0.0000  0.0000  0.0000
> sum(ene.coef2!=0)
  [1] 10
```

Vemos cuáles son los coeficientes estimados que no son ceros:

```
> round(ene.coef2[ene.coef2!=0],4)
(Inter)    Hits   Walks   CHits  CHmRun   CRuns
19.7715  1.8380  2.2490  0.0536  0.3517  0.1803

  CRBI    Leag         Div  PutOuts
0.2510  5.6345  -103.0141   0.2208
```

- **CASO ALPHA = 0.75**

```
> ene.mod3 =glmnet(x[train,],y[train],alpha =0.75)
```

Calculamos el valor del parámetro lambda mediante la validación cruzada

```
> set.seed(1)
> cv.out5=cv.glmnet(x[train,],y[train],alpha=0.75)
> bestlam5=cv.out5$lambda.min
> bestlam5
  [1] 22.37354
```

Vemos cuál es el error producido por este valor de lambda y los coeficientes estimados por este método:

```
> ene.pred3=predict(ene.mod3,s=bestlam5,newx=x[test,])
> mean((ene.pred3-y.test)^2)
  [1] 100726.3
> out5=glmnet(x,y,alpha=0.75)
> ene.coef3=predict(out5,type="coefficients",
                    s=bestlam5)[1:20,]
> round(head(ene.coef3),4)
(Inter)   AtBat    Hits   HmRun    Runs     RBI
21.6095  0.0000  1.8618  0.0000  0.0000  0.0000
> sum(ene.coef3!=0)
  [1] 10
```

Vemos cuáles son los coeficientes estimados que no son nulos:

```
> round(ene.coef3[ene.coef3!=0],4)
(Inter)    Hits   Walks   CHits  CHmRun   CRuns
21.6095  1.8618  2.1883  0.0045  0.1248  0.2267

  CRBI    Leag         Div  PutOuts
0.3527  2.5751  -101.9481   0.2202
```

- **CASO ALPHA = 0.95**

```
> ene.mod4 =glmnet(x[train,],y[train],alpha =0.95)
```

Calculamos el valor del parámetro lambda mediante la validación cruzada

```
> set.seed(1)
> cv.out6=cv.glmnet(x[train,],y[train],alpha=0.95)
> bestlam6=cv.out6$lambda.min
> bestlam6
  [1] 17.66332
```

Vemos cuál es el error producido por este valor de lambda y los coeficientes estimados por este método:

```
> ene.pred4=predict(ene.mod4,s=bestlam6,newx=x[test,])
> mean((ene.pred4-y.test)^2)
  [1] 100833.1
> out6=glmnet(x,y,alpha=0.95)
> ene.coef4=predict(out6,type="coefficients",
                    s=bestlam6)[1:20,]
> round(head(ene.coef4),4)
(Inter)   AtBat    Hits   HmRun    Runs     RBI
19.8663  0.0000  1.8666  0.0000  0.0000  0.0000
> sum(ene.coef4!=0)
  [1] 8
```

Vemos cuáles son los coeficientes estimados que no son ceros:

```
> round(ene.coef4[ene.coef4!=0],4)
(Inter)    Hits   Walks   CRuns    CRBI    Leag
19.8663  1.8666  2.2211  0.2104  0.4090  1.7034

      Div  PutOuts
-103.2794   0.2206
```

- **CASO ALPHA = 0.99**

```
> ene.mod5 =glmnet(x[train,],y[train],alpha =0.99)
```

Calculamos el valor del parámetro lambda mediante la validación cruzada

```
> set.seed(1)
> cv.out7=cv.glmnet(x[train,],y[train],alpha=0.99)
> bestlam7=cv.out7$lambda.min
> bestlam7
  [1] 16.94966
```

Vemos cuál es el error producido por este valor de lambda y los coeficientes estimados por este método:

```
> ene.pred5=predict(ene.mod5,s=bestlam7,newx=x[test,])
> mean((ene.pred5-y.test)^2)
  [1] 100838.1
> out7=glmnet(x,y,alpha=0.99)
> ene.coef5=predict(out7,type="coefficients",
                    s=bestlam7)[1:20,]
> round(head(ene.coef5),4)
(Inter)   AtBat    Hits   HmRun    Runs     RBI
19.5859  0.0000  1.8695  0.0000  0.0000  0.0000
> sum(ene.coef5!=0)
  [1] 8
```

Vemos cuáles son los coeficientes estimados que no son nulos:

```
> round(ene.coef5[ene.coef5!=0],4)
(Inter)    Hits   Walks   CRuns    CRBI    Leag
19.5859  1.8695  2.2192  0.2080  0.4120  1.7482
      Div  PutOuts
-103.4599   0.2207
```

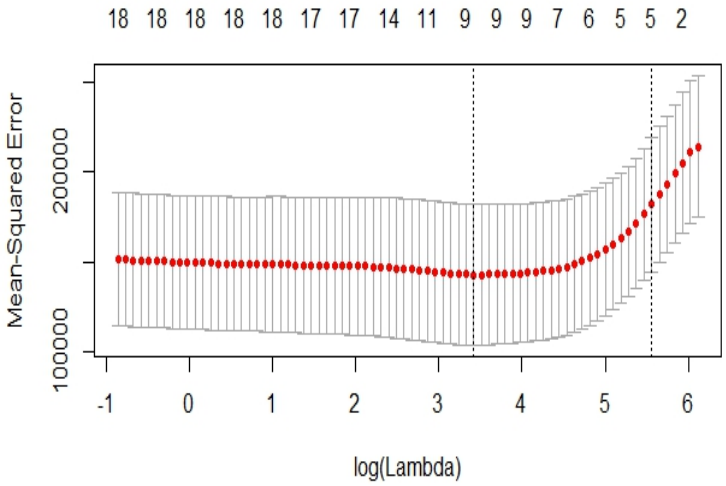
Viendo los distintos modelos que nos ofrece estos tres métodos, buscamos un criterio para quedarnos con el mejor modelo y así proceder un análisis en profundidad del mismo.
Ridge tiene muchos de sus coeficientes cercanos a cero pero al no hacer selección de variables y quedarnos con las 20 que hay en este ejemplo resulta un modelo bastante más complejo de interpretar, por lo que no se elegirá ese método. Lasso corrige ese "fallo" de Ridge eliminando 12 de las 20 variables disponibles, pero como ya se vio en teoría, este método también tiene sus limitaciones, por tanto nos quedaremos con el tercer método que es ahí donde utilizaremos un criterio, no tiene porqué ser siempre el mismo ya que el que tomamos es un criterio personal, para seleccionar nuestro modelo.
Como criterio voy a tener en cuenta el número de variables explicativas, los coeficientes estimados de cada variable, el error que se produce para cada valor del parámetro lambda (MSE, error cuadrático medio) y por último el valor de intercept, que es el coeficiente estimado β_0, ya que mientras mas bajo sea más influyen las variables en el modelo.

A continuación se crea una tabla con todos los datos que interesa según el criterio elegido:

ALPHA	LAMBDA	MSE	INTERCEPT	NÚM. PREDICTORES
0.1	87.49089	98362.68	14.8692	14
0.25	55.72473	99520.35	23.0936	12
0.5	30.57891	100322.3	19.7715	10
0.75	22.37354	100726.3	21.6095	10
0.95	17.66332	100833.1	19.8663	8
0.99	16.94966	100838.1	19.5859	8

Tomando entonces ese criterio, nos quedamos con el valor de $\alpha = 0{,}5$. El modelo asociado a $\alpha = 0{,}95$ elimina variables que parecen ser importantes, como es el número de golpes (CHits) y el número de home runs (CHmRun), durante su carrera. Estaríamos entre $\alpha = 0{,}5$ y $\alpha = 0{,}75$ que tienen el mismo número de coeficientes distintos de cero además de ser los mismas variables, y tiene el intercept mas bajo, luego el modelo depende más de las demás variables.
Una vez que ya tenemos nuestro modelo, hacemos el gráfico de la secuencia de valores de lambda y vemos que el mínimo se debe de obtener en el valor de lambda obtenido anteriormente ($\lambda = 30{,}57891$):

```
> plot(cv.out4)
```



Por la gráfica vemos que el valor de log(lambda) está entre 3.3 y 5.6 aproximadamente. Podríamos saber el valor exacto de la siguiente forma:

```
> log(bestlam4)
  [1] 3.420311
```

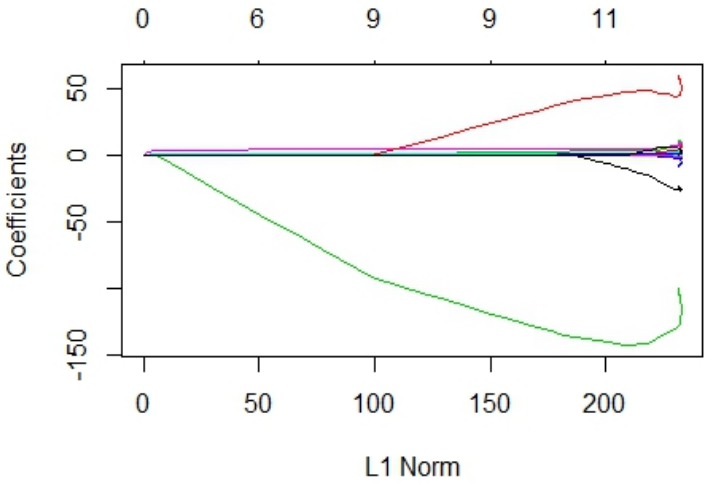
y es en ese punto sobre el eje x de la gráfica en el que se obtiene el menor error de precisión.
Este gráfico aparte de dibujar la curva de validación cruzada, dibuja la curva de la desviación estándar superior e inferior.

Utilizamos la función deviance para ver la secuencia de la desviación para los distintos valores de lambda del modelo glmnet:

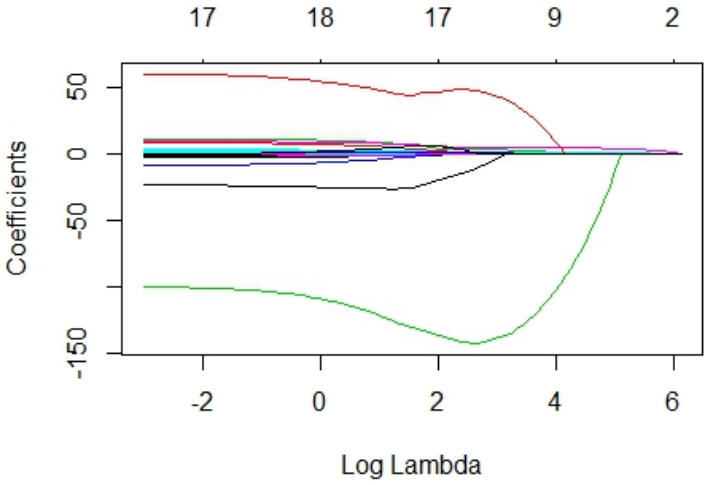
```
> deviance(out4)
   [1] 53319113 51037005 48901174 46990547 45019933 42680196
   [7] 40696291 39002710 37561090 36339292 35207188 34157476
  [13] 33269444 32518605 31762714 31070524 30484063 29987545
  [19] 29567510 29212422 28912415 28659083 28445255 28264635
  [25] 28112408 27985458 27881260 27794068 27721112 27653456
  [31] 27585776 27529123 27481554 27428280 27383728 27346352
  [37] 27315044 27288807 27161445 26931994 26651456 26409953
  [43] 26166182 25951242 25762034 25595750 25450041 25322763
  [49] 25211734 25114929 25030863 24957780 24894304 24838766
  [55] 24788222 24731469 24673908 24618650 24565361 24518456
  [61] 24477597 24441845 24410797 24383695 24360243 24339942
  [67] 24322207 24304760 24289724 24276907 24265886 24256501
  [73] 24248493 24241661 24238549 24230290 24229723 24222330
  [79] 24222056
```

Otra de las funciones de este paquete es **plot**, que realiza un gráfico de la estimación de los coeficientes en el modelo ajustado. Dibuja los coeficientes frente a la norma L1, la secuencia log(lambda) o el porcentaje de desviación explicada:

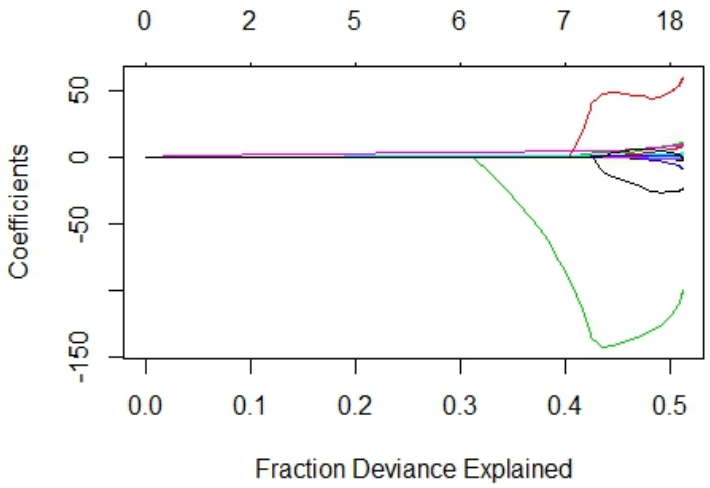
```
> plot(ene.mod2,xvar=c("norm"))
```




```
> plot(ene.mod2,xvar=c("lambda"))
```



```
> plot(ene.mod2,xvar=c("dev"))
```



Este es el porcentaje de desviación que se explica en los datos de entrenamiento.

Comparación práctica de los métodos

Pasamos a ver una comparación práctica de los métodos explicados para el valor $\lambda = 30{,}57891$:

- **Método de mínimos cuadrados**
  Recordamos que mínimos cuadrados es una simple regresion Ridge con lambda = 0.

```
> mco=predict(ridge.mod,s=0,newx=x[test,],exact=T)
> mean((mco-y.test)^2)
  [1] 115586.7
```

- **Método regresión Ridge**

```
> ridge=predict(ridge.mod,s=bestlam4,newx=x[test,])
> mean((ridge-y.test)^2)
  [1] 97072.54
```

- **Método regresión Lasso**

```
> lasso=predict(lasso.mod,s=bestlam4,newx=x[test,])
> mean((lasso-y.test)^2)
  [1] 101706.7
```

- **Método Elastic Net**

```
> ene=predict(ene.mod2,s=bestlam4,newx=x[test,])
> mean((ene-y.test)^2)
  [1] 100322.3
```

En resumen, se tienen los resultados obtenidos en la siguiente tabla:

MÉTODO	NÚM. PREDICTORES	MSE
MCO	20	115586.7
Ridge	20	97072.54
Lasso	8	101706.7
Elastic Net	10	100322.3

Podemos afirmar que regresión Ridge supera al MCO ya que el error de éste último es mayor, y además los coeficientes ridge están contraídos a cero. Entre Ridge y Lasso vemos que el error de Lasso es mayor puesto que hace selección de variables ya que llega a estimar algunos coeficientes por 0, eliminando las variables que acompaña a cada uno de esos parámetros. De esta manera obtenemos un modelo más fácil o menos complejo, con el hándicap de "perder" diversas variables predictoras y la información contenida en ellas. Dado que teníamos este problema de saber cual de los dos métodos seria en cada caso el más acertado para elegir, propusieron el método Elastic Net que retiene las ventajas tanto de Lasso como de Ridge. Basándonos en el modelo que elegimos anteriormente, vemos que tiene un error menor que Lasso, ya que satisface alguna de sus limitaciones. Lo contrario pasaría si comparamos el error de este método con el de Ridge. A pesar de que tiene mayor error nos conviene ya que en este caso tendríamos 10 variables en el modelo en vez de 20 y el error en sí no varía demasiado.

Bibliografía

[1] Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6): págs. 716–723.

[2] Schwarz, G.E. (1978). Estimating the dimension of a model. *Annals of Statistics*, 6(12): págs. 461–464.

[3] Tusell, F. (2011). *Análisis de Regresión. Introducción Teórica y Práctica basada en R.* Disponible en: http://www.et.bs.ehu.es/~etptupaf/nuevo/ficheros/estad3/nreg1.pdf

[4] Kleinbaum, D.G.; kupper, L.L. y Muller, K.F. (1988). *Applied regression analysis and other multivariate methods.* PWS-KEMT Publishing Co.

[5] Judge, G.G.; Griffiths, W.E.; Hill, R.C.; Lütkepohl, H.; Lee, T.C. (1985). *The theory and practice of econometrics.* J. Wiley and sons.

[6] Fu, W. (1998). Penalized regression: the bridge versus the lasso. *J. Computnl Graph. Statist.*, 7: págs. 397–416.

[7] Hoerl, A.E. y Kennard, R.W. (1970). Ridge regression: biased estimation for non-orthogonal problems. *Technometrics*, 12(1): págs. 55–67.

[8] Tibshirani,R. (1996). Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1): págs. 267–288.

[9] Tibshirani, R. (2011). Regression shrinkage and selection via the lasso. *A retrospective, Journal of the Royal Statistical Society: Series B (Methodological)*, 73(3): págs. 273–282.

[10] Zou, H. y Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal Royal Statistical Society: Series B*, 67(2): págs. 301–320.

[11] Segal, M.; Dahlquist, K. y Conklin, B. (2003). Regression approach for microarray data analysis. *J. Computational Biology*, 10(6): págs. 961–980.

[12] Efron, B.; Hastie, T.; Johnstone, I. y Tibshirani, R. (2004). Least angle regression. *Ann Statist*, 32(2): págs. 407–499.

[13] Friedman, J.; Hastie, T. y Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, 33(1): págs. 1–22.

[14] Games, G.; Witten, D.; Hastie, T. y Tibshirani, R. (2013). *An introduction to statistical Learning with applications in R.* Springer-Verlag, New York.