# Estimación de modelos de regresión por mínimos cuadrados penalizados



**Berta Pardos Cardiel**

Trabajo de fin de grado en Matemáticas
Universidad de Zaragoza

Director del trabajo: Jose Tomás Alcalá Nalvaiz
28 de junio de 2017

# Prólogo

El trabajo de fin de grado que se presenta a continuación lleva el título de "Estimación de modelos de regresión por mínimos cuadrados penalizados". Las técnicas de regresión por mínimos cuadrados pertenecen a la estadística básica en cuanto a predicción de modelos se refiere. Sin embargo, cuando nos encontramos con un conjunto de datos de grandes dimensiones son necesarios métodos más específicos. Uno de ellos es el caso de mínimos cuadrados penalizados, que será el objetivo principal de este trabajo.

La razón es que en el momento que se tiene un gran número de datos, pueden aparecer problemas de colinealidad entre variables o que el número de variables sea mayor al número de datos obtenidos, entre otros. Si se tienen variables que están muy relacionadas entre sí, es frecuente que el modelo estimado presente sobreajuste y que en este se incluyan todas las variables predictoras con coeficientes muy poco interpretables en la realidad. Por esto y otras razones que veremos, es importante aplicar los siguientes resultados sobre regularización.

En los últimos años, se ha estado dando cada vez más importancia al fenómeno conocido como Big Data. Este concepto hace referencia a conjuntos de datos tan grandes, que aplicaciones informáticas tradicionales del procesamiento de datos no son suficientes para tratar con ellos, así como a los procedimientos para encontrar patrones repetitivos dentro de esos datos. Hoy en día está presente en casi cualquier ámbito de la vida o empresa.

Es innegable que el avance de las tecnologías y de Internet, a través del cual todo el mundo puede comunicarse y compartir información, ha influido en este cambio. La cantidad de información ha ido aumentado exponencialmente, y los métodos estadísticos clásicos habían sido pensados para ser utilizados con cantidades relativamente pequeñas de datos.

# Summary

This project focuses on regularized least-squares linear regression and more specifically on Tikhonov regularization. We will see this topic from a theorical point of view and in the last chapter we will implement these mathematical results over two data sets by using the programming language R.

The document starts with a brief review about least-squares linear regression and some problems that will cause the introduction of the regularization term. The linear regression aim is to find good values of some parameters in order to have a model that is able to predict accurately the response over a new data set.

A common rule to do that is using least-squares through which we obtain the solution for the parameters $\hat{\beta} = (\mathbb{X}^T\mathbb{X})^{-1}\mathbb{X}^T\mathbb{Y}$ by minimizing the residual sum of squares $RSS(\beta) = \|\mathbb{Y} - \mathbb{X}\beta\|_2^2$. As we can see it is required to invert the matrix $\mathbb{X}^T\mathbb{X}$, however, it might be impossible because of being an ill-conditioned matrix. This happens when the columns of $\mathbb{X}$ are highly correlated, when the number of parameters is large or even because of an overfitting. These problems can be solved by using a variable selection, calculating the pseudoinverse or introducing a regularization function. We will develop the last solution where the new minimizing term become $RSS(\beta) + P_\lambda(\beta)$. Penalized or regularized methods always come with at least one regularization parameter, here $\lambda$, that controls the tradeoff between likelihood and penalty.

Now, let's see the most important methods: Ridge Regression uses the norm L2 and it enables fitted parameters to have less variance, LASSO uses the norm L1 and solutions become sparse, Elastic Net is a linear combination between the previous ones, LARS, nonparametric regression, classification, etc. Obviously, these methods depend heavely on the regularization parameter and we should select it. The most widely way is based on how well predictions do at predicting new instances of the response variable $\mathbb{Y}$, and $K-$fold Cross Validation will be the chosen method in this project. It consists in splitting the data into $K$ folds, fitting the model on $K-1$ of the folds and evaluating risk on the fold that was left out. It is worth stressing the values $K = 10$ and $K = n$ (known as Leave One Out CV, LOOCV).

Then in the second chapter, the Tikhonov regularization development will take place. Firstly, it is important to show that many linear models for regression can be reformulated in terms of kernel functions, what is known as the dual representation. If we consider the L2 penalized regression model, it is possible to express the least-squares loss function solution with kernel functions, through a variable change and the Gram matrix $\mathbf{K}$ definition. This kind of solution will let us work with high dimension spaces, even infinity.

The next section contains several theorical results about Hilbert spaces, reproducing kernel Hilbert spaces, the Representer Theorem, etc. This theory will be useful to obtain the general solution for the Tikhonov regularization and the most general way of showing this problem is through the next equation $min_{f\in H}\frac{1}{n}\sum_{i=1}^{n} V(f(x_i), y_i) + \lambda\|f\|_H^2$. The optimal solution can be expressed as $f_\lambda(x) = \mathbf{K}^T c$ being $c \in \mathbb{R}^n$, so minimizing over a Hilbert space is comparable to minimizing over $\mathbb{R}^n$.

Taking the loss function as $V(f(x_i), y_i) = (y_i - f(x_i))^2$, we are in the regularized least-squares case. Thanks to the Representer Theorem, the expression we want to minimize become $\frac{1}{2}\|\mathbb{Y} - \mathbf{K}c\|_2^2 + \frac{\lambda}{2}c^T\mathbf{K}c$ and the corresponding solution is $c = (\mathbf{K} + \lambda I)^{-1}\mathbb{Y}$. If we want to predict the response for new values, the formula obtained is $\hat{y}_{new} = \mathbf{k}(x_{new})^T c$ so we need to calculate $c$. A quick method is the eigendecomposition of the matrix $\mathbf{K}$, then calculating $c(\lambda)$ has basically no additional cost because when varying $\lambda$ just changes the diagonal matrix. However it is not enough.

As a consecuence, we need a device to find good values for the regularization parameter, so it is the

moment to use Cross Validation. In the case of regularized least-squares, the best way to obtain the error is by making use of LOOCV. If we define $f_{S^i}$ as the $i-$value of LOO for RLS, we can deduce that $f_{S^i}$ is the optimal solution for RLS by fitting the model over the set without the $i-$data and testing in $x_i$. Let $L_V$ and $L_E$ be the LOO values and errors over the data set, it is also shown that working with them is computational effective.

There exists another way to calculate the LOO values in order to validate the parameter $\lambda$ in the linear kernel case. It consists of getting the singular values decomposition (SVD) of the design matrix $\mathbb{X}$. After some algebraic manipulations, we obtain the same expression for the error $L_E$ as in the previous section. Moreover, the truncated SVD is important when there are some singular values much larger than the others, so we forget the last ones before performing the decomposition.

Finally, the most general Tikhonov regularization expression will be explained. If we consider the cuadratic norm of a vector $x$ as $\|x\|_P = x^T P x$, being $P$ a positive definite matrix, the loss function turns out to be $J(\beta) = \|\mathbb{Y} - \mathbb{X}\beta\|_P + \lambda\|\beta - \bar{\beta}\|_Q$. Taking $P = I = Q$ and $\bar{\beta} = 0$, we obtain the solution used along all this project.

In addition, we have included two recent optimization methods. The first one is an alternative to LASSO and there exist lots of approaches for the parameters' estimation able to solve the non-differential problem of the objective function. It is done by transforming the problem into a constrained one, defined by a convex set. Quadratic programming is a remarkable method. The second one is the alternating direction method. It is useful for constrained problems as well, like regularized problems with L1 and L2 norm. An important application is the image restoration.

In the third chapter, we will put these results in practice over two different data sets. The first one is called "Prostate" and it is a simple set with a few variables. It will be useful in order to show how penalized methods work and compare the results in a real data set. It is shown how coefficients tend to zero at the time the regularization parameter increases. It is also important to remark the difference between model coefficients estimated by all methods, because LASSO will be a great choice instead of OLS, in order to have a sparing model. Another result that we will compare is cross validation for different values of $K$. Some graphics of the CV errors will represent the behaviour of these techniques taking $K = 10$ and $K = 97$.

On the other hand, the second data set is called "Ames Housing" and it contains a large number of variables and data (more similar to a Big Data problem). It will present some collinearity problems because of the amount of variables, so techniques developed in this project will be the solution. We will show that the linear models obtained from ridge and LASSO are a good alternative (in terms of $R^2$) to avoid the OLS problems where coefficients have not an easy interpretation. Furthermore, the implementation of non-linear behaviour as it is kernel regularized least squares, will get better the variability explained almost like OLS one. This improvement will be more important in this type of big models in order to select just the representative variables for modeling the response.

# Índice general

# Capítulo 1

# Regresión lineal penalizada

## 1.1. Introducción

Las técnicas de regresión son modelos matemáticos que tratan de encontrar relaciones funcionales entre variables, en el caso de regresión lineal con funciones lineales. Están involucrados $n$ datos en un proceso descrito por $p+1$ variables, $p$ explicativas (o covariables) $\{x_1,...,x_p\}$ y la variable respuesta que queremos explicar $y$. El objetivo es obtener una combinación lineal, mediante $p+1$ parámetros, de las $p$ variables que mejor predicen la respuesta. El problema puede formularse de la siguiente forma:

$$y_i = \beta_0 + \sum_{j=1}^{p} x_{ji}\beta_j + \varepsilon_i \qquad i = 1,...,n \tag{1.1}$$

siendo $\beta_j$ los parámetros y $\varepsilon_i$ los términos de error para cada $i$, los cuales asumimos ser independientes entre sí y con distribución normal de media 0 e igual desviación típica.

El método de estimación de los parámetros más utilizado es el de mínimos cuadrados. En este caso la función de pérdida (*loss function*) que se desea minimizar es la suma residual de errores cuadráticos

$$RSS = \sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_{1i} - \cdots - \beta_p x_{pi})^2 \tag{1.2}$$

Normalmente se considera la matriz del diseño $\mathbb{X}$ con el fin de manejar una notación más compacta.

$$\mathbb{X} = \begin{bmatrix} 1 & x_{11} & x_{21} & \cdots & x_{p1} \\ 1 & x_{12} & x_{22} & \cdots & x_{p2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & x_{1n} & x_{2n} & \cdots & x_{pn} \end{bmatrix}$$

De la misma manera consideramos los vectores columna $\mathbb{Y}$, formado por los $n$ datos de la variable respuesta y $\beta$ por los $p+1$ parámetros $\beta_i$ incluyendo $\beta_0$, coeficiente del término independiente. Así podemos expresar (1,2) de forma matricial como $\|\mathbb{Y} - \mathbb{X}\beta\|_2^2$ y por tanto la estimación será el vector $\beta$ que la minimiza. Mediante un proceso de derivación respecto a estos parámetros e igualando a 0 se obtiene la expresión

$$\hat{\beta} = (\mathbb{X}^T\mathbb{X})^{-1}\mathbb{X}^T\mathbb{Y} \tag{1.3}$$

que da la solución a este problema de estimación. Un buen libro en el que se explica con más detalle es [12].

Sin embargo, si los datos cumplen una serie de características podríamos tener una matriz $\mathbb{X}^T\mathbb{X}$ mal condicionada y por tanto no ser posible invertirla. Veamos alguna de ellas:

- *Cuasicolinealidad de las variables*: cuando las columnas de $\mathbb{X}$ están muy correladas, por ejemplo si se están analizando los nutrientes de un determinado alimento habría colinealidad entre grasa, calorías, etc.

- *Sobreajuste*: al obtener un $R^2$ artificialmente muy alto, es decir, muy próximo a 1. Esto ocurre cuando se obtiene una función que ajusta perfectamente cada dato pero es tan poco parsimoniosa que resulta difícil de manejar y no refleja la realidad.

- *Número de variables mucho mayor que de datos ( $p \gg n$ )*: se suele dar en el ámbito médico a la hora de tomar datos de pacientes que tienen una "enfermedad rara". Se tiene un número elevado de variables (genes) en comparación a la cantidad de datos a analizar (pacientes).

Hay varias alternativas a la hora de solventar estos problemas aunque nosotros nos centraremos en la última a lo largo de este trabajo. Véase [15] y [2] para más detalle.

- *Selección de variables*: se elige el conjunto de variables que se consideran más relevantes para predecir y para ello hay varios métodos. Los más conocidos son: *backward elimination* (partiendo de todas las variables se selecciona la menos relevante y se elimina del modelo), *forward selection* (se selecciona la variable más significante y se van añadiendo al modelo una a una) y *stepwise selection* (combina las dos anteriores). Se necesitan criterios para seleccionar el mejor modelo entre todos los posibles. Algunos son: Akaike Information Criterion (AIC), Bayes Information Criterion (BIC), $C_p$ de Mallow, $R^2$ ajustado, etc.

- *Cálculo de la pseudoinversa (inversa generalizada o de Moore-Penrose)*: dada una matriz $A \in \mathbb{R}^{mxn}$ en un sistema de ecuaciones lineales $Ax = b$, la pseudoinversa $A^\dagger$ representa la solución $x$ que minimiza $\|Ax-b\|$. Si $A$ tiene rango máximo $n$ entonces se tiene la expresión $A^\dagger = (A^TA)^{-1}A^T$ y en el caso de que $A$ sea cuadrada y no singular, $A^\dagger = A^{-1}$.

- *Regularización o penalización*: en lugar de minimizar $RSS(\beta)$ dado en (1,2), lo hacemos con $RSS(\beta) + P_\lambda(\beta)$ siendo $P_\lambda$ una función que penaliza los valores de los parámetros desconocidos que se pueden considerar menos realistas. Estos métodos siempre involucran un parámetro, en este caso $\lambda > 0$, que controla simultáneamente el grado del ajuste y la penalización.

## 1.2. Penalización con norma L2

En este tipo de regresión penalizada, también conocida como *regresión contraída (ridge regression)*, se quiere minimizar $\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_{1i} - \cdots - \beta_p x_{pi})^2 + \lambda \sum_{j=0}^{p} \beta_j^2$, aplicando la norma L2 en el término de penalización. De forma matricial se expresa como

$$\|\mathbb{Y} - \mathbb{X}\beta\|_2^2 + \lambda \|\beta\|_2^2 \tag{1.4}$$

Siendo la matriz $\mathbb{X}^T\mathbb{X} + \lambda I$ no singular para cualquier $\mathbb{X}$, el vector de parámetros que minimiza sería

$$\hat{\beta}_{L_2} = (\mathbb{X}^T\mathbb{X} + \lambda I)^{-1}\mathbb{X}^T\mathbb{Y} \tag{1.5}$$

En el caso particular de que la matriz del diseño sea ortonormal, la solución es $\hat{\beta}_{L_2} = \frac{\hat{\beta}}{\lambda+1}$.

Aplicando esta penalización conseguimos que cualquier combinación lineal de los elementos de $\hat{\beta}_{L_2}$ tenga menor varianza que la de los elementos de $\hat{\beta}$. Como puede observarse en las siguientes expresiones, en este caso el sesgo es mayor y la diferencia de varianzas resulta ser una matriz semidefinida negativa. Un artículo que entra en detalle sobre esto es [10].

$$Bias(\hat{\beta}_{L_2}) = -\lambda(\mathbb{X}^T\mathbb{X} + \lambda I)^{-1}\beta$$

$$Var(\hat{\beta}_{L_2}) - Var(\hat{\beta}) = \sigma^2(\mathbb{X}^T\mathbb{X}+\lambda I)^{-1}\mathbb{X}^T\mathbb{X}(\mathbb{X}^T\mathbb{X}+\lambda I)^{-1} - \sigma^2(\mathbb{X}^T\mathbb{X}^{-1})$$

Por otro lado, la expresión dada en (1,5) se puede interpretar como un estimador Bayes de los parámetros de un modelo de regresión con distribución a priori sobre $\beta$ de tipo $N(\mathbb{O}|\sigma^2_\beta I_{p+1})$. Se puede deducir que la log-densidad a posterior es $logf(\beta|\mathbb{Y},\mathbb{X}) \propto -\frac{1}{2\sigma^2_\varepsilon}\sum_{i=1}^n(y_i-\beta^T x_i)^2 - \frac{1}{2\sigma^2_\beta}\beta^T\beta$, luego maximizar esta densidad a posteriori sería equivalente a minimizar (1,4) con $\lambda = \frac{\sigma^2_\varepsilon}{\sigma^2_\beta}$.

## 1.3. Penalización con norma L1

Consideramos ahora la siguiente expresión de mínimos cuadrados penalizada con la norma L1:

$$\frac{1}{2}RSS(\beta) + \lambda\sum_{j=0}^{p}|\beta_j| \tag{1.6}$$

El cambio con respecto a L2 es sutil, sin embargo, tiene un gran impacto en el resultado de la estimación. Como se puede consultar en [15], a diferencia de la anterior esta penalización mejora hacia un modelo más parsimonioso y una mejor interpretabilidad de los valores de los coeficientes. Además las soluciones pasan a ser *sparse*, es decir, muchos valores de los parámetros son nulos.

La solución resultante para los estimadores se conoce como *LASSO* (Least Absolute Shrinkage and Selection Operator) y no puede expresarse de forma explícita salvo en el caso de una matriz del diseño ortonormal. Esto se debe a la presencia del valor absoluto, haciendo que la función objetivo que se desea minimizar sea no diferenciable.

Por otro lado, esta expresión recuerda a un problema de optimización convexa sin restricciones en términos de $\beta$ para el que se han buscado multitud de técnicas con el fin de resolverlo. Algunos de estos algoritmos se verán en el Capítulo 2.

**Nota** : es frecuente trabajar con los datos tipificados en media y por tanto dejar fuera de la penalización al parámetro $\beta_0$ que coincidirá con la media del vector $\mathbb{Y}$.

## 1.4. Otras penalizaciones

Cabe destacar otros métodos de penalización aunque en este trabajo nos centraremos en los anteriores. Véase [15] y [18] para más detalle.

- **Red elástica ("Elastic Net")**: se trata de un método de regresión penalizada que combina linealmente las dos anteriores. La función que se quiere minimizar ahora depende de dos parámetros $\lambda \geq 0$ y $0 \leq \alpha \leq 1$

  $$\|\mathbb{Y}-\mathbb{X}\beta\|_2^2 + \lambda(\alpha\|\beta\|_1 + (1-\alpha)\|\beta\|_2^2)$$

  Si $\alpha = 1$ se tiene la penalización *LASSO* mientras que con $\alpha = 0$, penalización contraída.

  Aunque el proceso de validación de los parámetros es más lento ya que hay que considerar todas las combinaciones de ambos, el método de red elástica es más flexible y con él se pueden obtener soluciones únicas incluso cuando el número de parámetros a estimar es mucho mayor que el número de datos ($p \gg n$).

- **LARS** (Least Angle Regression): es un algoritmo iterativo rápido cuyo procedimiento es parecido al de *LASSO* y puede aplicarse a este para encontrar su solución. Veamos el algoritmo:

  1. Normalizar los datos con media 0 y desviación típica 1.
  2. Inicializar $\beta_1, ..., \beta_p = 0$.
  3. Calcular el error residual del modelo $r = \mathbb{Y} - \mathbb{X}\beta$.

4. Encontrar la variable $x_j$ que esté más correlada con $r$, es decir, a la que corresponda el mayor $|< x_j, r >|$.
5. Se hace un salto en la dirección de la variable seleccionada hasta que entre otra variable. Ahora en lugar de seguir la dirección de la variable $x_k$, se sigue una dirección equiangular entre ambas variables $x_j$ y $x_k$ lo que hace que la correlación sea igual al residuo actual.
6. Se sigue el proceso hasta que entren todas la variables.

- **Regresión no paramétrica**: otra idea de la regresión penalizada es la estimación de curvas de regresión más generales en las que se penalizan funciones excesivamente "onduladas". Ahora el término de penalización se define por una función $f$ cualquiera:

$$P_\lambda(f) = \lambda \int (f''(x))^2 dx$$

- **Clasificación**: se aplica cuando las covariables están organizadas en grupos. En este caso la penalización se presenta de forma jerárquica mediante la siguiente función, siendo $P_O$ el nivel de penalización de un grupo y $P_I$ el nivel individual:

$$P_\lambda(\beta) = \sum_{i=1}^{p} P_{O,\lambda_1}\{\sum_{j=1}^{J_i} P_{I,\lambda_2}(\beta_{ij})\}$$

## 1.5. Selección del parámetro $\lambda$

En cualquier método de regresión penalizada es importante la selección del parámetro $\lambda$. Hay varias formas aunque la más usada consiste en medir la capacidad de predicción de nuevos datos de la variable respuesta $\mathbb{Y}$, con respecto a predicciones basadas en los parámetros $\hat{\beta}_\lambda$. Sin embargo, si se usa el conjunto de datos completo para ajustar el modelo y a continuación para estimar su capacidad predictiva, nos encontraríamos con un problema de sobreajuste.

Una posible solución a este problema sería dividir el conjunto de datos en dos partes. Con una de ellas se ajustarían los parámetros, y con la otra se evaluaría la capacidad de predicción del modelo en este segundo subconjunto de datos. Sin embargo, aplicarlo a conjuntos con pocos datos no sería útil, ya que obtendríamos resultados poco realistas. Así, la *validación cruzada (CV)* con $K$ bloques da respuesta a este problema. Siendo $V$ el conjunto de datos, el algoritmo es:

---

1 function CVKfold $(V, f, \theta)$ ;
**Input :** $V$: conjunto de datos; $f(x;\theta)$: familia de modelos parametrizados por $\theta$
**Output:** $Err_{CV}$: Error de predicción asociado a $f$ estimado por CV
2 Particionar el conjunto de datos $V = V_1 \bigcup \cdots \bigcup V_K$ ;
3 **for** $i = 1$ **to** $K$ **do**
4 Ajustar el modelo con la muestra $V \backslash V_i$ ;
5 obtener $\hat{\theta}_{(i)}$ y $\hat{f}_{(i)} = f(x;\hat{\theta}_{(i)})$ ;
6 Calcular el error en la muestra $V_i$, $Err_{(i)}$
7 **end**
8 Calcular la estimación del error haciendo la media de los $K$ resultados, ;
9 $Err_{CV} = K^{-1}\sum_{i=1}^{K} Err_{(i)}$ .

---

Los valores de $K$ más utilizados son $K = 5, 10$. Un caso especial es *leave-one-out cross validation*, en el que $K = n$.

# Capítulo 2

# Regularización de Tikhonov

Hemos visto los elementos fundamentales de la regresión penalizada. En este capítulo vamos a desarrollar una formulación general de este tópico que la relaciona con el método de regularización de Tikhonov.

En el primer cuarto del siglo XX, Hadamard introdujo los conceptos de problema "bien planteado" - "mal planteado". Si un problema está mal planteado significa que no tiene solución o que las soluciones no son únicas, luego ofrece dificultades numéricas. Para abordar este problema, en 1963, el matemático ruso Andrei N. Tikhonov fundamentó y formuló por primera vez un "método de regularización". La idea principal es restringir la solución a un conjunto más pequeño, introduciendo la penalización como restricción.

## 2.1. Representación dual y funciones kernel

En regresión lineal, muchos modelos pueden reformularse mediante una representación dual. Si consideramos el modelo de regresión lineal penalizado con norma L2, la función de pérdida dada por $(1{,}4)$ también se puede denotar como

$$J(\beta) = \frac{1}{2}(\mathbb{X}\beta - \mathbb{Y})^T(\mathbb{X}\beta - \mathbb{Y}) + \frac{\lambda}{2}\beta^T\beta \tag{2.1}$$

Con el objetivo de minimizar la función $J(\beta)$, igualamos a cero el gradiente con respecto a $\beta$. Si introducimos el vector $\mathbf{a} = (a_1, ..., a_n)^T$ formado por los elementos

$$a_i = \frac{-1}{\lambda}(\hat{\beta}^T x_i - y_i),$$

$\hat{\beta}$ se puede expresar como combinación lineal de los vectores $x_i$ de la siguiente forma:

$$\hat{\beta} = \frac{-1}{\lambda}\sum_{i=1}^{n}(\hat{\beta}^T x_i - y_i)x_i = \sum_{i=1}^{n} a_i x_i = \mathbb{X}^T\mathbf{a}$$

Aplicando la nueva notación a $(2{,}1)$ obtenemos la expresión mediante representación dual.

$$\begin{aligned} J(\mathbf{a}) &= \frac{1}{2}(\mathbb{X}\mathbb{X}^T\mathbf{a} - \mathbb{Y})^T(\mathbb{X}\mathbb{X}^T\mathbf{a} - \mathbb{Y}) + \frac{\lambda}{2}\mathbf{a}^T\mathbb{X}\mathbb{X}^T\mathbf{a} = \\ &= \frac{1}{2}(\mathbf{a}^T\mathbb{X}\mathbb{X}^T\mathbb{X}\mathbb{X}^T\mathbf{a} - 2\mathbf{a}^T\mathbb{X}\mathbb{X}^T\mathbb{Y} + \mathbb{Y}^T\mathbb{Y}) + \frac{\lambda}{2}\mathbf{a}^T\mathbb{X}\mathbb{X}^T\mathbf{a} \end{aligned}$$

Definimos ahora la Matriz de Gram $\mathbf{K} = \mathbb{X}\mathbb{X}^T$ que es una matriz $n \times n$ simétrica con elementos $\mathbf{K}_{ij} = x_i^T x_j = < x_i, x_j >$. En términos de esta matriz, la expresión de la función del error por mínimos cuadrados queda

$$J(\mathbf{a}) = \frac{1}{2}(\mathbf{a}^T\mathbf{K}\mathbf{K}\mathbf{a} - 2\mathbf{a}^T\mathbf{K}\mathbb{Y} + \mathbb{Y}^T\mathbb{Y}) + \frac{\lambda}{2}\mathbf{a}^T\mathbf{K}\mathbf{a} \tag{2.2}$$

Procediendo de la misma forma sobre (2,2), obtenemos la siguiente solución para **a**:

$$\mathbf{a} = (\mathbf{K} + \lambda I)^{-1}\mathbb{Y}$$

A continuación se hace necesario dar una expresión para la predicción de nuevos datos usando la representación dual. Dado un nuevo dato $x_{new}$, si definimos el vector $\mathbf{k}(x) = \mathbb{X}^T x$ , la estimación de la variable respuesta vendrá dada por

$$\hat{y}_{new} = \hat{\beta}^T x_{new} = \mathbf{a}^T\mathbb{X}^T x_{new} = \mathbf{a}^T\mathbf{k}(x_{new}) = \mathbf{k}(x_{new})^T\mathbf{a} = \mathbf{k}(x_{new})^T(\mathbf{K} + \lambda I)^{-1}\mathbb{Y} \tag{2.3}$$

Como se puede ver, la representación dual permite que la solución del problema de mínimos cuadrados esté totalmente expresada en términos de productos escalares. Si cambiamos $< x_i, x_j >$ por $< \phi(x_i), \phi(x_j) >$, se observa que el cambio afecta a la matriz de Gram **K** donde sus elementos serían

$$\mathbf{K}_{ij} =< \phi(x_i), \phi(x_j) >$$

y el vector $\mathbf{k}(x) = (k(x_1, x), ..., k(x_n, x))$ siendo $k(x_i, x_j) =< \phi(x_i), \phi(x_j) >$ la función kernel para una $\phi$ cualquiera. El concepto de kernelización puede considerarse como un mecanismo para generalizar los modelos lineales a espacios de características no lineales.

Veamos los ejemplos más comunes de funciones kernel, siendo $d$ y $\sigma$ valores prefijados:

- lineal: $k(x_i, x_j) = x_i^T x_j$
- polinomial: $k(x_i, x_j) = (x_i^T x_j + 1)^d$
- gausiano: $k(x_i, x_j) = exp(\frac{\|x_i - x_j\|^2}{\sigma^2})$

De esta forma se determina el vector de parámetros **a** invirtiendo una matriz $n \times n$, mientras que en la formulación original se tiene que invertir una matriz $(p+1) \times (p+1)$ para determinar $\hat{\beta}$. A pesar de que habitualmente $n \gg p+1$, con las funciones kernel podemos pasar a espacios de dimensión alta e incluso infinita. Una condición necesaria y suficiente para construir funciones kernel válidas es que la Matriz de Gram sea semidefinida positiva para todos los posibles valores $\{x_i\}$. En [1] se pueden consultar estos resultados con más detalle.

## 2.2. Espacios de Hilbert con núcleo reproductor (RKHS)

Vamos a introducir una familia de espacios conocidos como Espacios de Hilbert con núcleo reproductor. Sus comienzos se remontan a la teoría del aprendizaje estadístico (Learning Theory) de Girosi en 1997. En 1989, Girosi y Poggio introdujeron la regularización de Tikhonov en esta teoría y trabajaron con RKHS implícitamente. Se basaron principalmente en espacios de hipótesis sobre dominios no acotados, se puede consultar en [7] y [4] para más detalle. Sin embargo, los RKHS ya se tuvieron en cuenta en teoría de aproximación en 1990 con los trabajos de Wahba, ver [16], y para informática con Torre, Poggio, etc.

Cada espacio de Hilbert está asociado a un tipo de kernel particular y serán útiles para deducir la solución general del problema de regularización de Tikhonov. Esencialmente, un espacio de Hilbert nos permitirá aplicar conceptos de álgebra lineal de dimensión finita a espacios de funciones de dimensión infinita. Veamos a continuación una serie de resultados teóricos de los que se pueden encontrar las demostraciones en libros como [11].

**Definición 1.** Un *espacio de Hilbert H* es un espacio lineal completo que puede tener dimensión infinita y tiene un producto escalar.

Una norma en $H$ puede definirse con el producto escalar como $\|\cdot\|_H = \sqrt{<\cdot,\cdot>}$. Además asumimos que $H$ es separable (contiene un subconjunto denso numerable), luego $H$ admite una base ortonormal numerable.

**Definición 2.** Un *funcional de evaluación* sobre un espacio de Hilbert de funciones $H$ es un funcional lineal $\mathscr{F}_t : H \longrightarrow \mathbb{R}$ que evalúa cada función del espacio en el punto $t$, es decir, $\mathscr{F}_t[f] = f(t)$ para todo $f \in H$.

**Definición 3.** Un espacio de Hilbert $H$ es un *RKHS* si los funcionales de evaluación están acotados, es decir, si para todo $t$ existe algún $M > 0$ tal que para todo $f \in H$

$$|\mathscr{F}_t[f]| = |f(t)| \leqslant M\|f\|_H$$

Esta condición es bastante general y la más débil posible que nos asegura la existencia de un producto escalar y la posibilidad de evaluar cada función del espacio en todos los puntos del dominio. Una noción más útil en la práctica es la de *kernel o núcleo reproductor* del que RKHS toma su nombre.

**Teorema 2.1.** *Si $H$ es un RKHS entonces para cada $t \in X$ existe una función $K_t \in H$ (llamada el representante de $t$) con la propiedad de reproducción para todo $f \in H$*

$$\mathscr{F}_t[f] = < K_t, f >_H = f(t)$$

Este teorema nos permite representar el funcional de evaluación tomando el producto interior con un elemento de $H$. Como $K_t$ es una función en $H$, por la propiedad de reproducción, para cada $x \in X$ podemos escribir $K_t(x) = < K_t, K_x >_H$. Consideramos esto como la definición de núcleo reproductor.

**Definición 4.** El *núcleo reproductor de $H$* es una función $K : X \times X \longrightarrow \mathbb{R}$ definida por $K(t,x) := K_t(x)$.

En general, tenemos la siguiente definición de núcleo reproductor.

**Definición 5.** Sea $X$ un conjunto, un subconjunto de $\mathbb{R}^n$ o él mismo. Una función $K : X \times X \longrightarrow \mathbb{R}$ es un núcleo reproductor si es simétrica, es decir, $K(x,y) = K(y,x)$ y definida positiva:

$$\sum_{i,j=1}^{n} c_i c_j K(t_i, t_j) \geqslant 0$$

para cualquier $n \in \mathbb{N}$ y unos $t_1, ..., t_n \in X$ y $c_1, ..., c_n \in \mathbb{R}$ dados.

El siguiente teorema establece la relación entre RKHS y núcleo reproductor.

**Teorema 2.2.** *Un RKHS define un núcleo reproductor correspondiente. Recíprocamente, un núcleo reproductor define un único RKHS.*

Ahora que ya tenemos la teoría necesaria de RKHS podemos ver la regularización de Tikhonov de una forma más general. Los algoritmos que queremos estudiar están definidos mediante un problema de optimización sobre RKHS:

$$f_\lambda = argmin_{f \in H} \frac{1}{n} \sum_{i=1}^{n} V(f(x_i), y_i) + \lambda \|f\|_H^2 \tag{2.4}$$

siendo $\lambda > 0$ el parámetro de regularización, $H$ el RKHS definido por el núcleo reproductor $K(\cdot,\cdot)$, $V(\cdot,\cdot)$ la función de pérdida y $\{x_i, y_i\}$ los datos para $i = 1, ..., n$.

Hemos impuesto estabilidad en el problema con el uso de la regularización, sin embargo todavía es necesario comprobar si siempre existe solución y si es única. Esta condición se exige en la función de pérdida. Si $V(\cdot,\cdot)$ es convexa respecto al primer argumento, la función objetivo

$$\Phi[f] = \frac{1}{n} \sum_{i=1}^{n} V(f(x_i), y_i) + \lambda \|f\|_H^2$$

es estrictamente convexa y coerciva, es decir, crece rápidamente en los extremos. En consecuencia tendrá un único mínimo local, luego global.

Gracias al siguiente Teorema de Representación, la solución dada por (2,4) puede representarse de una forma más compacta a pesar de que $H$ sea un espacio de funciones con dimensión infinita. Por lo tanto, minimizar sobre un espacio de Hilbert se reduce a minimizar sobre $\mathbb{R}^n$.

**Teorema 2.3.** *La solución $f_\lambda$ sobre el RKHS H del problema de regularización de Tikhonov* (2,4) *puede expresarse como*

$$f_\lambda(x) = \sum_{i=1}^{n} c_i K(x_i, x) \tag{2.5}$$

*para una n-tupla $(c_1, ..., c_n) \in \mathbb{R}^n$ concreta.*

**Dem** : Definimos el subespacio lineal de $H$ generado por los representantes del conjunto de datos

$$H_0 = \{f \in H | f = \sum_{i=1}^{n} \alpha_i K_{x_i}\}$$

Sea $H_0^\perp$ el subespacio lineal de $H$ ortogonal a $H_0$, es decir, $H_0^\perp = \{g \in H | < g, f >= 0, \forall\, f \in H_0\}$.

Podemos escribir $H = H_0 \oplus H_0^\perp$ ya que $H_0$ es de dimensión finita, luego cerrado.

Veamos ahora que cualquier $f \in H$ puede descomponerse de forma única en una componente de $H_0$, denotada por $f_0$ y una componente perpendicular a $H_0$, denotada por $f_0^\perp$. Así pues $f = f_0 + f_0^\perp$.

Por ortogonalidad se tiene que

$$\|f\|_H^2 =< f, f >_H =< f_0 + f_0^\perp, f_0 + f_0^\perp >_H = \|f_0 + f_0^\perp\|_H^2 = \|f_0\|_H^2 + \|f_0^\perp\|_H^2$$

Si definimos $I_S[f] = \frac{1}{n}\sum_{i=1}^{n} V(f(x_i), y_i)$, se cumple que

$$I_S[f] = I_S[f_0] + I_S[f_0^\perp] = I_S[f_0]$$

Veámoslo:

$$I_S[f_0 + f_0^\perp] = \frac{1}{n}\sum_{i=1}^{n} V(f_0(x_i) + f_0^\perp(x_i), y_i)$$

Por la propiedad reproductora, $f_0^\perp(x_i) =< f_0^\perp, K_{x_i} >$ que se anula por la definición del subespacio $H_0^\perp$.

Combinando estas dos resultados se observa que

$$I_S[f_0 + f_0^\perp] + \lambda\|f_0 + f_0^\perp\|_H^2 = I_S[f_0] + \lambda\|f_0\|_H^2 + \lambda\|f_0^\perp\|_H^2 \geqslant I_S[f_0] + \lambda\|f_0\|_H^2$$

En consecuencia, el mínimo $f_\lambda$ se alcanza para $f_0$ que debe pertenecer al espacio lineal $H_0$.

## 2.3. Mínimos cuadrados penalizados para una $f$ cualquiera

Cuando la función de perdida sea $V(f(x_i), y_i) = (y_i - f(x_i))^2$ estaremos en el caso de mínimos cuadrados penalizados. Siendo $f$ una función cualquiera del espacio de Hilbert con núcleo reproductor $H$, la expresión (2,4) queda

$$argmin_{f \in H} \frac{1}{2}\sum_{i=1}^{n} (y_i - f(x_i))^2 + \frac{\lambda}{2}\|f\|_H^2$$

Usando el Teorema de Representación (2,5) podemos escribir la solución del problema como

$$f(x_i) = \sum_{j=1}^{n} c_j k(x_i, x_j) = \mathbf{K}_{(i\cdot)}^T c$$

siendo $c \in \mathbb{R}^n$ y $\mathbf{K}_{(i\cdot)}$ la $i$-ésima fila de la matriz de Gram. Sustituyendo $f$ por la nueva expresión podemos reescribir el problema

$$argmin_{c\in\mathbb{R}^n} \frac{1}{2}\|\mathbb{Y} - \mathbf{K}c\|_2^2 + \frac{\lambda}{2}\|f\|_H^2$$

Aplicando de nuevo el Teorema de Representación y la propiedad reproductora al término de penalización obtenemos

$$\|f\|_H^2 = < f, f >_H = < \sum_{i=1}^{n} c_i k(\cdot, x_i), \sum_{j=1}^{n} c_j k(\cdot, x_j) >_H =$$

$$= \sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j < k(\cdot, x_i), k(\cdot, x_j) > = \sum_{i=1}^{n}\sum_{j=1}^{n} c_i c_j k(x_i, x_j) = c^T \mathbf{K} c$$

Finalmente tenemos la siguiente expresión que es una función convexa de $c$, ya que $\mathbf{K}$ es una matriz semidefinida positiva.

$$argmin_{c\in\mathbb{R}^n} \frac{1}{2}\|\mathbb{Y} - \mathbf{K}c\|_2^2 + \frac{\lambda}{2} c^T \mathbf{K} c$$

La solución se puede obtener igualando a cero la derivada parcial respecto a $c$ y resulta ser

$$c = (\mathbf{K} + \lambda I)^{-1}\mathbb{Y}$$

Para obtener una solución de $c$ dado un valor fijo de $\lambda$ hace falta resolver el siguiente sistema lineal de ecuaciones $(\mathbf{K} + \lambda I)c = \mathbb{Y}$. Como la matriz $(\mathbf{K} + \lambda I)$ es simétrica y definida positiva para cualquier $\lambda > 0$, un buen algoritmo para resolver el sistema sería la factorización de Cholesky. En el momento que se conoce $c$, la siguiente expresión da la predicción en un nuevo punto $x_{new}$

$$\hat{y}_{new} = f(x_{new}) = \sum_{j=1}^{n} c_j k(x_{new}, x_j) = \mathbf{k}(x_{new})^T c$$

Notar que esta solución de mínimos cuadrados para una función $f$ cualquiera coincide con la expresión (2,3) obtenida mediante representación dual. La primera depende de la n-tupla $c$ mientras que en esta definimos el vector $\mathbf{a}$.

Normalmente no se sabe a priori el mejor valor del parámetro y resolver el sistema lineal para cada valor de $\lambda$ es caro computacionalmente. Un método mejor que resolver el sistema para cada valor de $\lambda$, es la descomposición en valores y vectores propios de la matriz $\mathbf{K} = Q\Lambda Q^T$, donde $\Lambda$ es diagonal con sus elementos $\Lambda_{ii} \geqslant 0$ y $QQ^T = I$. Esta operación tiene un coste computacional de $O(n^3)$.

De esta forma se puede representar $G(\lambda) = (\mathbf{K} + \lambda I)$ como

$$G(\lambda) = Q\Lambda Q^T + \lambda QQ^T = Q(\Lambda + \lambda I)Q^T,$$

luego,

$$G^{-1}(\lambda) = Q(\Lambda + \lambda I)^{-1}Q^T \tag{2.6}$$

Como la matriz a invertir es diagonal, su inversa también es diagonal siendo sus elementos $\frac{1}{\Lambda_{ii}+\lambda}$ los valores propios de la matriz $G^{-1}(\lambda)$.

Se observa que la función de $\lambda$ es estabilizar el sistema. Un buen valor de éste podría estar entre los valores propios de $\mathbf{K}$ más pequeño y más grande, lo que proporcionaría una solución más generalizable. Variando el valor de $\lambda$ solamente cambia la matriz diagonal, así pues el coste computacional de calcular los valores de $c(\lambda)$ es esencialmente nulo. Una vez que se tiene $c$, se pueden obtener las predicciones $\mathbf{K}c$ con un coste de $O(n^2)$.

## 2.4. Validación cruzada

Ya hemos visto cómo calcular $c(\lambda)$ de una forma rápida al tiempo que $\lambda$ va variando, pero en este problema se necesita un mecanismo para encontrar buenos valores del parámetro de regularización. Por "buenos" se entienden los parámetros que mejor generalizan la solución, es decir, cuyas soluciones predicen bien la respuesta en valores futuros de las covariables.

Para ello podemos usar validación cruzada vista en el capítulo anterior, el método más usado para problemas de altas dimensiones. También existen otros métodos como el criterio de información de Akaike (AIC) o métodos Bayesianos, que son efectivos para problemas de dimensiones bajas.

Ya vimos el algoritmo de CV para $K$ bloques en $(1,5)$, así como el caso límite conocido como LOOCV (leave-one-out cross validation). En este método, para cada dato $x_i$ se ajusta el modelo en los $n-1$ datos restantes y se calcula la predicción del error en ese dato. La desventaja es que requiere el cálculo de $n$ predicciones del error diferentes en conjuntos de datos de tamaño $n-1$. Sin embargo, veamos que en el caso de mínimos cuadrados regularizados obtener el error LOO no tiene apenas coste.

Definimos el conjunto $S^i$ formado por todo el conjunto de datos excepto el dato $(x_i, y_i)$,

$$S^i = \{(x_1,y_1),...,(x_{i-1},y_{i-1}),(x_{i+1},y_{i+1}),...,(x_n,y_n)\}$$

Sea $f_{S^i}$ el $i-$ésimo valor de LOO para RLS (mínimos cuadrados penalizados) ajustando el modelo en el conjunto $S^i$ y evaluándolo en $x_i$. Así, el correspondiente error LOO es $(y_i - f_{S^i}(x_i))$. Definimos entonces $L_V$ y $L_E$ como los vectores formados por los valores y errores de LOO sobre el conjunto de datos, respectivamente.

Veamos que trabajar con $L_V$ y $L_E$ para RLS es computacionalmente efectivo. Definimos el vector $\mathbb{Y}^i$ cuyas componentes son las mismas que las de $\mathbb{Y}$ excepto la $i-$ésima que se reemplaza por el valor desconocido $f_{S^i}(x_i)$.

$$y^i_j = \begin{cases} y_j & \text{si } j \neq i \\ f_{S^i}(x_i) & \text{si } j = i \end{cases}$$

A continuación una cadena de desigualdades que muestran que resolviendo RLS con el vector $\mathbb{Y}^i$ en vez de $\mathbb{Y}$, se obtiene la función óptima $f_{s^i}$. Es claro que

$$\frac{1}{2}\sum_{j=1}^{n}(y^i_j - f(x_j))^2 + \frac{\lambda}{2}\|f\|^2_H \geqslant \frac{1}{2}\sum_{j\neq i}(y^i_j - f(x_j))^2 + \frac{\lambda}{2}\|f\|^2_H$$

Ahora como $f_{S^i}$ es una función óptima, el valor de la función de pérdida es menor que para cualquier función $f \in H$, entonces se cumple que

$$\frac{1}{2}\sum_{j\neq i}(y^i_j - f(x_j))^2 + \frac{\lambda}{2}\|f\|^2_H \geqslant \frac{1}{2}\sum_{j\neq i}(y^i_j - f_{S^i}(x_j))^2 + \frac{\lambda}{2}\|f_{S^i}\|^2_H$$

Por último, de la definición de $\mathbb{Y}^i$ se observa que $y^i_i = f_{s^i}(x_i) = 0$. Entonces podemos escribir

$$\frac{1}{2}\sum_{j\neq i}(y^i_j - f_{S^i}(x_j))^2 + \frac{\lambda}{2}\|f_{S^i}\|^2_H = \frac{1}{2}\sum_{j=1}^{n}(y^i_j - f_{S^i}(x_j))^2 + \frac{\lambda}{2}\|f_{S^i}\|^2_H$$

En resumen, de la desigualdad

$$\frac{1}{2}\sum_{j=1}^{n}(y^i_j - f(x_j))^2 + \frac{\lambda}{2}\|f\|^2_H \geqslant \frac{1}{2}\sum_{j=1}^{n}(y^i_j - f_{S^i}(x_j))^2 + \frac{\lambda}{2}\|f_{S^i}\|^2_H$$

se deduce que $f_{S^i}$ es la solución óptima para el problema de mínimos cuadrados penalizados.

Representamos ahora esta función optima para RLS en términos de la matriz $\mathbf{K}$, así como la nueva solución de $c$. La notación $(i)$ refiere a la $i-$ésima componente de un vector y en el caso de $(ij)$ al elemento de la fila $i$ y columna $j$ de una matriz.

$$c^i = G^{-1}\mathbb{Y}^i$$
$$f_{S^i}(x_i) = (\mathbf{K}G^{-1}\mathbb{Y}^i)_{(i)}$$

Llegados a este punto vamos a deducir las expresiones de los valores y errores de LOO, $L_V$ y $L_E$.

Si $f_{S^i}$ representa la función óptima resultado de ajustar en el modelo completo, veamos la diferencia entre $f_S$ y $f_{S^i}$ en el valor $x_i$. Como $f_S = \mathbf{K}G^{-1}\mathbb{Y}$ y $f_{S^i} = \mathbf{K}G^{-1}\mathbb{Y}^i$ se tiene que

$$f_{S^i}(x_i) - f_S(x_i) = \sum_{j=1}^{n} (\mathbf{K}G^{-1})_{(ij)}(y^i_j - y_j) = (\mathbf{K}G^{-1})_{(ii)}(f_{S^i}(x_i) - y_i)$$

y despejando $f_{S^i}(x_i)$ obtenemos $f_{S^i}(x_i)(1-(\mathbf{K}G^{-1})_{(ii)}) = f_S(x_i) - (\mathbf{K}G^{-1})_{(ii)}y_i$, luego

$$f_{S^i}(x_i) = \frac{f_S(x_i) - (\mathbf{K}G^{-1})_{(ii)}y_i}{1-(\mathbf{K}G^{-1})_{(ii)}}$$

Denotamos por $diag_m(A)$ a la matriz diagonal cuyos elementos diagonales son $A_{(ii)}$ y $diag_v(A)$ al vector columna cuyos elementos son $A_{(ii)}$. Así, para todo el conjunto de datos $L_V$ se expresa como

$$L_V = \frac{\mathbf{K}G^{-1}\mathbb{Y} - diag_m(\mathbf{K}G^{-1})\mathbb{Y}}{diag_v(I-(\mathbf{K}G^{-1}))}$$

**Nota** : La notación anterior representa el cociente de dos vectores elemento a elemento. De ahora en adelante la usaremos para deducir las expresiones de $L_V$ y $L_E$.

Por tanto ya estamos en condiciones de calcular el error correspondiente $L_E = \mathbb{Y} - L_V$:

$$\begin{aligned} L_E &= \mathbb{Y} + \frac{diag_m(\mathbf{K}G^{-1})\mathbb{Y} - \mathbf{K}G^{-1}\mathbb{Y}}{diag_v(I-\mathbf{K}G^{-1})} = \\ &= \frac{diag_m(I-\mathbf{K}G^{-1})\mathbb{Y} + diag_m(\mathbf{K}G^{-1})\mathbb{Y} - \mathbf{K}G^{-1}\mathbb{Y}}{diag_v(I-\mathbf{K}G^{-1})} = \\ &= \frac{\mathbb{Y} - \mathbf{K}G^{-1}\mathbb{Y}}{diag_v(I-\mathbf{K}G^{-1})} = \frac{(I-\mathbf{K}G^{-1})\mathbb{Y}}{diag_v(I-\mathbf{K}G^{-1})} \end{aligned}$$

Podemos simplificar la expresión del error $L_E$ usando la descomposición en valores y vectores propios desarrollada en el apartado anterior. Sustituyendo $\mathbf{K}$ y $G^{-1}$ de (2,6) se tiene

$$\begin{aligned} \mathbf{K}G^{-1} &= Q\Lambda Q^T Q(\Lambda+\lambda I)^{-1}Q^T = Q\Lambda(\Lambda+\lambda I)^{-1}Q^T = \\ &= Q(\Lambda+\lambda I-\lambda I)(\Lambda+\lambda I)^{-1}Q^T = I-\lambda G^{-1} \end{aligned}$$

Se deduce que $I-\mathbf{K}G^{-1} = \lambda G^{-1}$ y siendo $c$ la solución para el problema RLS usando el conjunto de datos completo, la expresión del error queda

$$L_E = \frac{\lambda G^{-1}\mathbb{Y}}{diag_v(\lambda G^{-1})} = \frac{c}{diag_v(G^{-1})} \tag{2.7}$$

Ya vimos que dada la descomposición de $\mathbf{K}$ se podía calcular $c(\lambda)$ en tiempo $O(n^2)$. También se puede obtener el término $G^{-1}$ en tiempo $O(n)$ haciendo lo siguiente

$$G^{-1}_{(ij)} = (Q(\Lambda+\lambda I)^{-1}Q^T)_{(ij)} = \sum_{k=1}^{n} \frac{Q_{ik}Q_{jk}}{\Lambda_{kk}+\lambda}$$

Ya tenemos todo lo necesario para calcular la expresión de $L_E$ con coste $O(n^2)$, el mismo necesario para resolver una vez el problema de RLS.

## 2.5. Aproximación por descomposición en valores singulares

En el caso del kernel lineal, existe otra forma de proceder a la hora de calcular los valores de LOO para validar el parámetro $\lambda$, cuyo desarrollo se puede consultar en [16]. Consiste en obtener la descomposición en valores singulares (SVD) de la matriz del diseño $\mathbb{X}$. Esta descomposición también podría ser válida y útil para otros kernels, con la condición de que la transformación en el espacio de características $\phi$ tenga dimensión finita.

**Teorema 2.4.** *Sea $A \in \mathbb{R}^{n\times m}$ con $n > m$ y rango(A)=$r \leqslant m$. Sean $\lambda_1, ..., \lambda_r$ los valores propios de $A^T A$, incluyendo las multiplicidades y en orden decreciente $\lambda_1 \geqslant ... \geqslant \lambda_r > 0$. Los valores singulares $\rho_i$ de A se definen como $\rho_i = \sqrt{\lambda}_i$ para $i = 1, ..., r$. Entonces existe una factorización para A tal que $A = USV^T$ siendo $\tilde{U} \in \mathbb{R}^{n\times n}$ y $\tilde{V} \in \mathbb{R}^{m\times m}$ ortonormales ($\tilde{U}^T\tilde{U} = \tilde{U}\tilde{U}^T = I_n$ y $\tilde{V}^T\tilde{V} = \tilde{V}\tilde{V}^T = I_m$), y $\tilde{S} \in \mathbb{R}^{n\times m}$ con $\rho_i$ en los elementos diagonales para $i = 1, ..., m$.*

Las columnas de $\tilde{U}$ y $\tilde{V}$ son los vectores singulares a izquierda y derecha respectivamente.

Los valores singulares $\rho_i$ están ordenados de forma no creciente $\rho_1 \geqslant ... \geqslant \rho_m \geqslant 0$. El número de valores singulares nulos es igual al de columnas linealmente dependientes de $A$. Si el rango de $A$ es $r$ entonces hay $m - r$ valores singulares nulos. Así, si hay algún valor singular de $A$ nulo, el rango de $A$ será $r < m$.

Se define el número de condición de $A$ como $c_A = \frac{\rho_1}{\rho_m}$. Si $c_A = \infty$ (por ser $\rho_m = 0$), $A$ estará mal condicionada. Será singular, luego no invertible.

En nuestro problema $m = p+1$ con $n \gg p+1$ y $A = \mathbb{X}$ que tiene rango completo $p+1$. Notar que se puede formular esta descomposición de una forma más compacta. Sea $U$ la matriz compuesta por las primeras $p+1$ columnas de $\tilde{U}$ (y así $U'$ estará compuesta por las $n-(p+1)$ columnas restantes) y $S = diag(\rho_1, ..., \rho_{p+1}) \in \mathbb{R}^{(p+1)\times(p+1)}$, entonces se tiene que $\mathbb{X}$ se puede descomponer como

$$\mathbb{X} = \tilde{U}\tilde{S}\tilde{V}^T = \tilde{U}\begin{bmatrix} S \\ O \end{bmatrix}\tilde{V}^T = \begin{bmatrix} U & U' \end{bmatrix}\begin{bmatrix} S \\ O \end{bmatrix}\tilde{V}^T$$

Veamos ahora las expresiones de $\mathbf{K}$, $(\mathbf{K}+\lambda I)^{-1}$, etc para deducir (2,7) en función de esta descomposición.

$$\mathbf{K} = \mathbb{X}\mathbb{X}^T = (\tilde{U}\tilde{S}\tilde{V}^T)(\tilde{U}\tilde{S}\tilde{V}^T)^T = (\tilde{U}\tilde{S}\tilde{V}^T)(\tilde{V}\tilde{S}\tilde{U}^T) = \tilde{U}\tilde{S}^2\tilde{U}^T$$

Ahora sustituimos la expresión de $\mathbf{K}$ para obtener $\mathbf{K}+\lambda I$,

$$\begin{aligned} \mathbf{K}+\lambda I &= \tilde{U}\tilde{S}^2\tilde{U}^T + \lambda I_n = \begin{bmatrix} U & U' \end{bmatrix}\begin{bmatrix} S^2+\lambda I_{p+1} & 0 \\ 0 & \lambda I_{n-(p+1)} \end{bmatrix}\begin{bmatrix} U^T \\ U'^T \end{bmatrix} = \\ &= U(S^2+\lambda I_{p+1})U^T + \lambda U'U'^T = U(S^2+\lambda I_{p+1})U^T + \lambda(I_n - UU^T) \end{aligned}$$

luego su inversa será

$$\begin{aligned} (\mathbf{K}+\lambda I)^{-1} &= (\tilde{U}\tilde{S}^2\tilde{U}^T + \lambda I_n)^{-1} = (\begin{bmatrix} U & U' \end{bmatrix}\begin{bmatrix} S^2+\lambda I_{p+1} & 0 \\ 0 & \lambda I_{n-(p+1)} \end{bmatrix}\begin{bmatrix} U^T \\ U'^T \end{bmatrix})^{-1} = \\ &= \begin{bmatrix} U & U' \end{bmatrix}\begin{bmatrix} S^2+\lambda I_{p+1} & 0 \\ 0 & \lambda I_{n-(p+1)} \end{bmatrix}^{-1}\begin{bmatrix} U^T \\ U'^T \end{bmatrix} = U(S^2+\lambda I)^{-1}U^T + \lambda^{-1}U'U'^T = \\ &= U(S^2+\lambda I)^{-1}U^T + \lambda^{-1}(I - UU^T) = U[(S^2+\lambda I)^{-1} - \lambda^{-1}I]U^T + \lambda^{-1}I \end{aligned}$$

Ya estamos en condiciones de calcular el vector $c$:

$$c = (\mathbf{K}+\lambda I)^{-1}\mathbb{Y} = U[(S^2+\lambda I)^{-1} - \lambda^{-1}I]U^T\mathbb{Y} + \lambda^{-1}\mathbb{Y}$$

Como $G^{-1}_{(ij)} = \sum_{k=1}^{p+1} U_{(ik)}U_{(jk)}[(S_{(kk)}+\lambda)^{-1} - \lambda^{-1}] + [i=j]\lambda^{-1}$, se tiene que $G^{-1}_{(ii)}$ es

$$G^{-1}_{(ii)} = \sum_{k=1}^{p+1} U^2_{(ik)}[(S_{(kk)}+\lambda)^{-1} - \lambda^{-1}] + \lambda^{-1}$$

Finalmente, sustituyendo estas expresiones de $c$ y $G^{-1}$ tenemos que el error LOO es

$$L_E = \frac{c}{diag_v(G^{-1})} = \frac{U[(S^2+\lambda I)^{-1} - \lambda^{-1}I]U^T\mathbb{Y} + \lambda^{-1}\mathbb{Y}}{diag_v(U[(S^2+\lambda I)^{-1} - \lambda^{-1}I]U^T + \lambda^{-1}I)}$$

Cabe destacar la descomposición en valores singulares truncada. Es útil cuando los primeros $r$ valores singulares son mucho más grandes que los $(p+1)-r$ restantes, es decir, $\rho_1 \geqslant \rho_2 \geqslant ... \geqslant \rho_r \gg \rho_{r+1} \geqslant \rho_{r+2} \geqslant ... \geqslant \rho_{p+1} \geqslant 0$. Entonces una representación adecuada de $\mathbb{X}$ consiste en quedarnos con los $r$ primeros valores singulares de $\mathbb{X}$ y las columnas correspondientes de $U$ y $V$. $V_r$ y $U_r$ contienen las $r$ primeras columnas de $V$ y $U$, y $S_r$ contiene las primeras $r$ filas y columnas de $S$.

A continuación damos la expresión de $\mathbb{X}$ y la solución óptima truncada correspondiente del vector de parámetros $\beta$.

$$\mathbb{X}_{(r)} = U_r S_r V_r^T$$

$$\beta^*_{(r)} = V_r S_r^{-1} U_r^T \mathbb{Y}$$

## 2.6. Formulación general para la regularización de Tikhonov

En este apartado vamos a considerar una expresión para la regularización de Tikhonov más general que la vista hasta ahora. Puede consultarse más información acerca de esta generalización en el artículo [6]. Comencemos definiendo un vector $x$ con norma cuadrática como $\|x\|_P = x^T P x$, siendo $P$ una matriz definida positiva. Manteniendo el resto de matrices, vectores y parámetros como hasta ahora, la expresión para la función objetivo queda

$$J(\beta) = \|\mathbb{Y} - \mathbb{X}\beta\|_P + \lambda\|\beta - \bar{\beta}\|_Q$$

En términos de modelos estadísticos, si las componentes $y_i$ del vector $\mathbb{Y}$ son estadísticamente independientes, la matriz $P$ será diagonal. Sus elementos diagonales $P_{ii}$ serán la inversa de la varianza del error muestral de $y_i$, $P_{ii} = \frac{1}{\sigma^2_{y_i}}$.

Por el contrario, si los errores de $y_i$ no son estadísticamente independientes entonces $P$ será la inversa de la matriz de convarianzas de $\mathbb{Y}$, $P = V_{\mathbb{Y}}^{-1}$. Además, la matriz $Q$ es definida positiva y el vector de referencia de parámetros $\bar{\beta}$ permite imponer restricciones sobre estos.

Si desarrollamos la función objetivo $J(\beta)$ e igualamos a cero la derivada parcial respecto a $\beta$, obtenemos la solución óptima para $\hat{\beta}$. Sea:

$$J(\beta) = \beta^T\mathbb{X}^T P\mathbb{X}\beta - 2\beta^T\mathbb{X}^T P\mathbb{Y} + \mathbb{Y}^T P\mathbb{Y} + \lambda\beta^T Q\beta - 2\lambda\beta^T Q\bar{\beta} + \lambda\bar{\beta}^T Q\bar{\beta}$$

entonces,

$$\left(\frac{\partial J(\beta)}{\partial\beta}\right) = 2\mathbb{X}^T P\mathbb{X}\beta - 2\mathbb{X}^T P\mathbb{Y} + 2\lambda Q\beta - 2\lambda Q\bar{\beta} = 0$$

y por tanto

$$\hat{\beta}_\lambda = (\mathbb{X}^T P\mathbb{X} + \lambda Q)^{-1}(\mathbb{X}^T P\mathbb{Y} + \lambda Q\bar{\beta})$$

Si los $n$ errores muestrales de $y_i$ no son conocidos individualmente, usualmente se toma $P = I_n$. De la misma manera, si las $p+1$ diferencias de parámetros $\beta - \bar{\beta}$ tienen todas la misma importancia, se toma $Q = I_{p+1}$. Por último, si no se tiene un conjunto de parámetros de referencia, entonces $\bar{\beta} = 0$. Con

todas estas simplificaciones volvemos a tener la solución óptima (1,5) con la que hemos trabajado desde el principio de este trabajo :

$$\hat{\beta}_\lambda = (\mathbb{X}^T\mathbb{X} + \lambda I)^{-1}\mathbb{X}^T\mathbb{Y}$$

## 2.7. Métodos alternativos de optimización

Existen estudios recientes que han desarrollado métodos de optimización para la resolución del problema de mínimos cuadrados penalizados. Hemos obtenido información resumida sobre dos de ellos de las siguientes publicaciones [14] y [17].

El primero es una alternativa al método LASSO que estudiamos en el Capítulo 1. Consiste en optimizar la función objetivo de RLS con penalización de norma L1, lo cual parece atractivo para obtener modelos predictivos adecuados que también tengan una representación mas interpretable y parsimoniosa. Existe una gran variedad de aproximaciones para la estimación de los parámetros que solventan el inconveniente de la no-diferenciabilidad de la función objetivo, en el caso L1. Notar que la formulación recuerda a un problema de optimización convexo sin restricciones.

$$min\|\mathbb{Y} - \mathbb{X}\beta\|_2^2 + \lambda\|\beta\|_1$$

Para evitar la no-diferenciabilidad se puede transformar en uno con restricciones definido por un conjunto convexo como a continuación, estando $t$ relacionado con $\lambda$ de forma inversa.

$$min\|\mathbb{Y} - \mathbb{X}\beta\|_2^2$$
$$s.a.\|\beta\|_1 \leqslant t$$

Entre todas las técnicas que hay, cabe destacar la programación cuadrática. Consúltese [14] para más detalle.

Otro método importante es el de las direcciones alternantes (ADM). Se aplica a problemas cuadráticos con restricciones como pueden ser la regularización L2 y L1. Sean $l,u \in \mathbb{R}^{p+1}$ vectores con componentes finitas, $A \in R^{n\times(p+1)}$ y $e = (1,...,1)^T$. Se consideran los siguientes problemas, equivalentes a los de regularización con L2 en (1,4) y L1 en (1,6), respectivamente.

$$min_{l\leqslant\beta\leqslant u}\frac{1}{2}\|\mathbb{Y} - \mathbb{X}\beta\|_2^2 + \frac{1}{2}\lambda\|A\beta\|_2^2$$

$$min_{l\leqslant\beta\leqslant u}\frac{1}{2}\|\mathbb{Y} - \mathbb{X}\beta\|_2^2 + \lambda e^T\beta$$

La importancia de estos problemas aparece cuando las matrices $\mathbb{X}$ y $A$ tienen una estructura específica y las técnicas de optimización son capaces de explotarla.

Una aplicación relevante es la recuperación de imágenes con ruido y observaciones borrosas. Se asume que las funciones son estrictamente convexas para que admitan una única solución. Ejemplos típicos son sistemas que capturan una imagen $\beta$ y devuelven datos dañados $\mathbb{Y}$. El modelo más común del proceso de degradación es $\mathbb{Y} = \mathbb{X}\beta + \eta$ donde $\eta$ es el ruido añadido. Recuperar $\beta$ de $\mathbb{Y}$ es normalmente un problema inverso mal condicionado que debe ser regularizado.

# Capítulo 3

# Aplicación a datos reales

El objetivo de este capítulo va a ser mostrar, de forma práctica e ilustrativa, la teoría desarrollada anteriormente. Esto se hará sobre dos conjuntos de datos reales: el primero estará formado por datos clínicos de hombres con cáncer de próstata y el segundo por características sobre los distintos tipos de casas vendidas en Ames, una ciudad estadounidense. Para ello se utilizará el lenguaje de programación R (ver [13]).

En primer lugar, el conjunto de datos Prostate nos será útil a la hora de visualizar analítica y gráficamente el funcionamiento de los distintos métodos de penalización, veremos Ridge, Elastic Net y LASSO. El algoritmo de validación cruzada será el encargado de seleccionar los parámetros de regularización óptimos, minimizando el error medio de VC. Lo haremos tomando $K = 10$ y $K = n$ (LOOCV). Además, compararemos los parámetros de los modelos obtenidos por las distintas penalizaciones y por mínimos cuadrados ordinarios.

Por otro lado, se ha elegido AmesHousing por ser un conjunto de datos con un número elevado de datos y variables. La importancia recae en la capacidad de ajuste de la regresión penalizada cuando muchas de las variables están muy correladas entre sí. También introduciremos no linealidad en el modelo mediante un núcleo gaussiano.

El paquete `glmnet`, cuyos autores son J. Friedman, T. Hastie, R. Tibshirani y N.Simon, va a ser principal en este trabajo. Está basado en un algoritmo muy rápido y aparece explicado con más detalle en [9] y [5]. Se usan coordenadas cíclicas descendentes que optimizan la función objetivo sucesivamente sobre cada parámetro y se repite de forma cíclica hasta que converge. Serán útiles las funciones `glmnet` y `cv.glmnet` entre otras. Para la implementación del núcleo gaussiano será necesario el paquete `KRLS`, que puede consultarse en [8].

## 3.1. Conjunto de datos "Prostate"

El conjunto de datos Prostate está recogido de un estudio hecho por Stamey en el año 1989 a hombres con cáncer de próstata. Consta de 9 variables (8 predictores clínicos y la variable respuesta que se quiere estudiar, el logaritmo del índice de PSA) y datos de 97 hombres que recibieron un tratamiento de prostatectomía radical. Este conjunto está incluido en el paquete de R `ElemStatLearn`. Veamos a continuación cada una de las variables

- lcavol: el logaritmo del volumen del cáncer
- lweight: el logaritmo del peso de la próstata
- age: la edad en años del hombre
- lbph: el logaritmo de la cantidad de hiperplasia prostática benigna
- svi: variable binaria sobre si hay invasión a la vesícula seminal o no

- lcp: el logaritmo de la penetración capsular
- Gleason: el grado de agresividad del cáncer (categoría ordinal: 2-6 escasa, 7 intermedia, 8-10 alta)
- ppg45: porcentaje del grado de Gleason 4 ó 5
- lpsa: logaritmo del antígeno prostático específico

Se tipifican todas las variables explicativas, lo cual nos permite eliminar las unidades de cada una de las variables y en consecuencia, que sean comparables.

Para empezar, veamos el resultado de aplicar regresión penalizada mediante la función `glmnet`, para los valores del parámetro $\alpha = 0, 0'5, 1$. Estos corresponden a regresión contraída (más conocida como ridge), elastic net para ese valor de $\alpha$ que selecciona las variables predictoras correladas en grupos y LASSO, respectivamente. Se puede ver el efecto de estos métodos sobre los coeficientes ajustados del modelo en tanto que, conforme el parámetro de penalización aumenta estos tienden a cero. En las siguientes gráficas se representa el valor de los 8 parámetros en función del logaritmo de $\lambda$. Puede observarse cómo en el primer caso los coeficientes tienden asintóticamente a cero, mientras que en LASSO algunos de ellos toman el valor nulo a partir de un determinado $\lambda$. Por esta razón, podemos considerar estas técnicas como un método alternativo a la selección de variables. Las gráficas para Elastic Net y LASSO se pueden consultar en el anexo $(A,1)$.

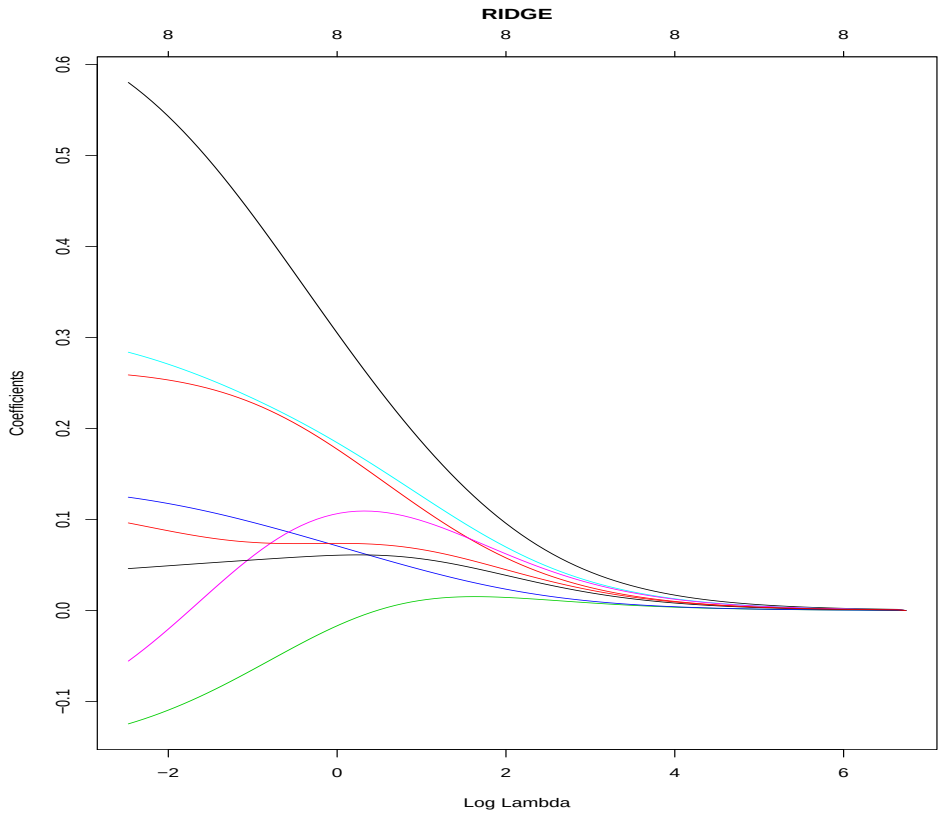


Figura 3.1: Gráfica de los coeficientes en función de $\lambda$, en el caso Ridge.

A continuación seleccionamos los parámetros óptimos de estos métodos por validación cruzada, los cuales corresponden al parámetro cuyo error medio de validación cruzada (RSME) es mínimo. Para ello se usa la función `cv.glmnet`, que contiene un argumento para determinar el valor de $K$. En este caso, hemos tomado $K = 10$ ya que suele ser el más común en la literatura y $K = n = 97$, el correspondiente LOOCV. En la siguiente tabla se observa que tanto el valor del parámetro óptimo como su respectivo

error, es menor en el caso de LOO en los tres métodos. Esta diferencia se debe a que el algoritmo de cálculo difiere en el número de bloques, sin embargo el modelo óptimo que se obtiene es similar en ambos casos.

| Métodos | **Ridge** | **ElasticNet** | **LASSO** |
|---|---|---|---|
| $\lambda_{10}$ | 0.1223669 | 0.05922871 | 0.03250172 |
| $RSME_{10}$ | 0.5548516 | 0.5566571 | 0.5588415 |
| $\lambda_{97}$ | 0.09256606 | 0.005272629 | 0.001508596 |
| $RSME_{97}$ | 0.5368554 | 0.5410124 | 0.5413085 |

Cuadro 3.1: Tabla comparativa de los $\lambda$ óptimos y errores de CV $K = 10$ y LOOCV.

Es posible obtener en un gráfico los errores por validación cruzada frente al parámetro $\lambda$ (en este caso la función toma por defecto como eje de abscisas $\log(\lambda)$). Para cada parámetro, se representa un intervalo de longitud 2 en el que se encuentra el error correspondiente, y el error medio marcado en color rojo. Además, se señala con una recta vertical el valor de $\lambda$ cuyo error medio es mínimo y por otro lado el mayor valor de $\lambda$ que tiene mínimo error, aunque nosotros sólo nos fijaremos en el primero. Las gráficas del error por validación cruzada de Elastic Net y LASSO se encuentran en el anexo $(A,1)$.

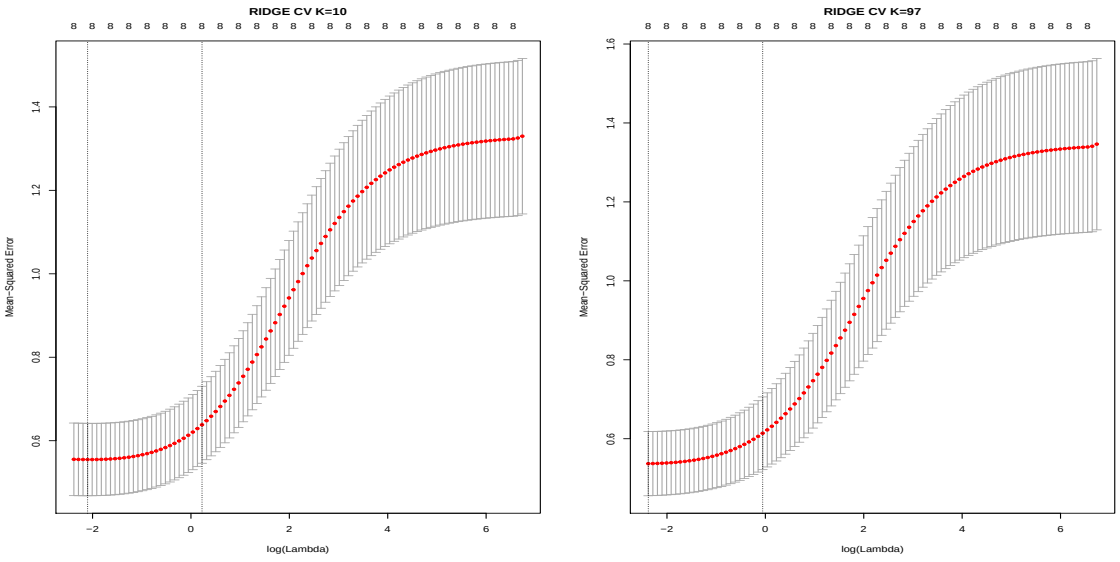


Figura 3.2: Error RSME por CV $K = 10$ (izquierda) y LOOCV (derecha), en el caso ridge.

En la siguiente tabla aparecen los coeficientes de los modelos óptimos y la norma L1 y L2 de estos, para cada uno de los métodos de penalización y para el modelo obtenido por mínimos cuadrados. También se incluye el parámetro óptimo seleccionado y el error por VC. Se puede observar una mejora del modelo LASSO con respecto a OLS, en el que algunos coeficientes son nulos o casi nulos. Asimismo, la norma de estos coeficientes va disminuyendo. De esta forma se tiene una solución más parsimoniosa e interpretable del modelo estimado.

| Coeficientes | **OLS** | **Ridge** | **ElasticNet** | **LASSO** |
|---|---|---|---|---|
| (Intercept) | 2.47839 | 2.47838688 | 2.47838688 | 2.478386878 |
| lcavol | 0.66515 | 0.55165420 | 0.58088089 | 0.598981930 |
| lweight | 0.26648 | 0.25472498 | 0.23751685 | 0.236691077 |
| age | -0.15820 | -0.11301143 | -0.07019143 | -0.069821184 |
| lbph | 0.14031 | 0.11918711 | 0.09414722 | 0.093914106 |
| svi | 0.31533 | 0.27376630 | 0.24906789 | 0.246124889 |
| lcp | -0.14829 | -0.02842399 | 0.00000000 | 0.000000000 |
| gleason | 0.03555 | 0.04853070 | 0.01230465 | 0.003326796 |
| pgg45 | 0.12572 | 0.08941581 | 0.06734462 | 0.066431463 |
| **NormaL1** | 1.855017 | 1.478715 | 1.311454 | 1.315291 |
| **NormaL2** | 0.8346466 | 0.6944715 | 0.6887299 | 0.7024945 |
| $\lambda$ | - | 0.1223669 | 0.05922871 | 0.03250172 |
| **RSME** | - | 0.5548516 | 0.5566571 | 0.5588415 |

Cuadro 3.2: Tabla comparativa para los coeficientes de los modelos óptimos por OLS, Ridge, Elastic Net y LASSO.

Por último, una representación gráfica de los coeficientes de los métodos Ridge y LASSO en la que aparecen señalados mediante rectas verticales los parámetros óptimos correspondientes a OLS y validación cruzada para $K = 10$ y 97. Notar que en los dos casos la penalización es menor para LOOCV y que LASSO presenta una mayor diferencia entre coeficientes con respecto a OLS.

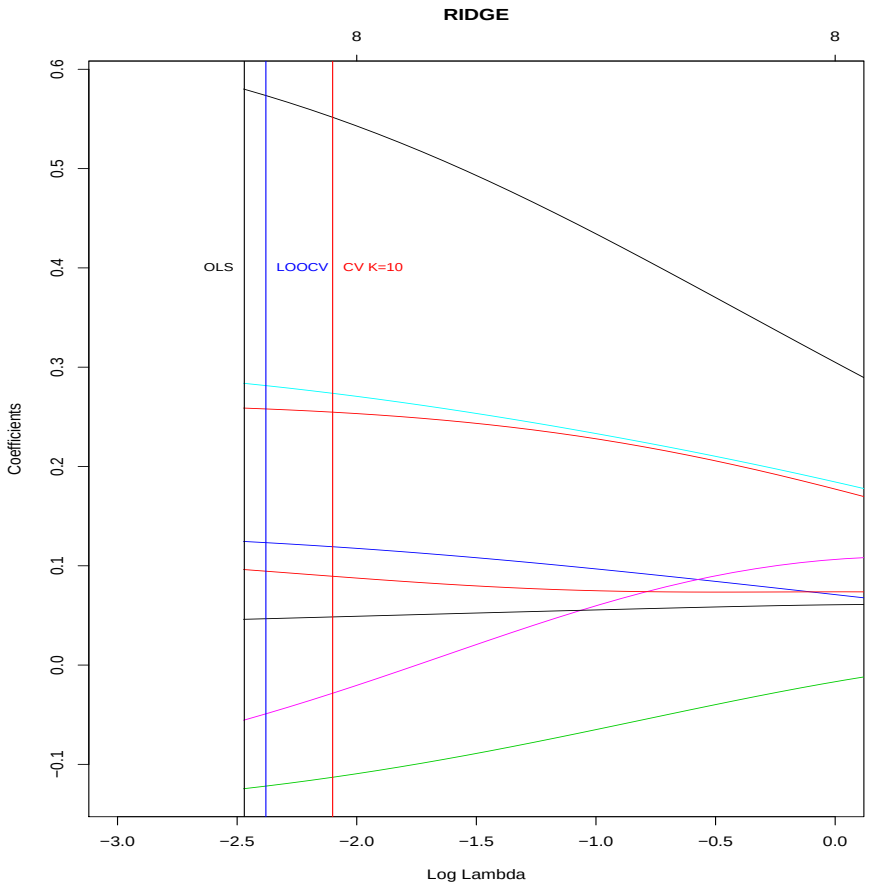


Figura 3.3: Representación de los coeficientes óptimos para el modelo ridge mediante OLS, CV $K = 10$ y LOOCV.

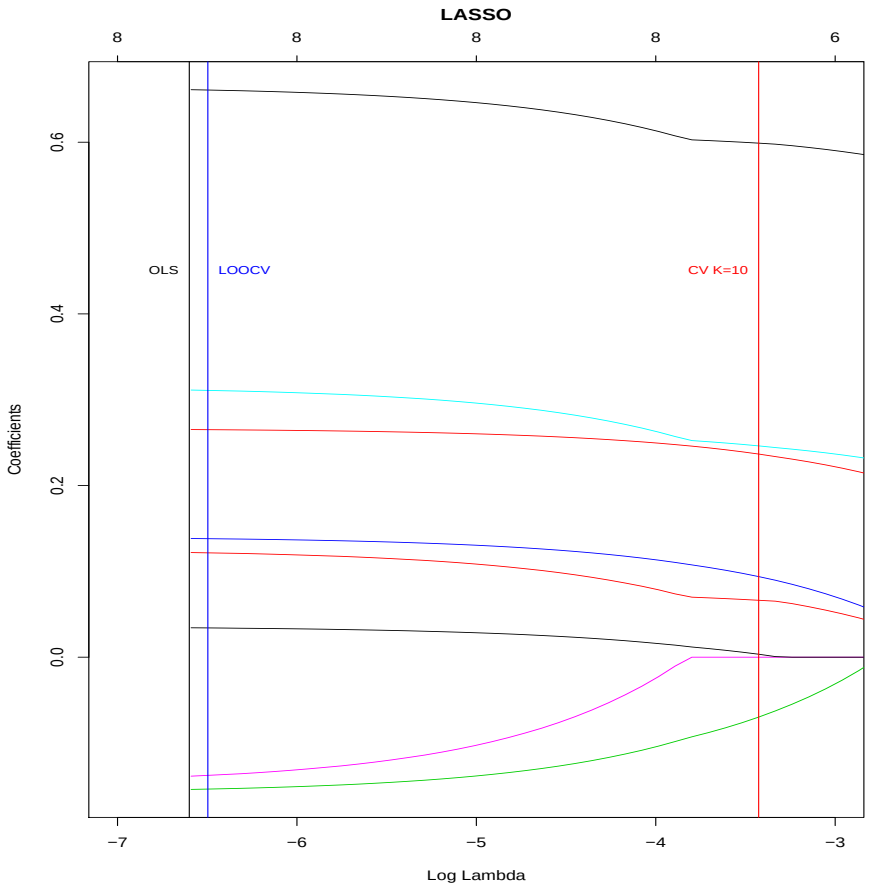


Figura 3.4: Representación de los coeficientes óptimos para el modelo LASSO mediante OLS, CV $K = 10$ y LOOCV.

## 3.2. Conjunto de datos "Ames Housing"

El conjunto de datos AmesHousing contiene información sobre las ventas de propiedades que tuvieron lugar en la ciudad de Ames (en el estado de Iowa, Estados Unidos) en el período de tiempo del año 2006 al 2010. Estos datos fueron proporcionados por el Departamento de Asesoría de la ciudad de Ames, cuya información puede encontrarse en [3].

Hay 2930 datos de los cuales se describen 82 variables. Entre estas variables, 80 están directamente relacionadas con la calidad y cantidad de varios factores físicos de las propiedades y las 2 restantes son identificadores de estas (Order y PID), luego no las incluiremos entre las variables predictoras.

Estas variables pueden clasificarse por su naturaleza aunque si se desea información individual, consúltese [3].

- 20 continuas: relacionadas con dimensiones de superficies más frecuentes como pueden ser el tamaño del terreno y la superficie total de la vivienda. Además, se incluyen otras características más específicas como el área del sótano, de la sala de estar o incluso del porche.

- 14 discretas: normalmente cuantifican la cantidad de algo en la casa como puede ser el número de cocinas, habitaciones y baños que hay en e sótano y en la parte principal de la vivienda. También se incluyen variables como la capacidad del garaje y las fechas de construcción o remodelación de la casa.

- 46 categóricas (23 ordinales y 23 nominales): están formadas por varias categorías, desde 2 hasta 28. La más pequeña es la variable Street con las categorías Gravel y Paved, y la más extensa siendo Neighborhood con todas las áreas que limitan la ciudad de Ames. Las variables nominales generalmente identifican varios tipos de viviendas, garajes, materiales y condiciones ambientales. Sin embargo, las variables ordinales clasifican varias cosas sobre la vivienda.

Comenzamos transformando el conjunto de datos original en uno más adecuado a la hora de ajustar modelos de regresión. Eliminando las filas que contienen datos ausentes (*NA*) nos quedamos con 2269 de las 2930 iniciales. Respecto a las variables explicativas, separamos las numéricas de las categóricas. Las primeras van a ser tipificadas y finalmente tendremos 33, ya que no se consideran PID y Order. Por otro lado, procedemos a convertir los factores en variables "dummies" (para cada variable categórica, se toma una categoría como base y se generan variables binarias del resto de categorías con respecto a esta). Además, definimos como variable respuesta SalePrice. Así, el nuevo conjunto de datos consta de 2269 datos y 291 variables, que finalmente se reducirán a 276 eliminando variables "dummies" con varianza nula que no aportarán información a la hora de explicar la respuesta.

En este caso, estamos manejando una gran cantidad de datos que van a presentar problemas de colinealidad entre variables y en consecuencia, correlación entre las columnas de la matriz del diseño. Esto se reduce a que $\mathbb{X}^T\mathbb{X}$ va a estar mal condicionada a la hora de obtener la solución de los coeficientes ajustados por mínimos cuadrados. Como veremos, los coeficientes de OLS van a ser extremadamente elevados y por tanto no serán interpretables en la realidad.

Se han calculado los valores propios de esta matriz $\mathbb{X}^T\mathbb{X}$ para ver que efectivamente está mal condicionada, ya que muchos de ellos serán nulos o estarán muy próximos a cero. Debido a la gran cantidad de datos, será más representativo verlo en un gráfica en escala logarítmica. Los valores más pequeños (tienden a -∞) corresponden con los valores propios nulos y los valores negativos cercanos al 0 son los valores propios muy próximos este.

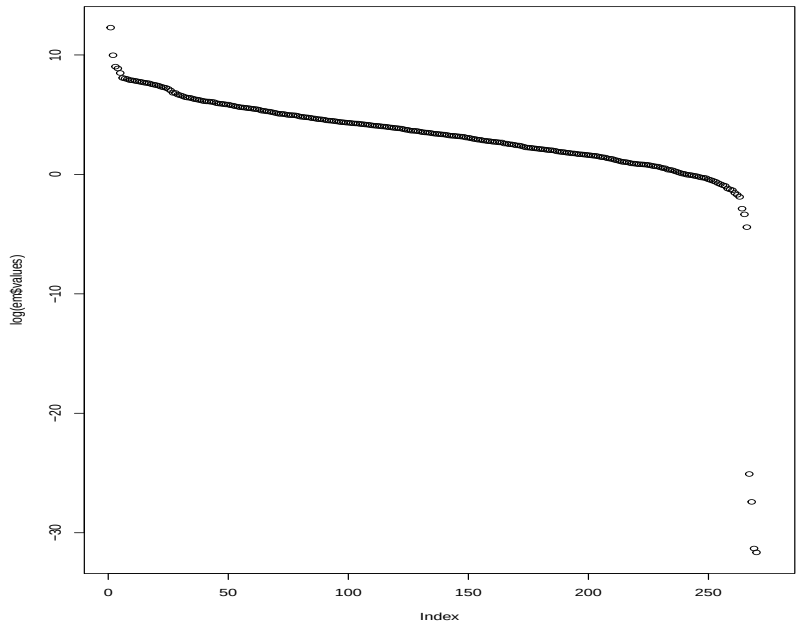


Figura 3.5: Representación de los valores propios de $\mathbb{X}^T\mathbb{X}$.

Llegados a este punto, se hace necesaria la implementación de técnicas de penalización a este conjunto de datos que presenta problemas de colinealidad. A continuación se ajustan los modelos por mínimos cuadrados, ridge y LASSO, para comprobar que efectivamente con estos métodos se obtiene una

mejora respecto a OLS a la hora de interpretar los coeficientes. Veámoslo en la siguiente tabla en la que también se ha incluido la proporción de varianza explicada $R^2$ para cada modelo.

| Métodos | **OLS** | **Ridge** | **LASSO** |
|---|---|---|---|
| $\lambda_{opt}$ | - | 17001.34 | 484.2042 |
| *RSME* | - | 986736253 | 1031339728 |
| NormaL1 | 1.045e+07 | 2.24e+06 | 1.6e+06 |
| $R^2$ | 0.9392 | 0.8481959 | 0.8744792 |

Cuadro 3.3: Tabla comparativa de los modelos OLS, ridge y LASSO.

Es evidente la reducción de magnitud de los coeficientes de los modelos obtenidos mediante regresión penalizada. La norma L1 de estos por LASSO es casi diez veces menor que por mínimos cuadrados, al mismo tiempo que la proporción de varianza explicada se mantiene alta. Es lógico que el mayor valor de $R^2$ aparezca para OLS ya que es un modelo que incluye todas las variables, sin embargo, a la hora de interpretar el modelo en la realidad es mucho más útil un modelo parsimonioso como el de LASSO, que además mantiene un $R^2$ alto. Veamos las gráficas del error por VC (con $K = 10$) de ridge y LASSO para comparar los grados de libertad obtenidos en cada uno, es decir, número de variables explicativas seleccionadas (no nulas).

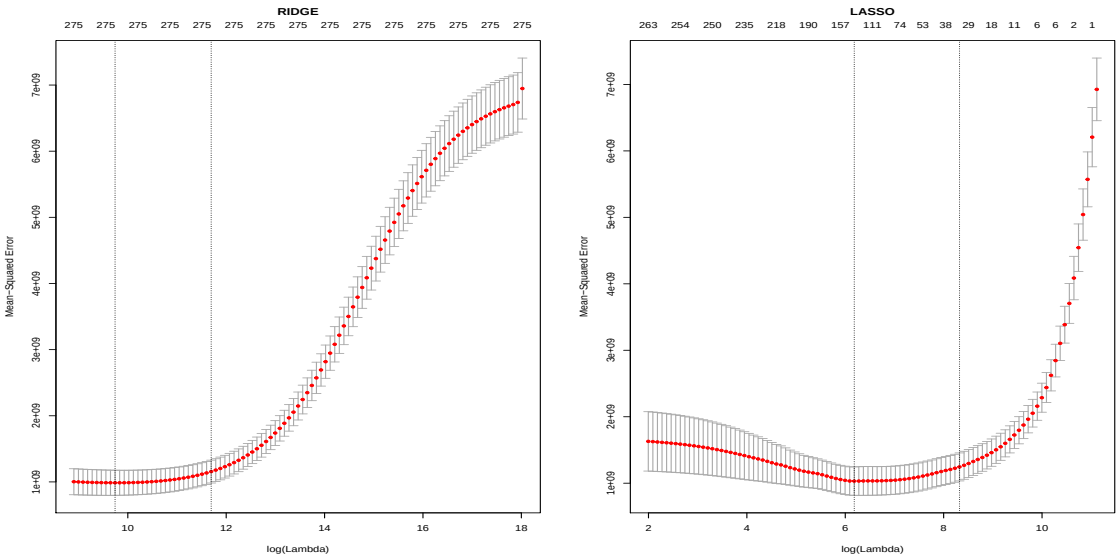


Figura 3.6: Gráficas de los RSME por CV $K = 10$ para ridge (izquierda) y LASSO (derecha).

En conclusión, ridge mantiene todas las variables (con 275 grados de libertad) aunque con valores más pequeños para los coeficientes, luego se comprueba el efecto de contraer que se explicó al introducir esta penalización con norma L2. Sin embargo, para LASSO se observa una solución sparse en la que los grados de libertad están entre los valores 111 y 157. Por tanto, ha reducido el número de variables explicativas a prácticamente la mitad como cabía esperar.

Comparemos ahora a grandes rasgos los coeficientes obtenidos por mínimos cuadrados y LASSO. Entre las variables numéricas llaman la atención 1st Flr SF y 2nd Flr SF (superficie de la 1ª y 2ª planta respectivamente) cuyos coeficientes eran altos y LASSO las elimina por completo a cambio de introducir un coeficiente de los más elevados para la variable Gr.Liv.Area, que representa el área de la parte de la casa que no es sótano. Por ejemplo, la variable Year Built (año de construcción) mantiene un coeficiente alto en los dos modelos. Así se puede considerar que junto a Gr.Liv.Area, son de las variables más influyentes para predecir el precio de venta de una vivienda (SalePrice). También se ve anulada la dependencia de la variable Year Sold (año de venta), que en caso de mínimos cuadrados venía con un

coeficiente negativo del orden de $10^2$.

Por otro lado, se observa que muchas de las variables dummies (categorías) que convertimos de los factores tienen coeficientes nulos mientras que en el modelo OLS tienen coeficientes del orden de $10^3$ y $10^4$. Destaca el caso de las variables categóricas Ms.Subclass (tipo de vivienda), Exterior 1 y 2 (cerramiento exterior), Heating (tipo de calefacción), Electrical (tipo de sistema eléctrico), Garage type (tipo de garaje), Fence (tipo de vallado), etc, en las que la mayoría de categorías toman valores nulos. Esto significa que con respecto a la categoría tomada como base, no se incrementa ni se reduce el precio de una vivienda en el caso de que fuera de otra categoría. Dicho de otra forma, no son variables significativas de las que vaya a depender la variable respuesta, precio de venta. Sin embargo, en el caso de mínimos cuadrados estaban incrementando este precio de forma desmesurada y poco realista.

Por último, vamos a introducir la regularización de Tikhonov basada en espacios de Hilbert con núcleo reproductor que vimos en el anterior capítulo. Lo haremos mediante un kernel gaussiano del que también seleccionaremos el parámetro $\sigma$, de esta manera, nos permite ajustar modelos basados en no-linealidad para conjuntos de datos de gran dimensión. Usando la función de R `krls`, se ajusta un modelo gaussiano óptimo mediante validación cruzada (con $K = 10$). Se obtienen los parámetros óptimos $\lambda = 1{,}20325$ y $\sigma = 275$, y un $R^2 = 0{,}9218$ para el modelo óptimo.

Va a ser útil representar los valores ajustados de la variable respuesta (SalePrice) sobre el mismo conjunto de datos, frente a los valores reales de esta variable. Veámoslo para mínimos cuadrados y el modelo kernel y así comprobar la gran utilidad de estas técnicas. Además de que el $R^2$ de este método es muy próximo al de OLS, se observa la poca diferencia a la hora de predecir con ambos métodos.

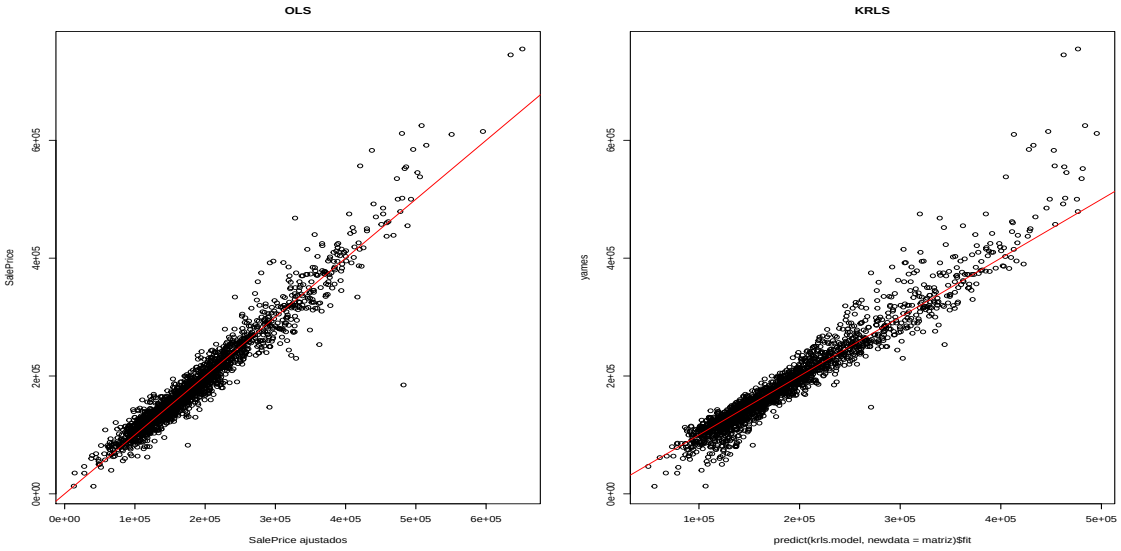


Figura 3.7: Gráficas de los valores predichos frente a los valores reales mediante OLS (izquierda) y KRLS (derecha).

# Bibliografía


[1] C.M. BISHOP, *Pattern Recognition and Machine Learning*. Springer, 2006.

[2] S. CHATTERJEE, A.S. HADI, *Regression analysis by example*. Wiley, 5ª Edición, Octubre 2012.

[3] D.D.COOK, *Journal of Statistics Education Volume 19, Number 3*. 2011, `www.amstat.org/publications/jse/v19n3/decock.pdf`

[4] T. EVGENIOU, M. PONTIL, T. POGGIO *Regularization networks and support vector machines*. Advances in Computational Mathematics 13(1): 1?50, 2000.

[5] J. FRIEDMAN, T. HASTIE, R. TIBSHIRANI, *Regularization Paths for Generalized Linear Models via Coordinate Descent.*. Journal of Statistical Software, 33(1), 1-22, 2010. `http://www.jstatsoft.org/v33/i01/`

[6] H.P. GAVIN, *Least Squares Parameter Estimation, Tikhonov Regularization, and Singular Value Decomposition*. September 16, 2013.

[7] F. GIROSI, M. JONES, T. POGGIO, *Regularization theory and neural network architectures*. Neural Computation 7: 219?269, 1995.

[8] J. HAINMUELLER, C. HAZLETT, *KRLS: Kernel-based Regularized Least squares (KRLS).*. R package version 0.3-7, 2014 `http://CRAN.R-project.org/package=KRLS`

[9] T.HASTIE, J.QIAN, *Glmnet Vignette*. September 13, 2016. disponible en `https://web.stanford.edu/~hastie/Papers/Glmnet_Vignette.pdf`

[10] A.E. HOERL, R.W. KENNARD, *Ridge Regression: Biased Estimation for Nonorthogonal Problems*. Publicado por Taylor and Francis, Ltd. on behalf of American Statistical Association and American Society for Quality, 1970, disponible en `http://www.jstor.org/stable/1267351/`

[11] L. MÁTÉ, *Hilbert Space Methods in Science and Engineering*. Taylor and Francis, 1990.

[12] D. PEÑA, *Regresión y diseño de experimentos*. Editorial Alianza, 2002.

[13] R CORE TEAM, *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, 2015. `https://www.R-project.org/`

[14] M. SCHMIDT, *Least Squares Optimization with L1-Norm Regularization*. CS542B Project Report, December 2005.

[15] R.TIBSHIRANI, J.FRIEDMAN, T.HASTIE, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* Springer, 2ª Edición, 2009

[16] G. WAHBA, *Splines Models for Observational Data, volume 59 of CBMS-NSF Regional Conference series in Applied Mathematics*. Society for Industrial and Applied Mathematics, 1990.

[17] J. ZHANG, B. MORINI, *Solving regularized linear least-squares problems by the alternating direction method with applications to image restoration*. Electronic Transactions on Numerical Analysis, Volume 40, 2013.

[18] H.ZOU, T.HASTIE, *Regularization and variable selection via the elastic net*. Stanford University, USA, 2005, disponible en `https://web.stanford.edu/~hastie/Papers/B67.2%20(2005)%20301-320%20Zou%20&%20Hastie.pdf/`

# Anexos

## A.1. Resultados del conjunto “Prostate”

En este anexo se presentan los gráficos no incluidos en el Capítulo 3.

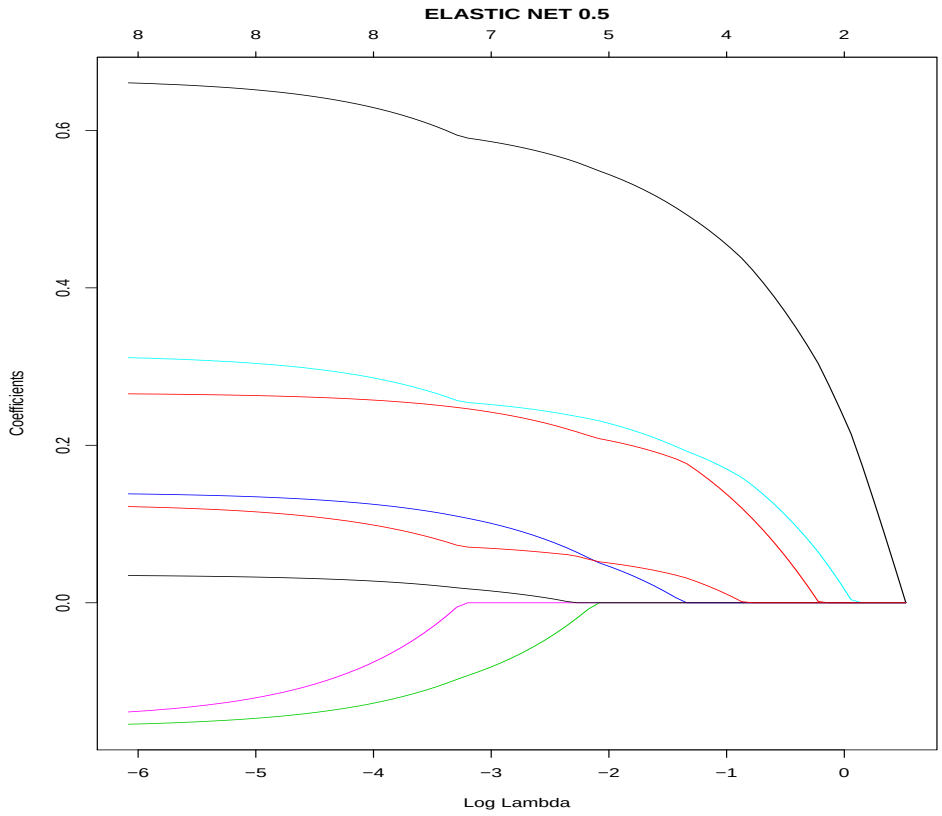


Figura A.1: Gráfica de los coeficientes en función de $\lambda$, en el caso Elastic Net $\alpha = 0{,}5$.

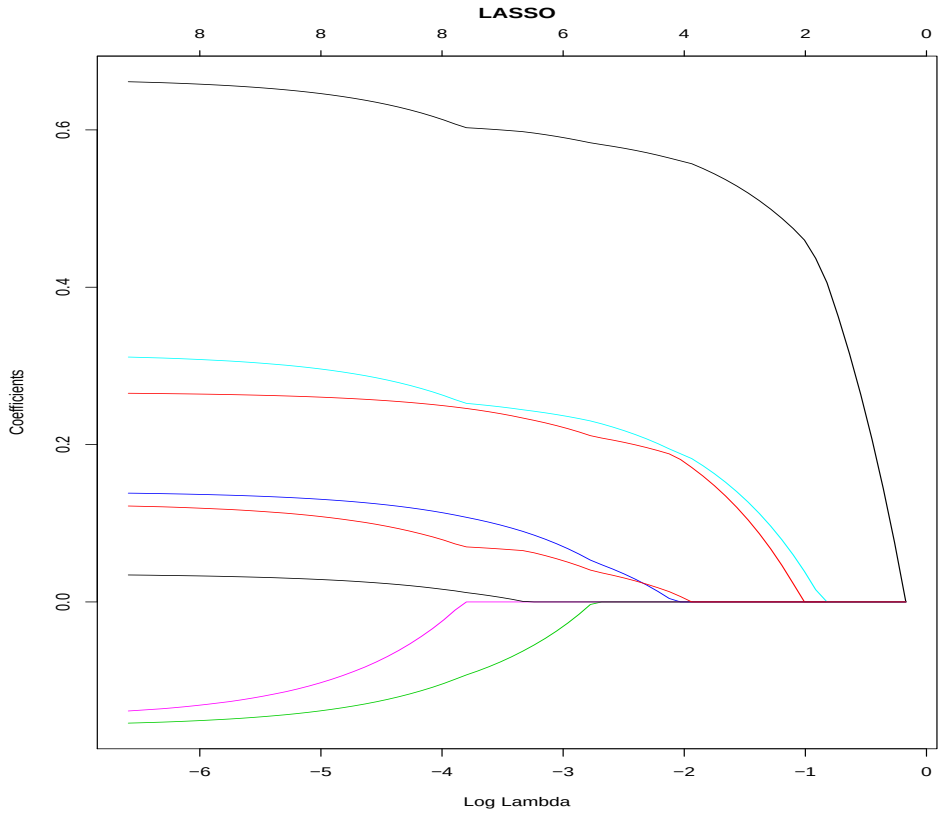


Figura A.2: Gráfica de los coeficientes en función de $\lambda$, en el caso LASSO.

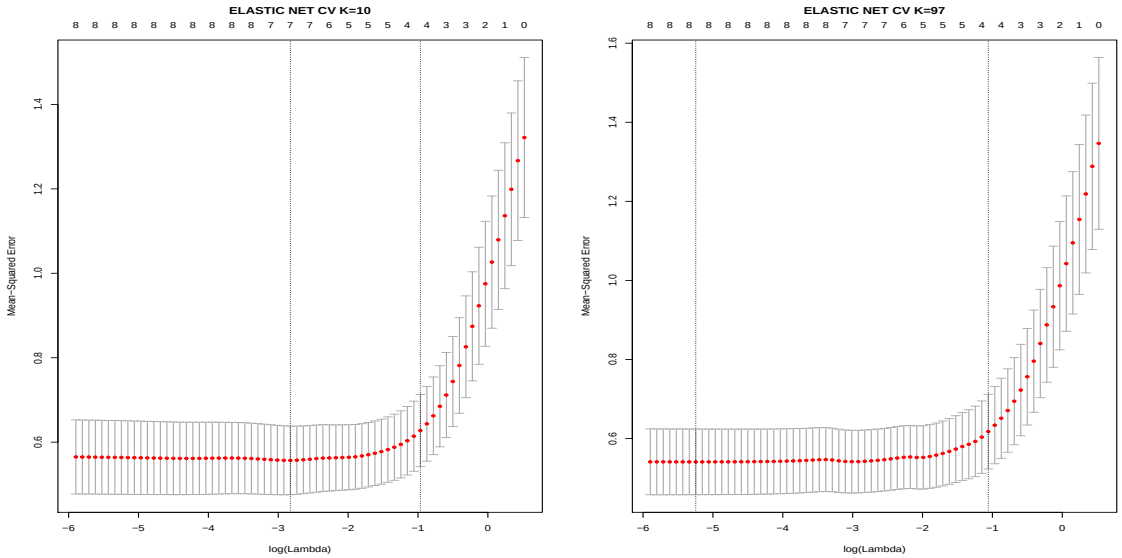


Figura A.3: Error RSME por CV $K = 10$ (izquierda) y LOOCV (derecha), en el caso Elastic Net $\alpha = 0{,}5$.

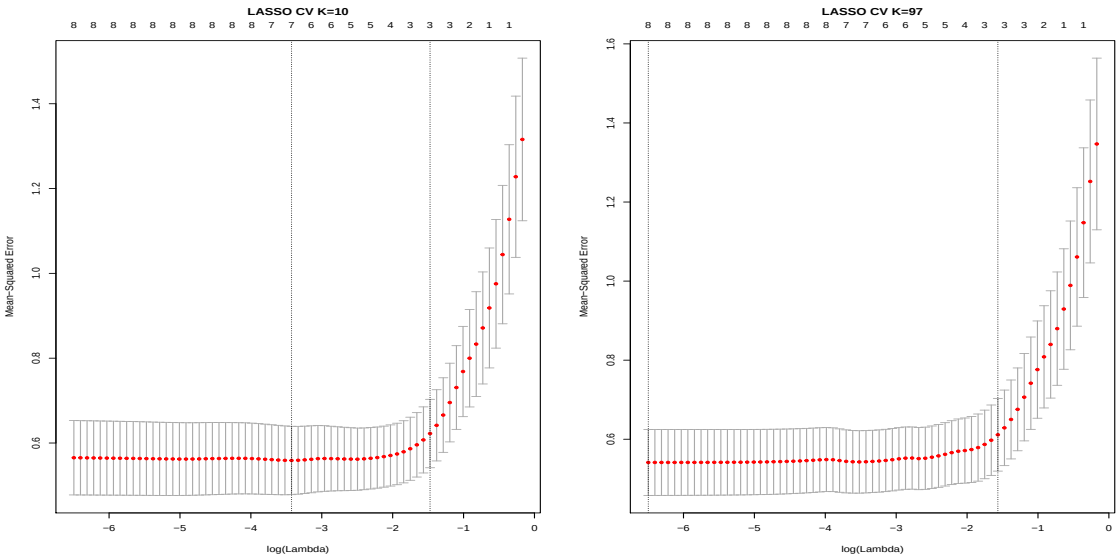


Figura A.4: Error RSME por CV $K = 10$ (izquierda) y LOOCV (derecha), en el caso LASSO.

## A.2. Script de R con las funciones para el conjunto "Prostate"

```
require(glmnet)
require(ggplot2)
require(ElemStatLearn)
require(parcor)
require(pracma)

#Se cargan los datos Prostate de la libreria ElemStatLearn
data(prostate)
summary(prostate)

# Tipificacion de variables predictoras
prostatePreprocess<- preProcess(prostate[,1:8], method = c("center", "scale"), uniqueCut=2)
prostate1Scaled<-predict(prostatePreprocess, newdata=prostate )
summary(prostate1Scaled)

#Definicion de matriz X y vector respuesta
Xpros<-as.matrix(prostate1Scaled[,1:8])
ypros<-prostate1Scaled[,9]

## MINIMOS CUADRADOS ORDINARIOS ##

set.seed(1234)
OLS <- coef(lm(lpsa~., prostate1Scaled[,1:9]))
OLS
summary(lm(lpsa~., prostate1Scaled[,1:9]))
Norm(OLS[-1], p=1)
Norm(OLS[-1], p=2)
```

```
## REGRESION LINEAL PENALIZADA: RIDGE, ELASTIC NET, LASSO ##

#Representacion del vector de coeficientes del modelo frente a lambda (parametro de
penalizacion)
# alpha = 0 (ridge)
# alpha = 0.5 (elastic net)
# alpha = 1 (lasso)

#Ridge
glmnet(Xpros, ypros, family="gaussian", alpha=0)
plot(glmnet(Xpros, ypros, family="gaussian", alpha=0), xvar="lambda" )
title(main="RIDGE",line=2.5)

#Elastic Net
glmnet(Xpros, ypros, family="gaussian", alpha=0.5 )
plot(glmnet(Xpros, ypros, family="gaussian", alpha=0.5 ), xvar="lambda")
title(main="ELASTIC NET 0.5",line=2.5)

#Lasso
glmnet(Xpros, ypros, family="gaussian", alpha=1)
plot(glmnet(Xpros, ypros, family="gaussian", alpha=1), xvar="lambda")
title(main="LASSO",line=2.5)

## SELECCION DE LAMBDA OPTIMO POR VALIDACION CRUZADA PARA K=10

#Ridge

set.seed(1234)
cvGlmnet<- cv.glmnet(x = Xpros, y = ypros, alpha=0)
plot(cvGlmnet)
title(main="RIDGE CV K=10",line=2.5)
rilam0<-cvGlmnet$lambda.min
rilam0 # lambda optimo
cvGlmnet$cvm
min(cvGlmnet$cvm) #RSME

#Modelo para el lambda optimo
glmnet(x = Xpros, y = ypros, alpha=0., lambda=rilam0)
coef0<-coefficients(glmnet(x = Xpros, y = ypros, alpha=0, lambda=rilam0))

#Norma del vector de coeficientes del modelo optimo
beta0<- as.vector(coef0)
beta0
Norm(beta0[-1], p=1)
Norm(beta0[-1], p=2)

#Elastic Net

set.seed(1234)
cvGlmnet<- cv.glmnet(x = Xpros, y = ypros, alpha=0.5)
```

```
plot(cvGlmnet)
title(main="ELASTIC NET CV K=10",line=2.5)
enlam0 <-cvGlmnet$lambda.min
enlam0 #lambda optimo
cvGlmnet$cvm
min(cvGlmnet$cvm) #RSME

#Modelo para el lambda optimo
coef1<- coefficients(glmnet(Xpros, ypros, family="gaussian", alpha=0.5, lambda=enlam0))
coef1

#Norma del vector de coeficientes del modelo optimo
beta1<- as.vector(coef1)
beta1
Norm(beta1[-1], p=1)
Norm(beta1[-1], p=2)

#Lasso

set.seed(1234)
cvGlmnet<- cv.glmnet(x = Xpros, y = ypros, alpha=1)
plot(cvGlmnet)
title(main="LASSO CV K=10",line=2.5)
lalam0<-cvGlmnet$lambda.min
lalam0 #lambda optimo
cvGlmnet$cvm
min(cvGlmnet$cvm) #RSME

#Modelo para el lambda optimo
coef2 <-coefficients(glmnet(Xpros, ypros, family="gaussian", alpha=1., lambda=lalam0 ))
coef2

#Norma del vector de coeficientes del modelo optimo

beta2<- as.vector(coef2)
beta2
Norm(beta2[-1], p=1)
Norm(beta2[-1], p=2)

## SELECCION DE LAMBDA OPTIMO POR VALIDACION CRUZADA PARA K=n=97

#Ridge

set.seed(1234)
cvGlmnet<- cv.glmnet(x = Xpros, y = ypros, alpha=0, nfolds=97)
plot(cvGlmnet)
title(main="RIDGE CV K=97",line=2.5)
rilam1<-cvGlmnet$lambda.min
rilam1 # lambda optimo
cvGlmnet$cvm
min(cvGlmnet$cvm) #RSME
```

```
#Modelo para el lambda optimo
glmnet(x = Xpros, y = ypros, alpha=0., lambda=rilam1)
coef0<-coefficients(glmnet(x = Xpros, y = ypros, alpha=0, lambda=rilam1))
coef0

#Norma del vector de coeficientes del modelo optimo
beta0<- as.vector(coef0)
Norm(beta0[-1], p=1)
Norm(beta0[-1], p=2)

#Elastic Net

set.seed(1234)
cvGlmnet<- cv.glmnet(x = Xpros, y = ypros, alpha=0.5, nfolds=97)
plot(cvGlmnet)
title(main="ELASTIC NET CV K=97",line=2.5)
enlam1<-cvGlmnet$lambda.min
enlam1 #lambda optimo
cvGlmnet$cvm
min(cvGlmnet$cvm) #RSME

#Modelo para el lambda optimo
coef1<- coefficients(glmnet(Xpros, ypros, family="gaussian", alpha=0.5, lambda=enlam1 ))
coef1

#Norma del vector de coeficientes del modelo optimo
beta1<- as.vector(coef1)
Norm(beta1[-1], p=1)
Norm(beta1[-1], p=2)

#Lasso

set.seed(1234)
cvGlmnet<- cv.glmnet(x = Xpros, y = ypros, alpha=1, nfolds=97)
plot(cvGlmnet)
title(main="LASSO CV K=97",line=2.5)
lalam1<-cvGlmnet$lambda.min
lalam1 #lambda optimo
cvGlmnet$cvm
min(cvGlmnet$cvm) #RSME

#Modelo para el lambda optimo
coef2 <-coefficients(glmnet(Xpros, ypros, family="gaussian", alpha=1., lambda=lalam1 ))
coef2

#Norma del vector de coeficientes del modelo optimo
beta2<- as.vector(coef2)
Norm(beta2[-1], p=1)
Norm(beta2[-1], p=2)
```

```
# Representacion de los coeficientes para OLS y los lambdas optimos obtenidos por CV

#Ridge

glmnet(Xpros, ypros, family="gaussian", alpha=0)
plot(glmnet(Xpros, ypros, family="gaussian", alpha=0), xvar="lambda", xlim=c(-3,0))
abline(v=-2.47, col="black")
abline(v=log(rilam0), col="red")
abline(v=log(rilam1), col="blue")
text(-2.47,0.4,labels="OLS",col="black",pos=2,cex=0.8)
text(log(rilam0),0.4,labels="CV K=10",col="red",pos=4,cex=0.8)
text(log(rilam1),0.4,labels="LOOCV",col="blue",pos=4,cex=0.8)
title(main="RIDGE",line=2.5)

#Lasso

glmnet(Xpros, ypros, family="gaussian", alpha=1)
plot(glmnet(Xpros, ypros, family="gaussian", alpha=1 ), xvar="lambda", xlim=c(-7,-3))
abline(v=-6.6, col="black")
abline(v=log(lalam0), col="red")
abline(v=log(lalam1), col="blue")
text(-6.6,0.45,labels="OLS",col="black",pos=2,cex=0.8)
text(log(lalam0),0.45,labels="CV K=10",col="red",pos=2,cex=0.8)
text(log(lalam1),0.45,labels="LOOCV",col="blue",pos=4,cex=0.8)
title(main="LASSO",line=2.5)
```

## A.3. Script de R con las funciones para el conjunto “Ames Housing”

```
require(glmnet)
require(ggplot2)
require(ElemStatLearn)
require(parcor)
require(pracma)
require(KRLS)
require(caret)

# Cargamos los conjuntos de datos, AH2 conjunto sin datos ausentes.
head(load("AmesHousing.RData"))
head(load("AH2.RData"))

summary(AH2)

# Dividimos las variables explicativas entre numericas y categoricas
numer <- c(5,6,21,22,28,36,38:40,45:54,56,58,61,63,64,68:73,77:79)
factor<- c(3,4,7:20,23:27,29:35,37,41:44,55,57,59,60,62,65:67,74:76,80,81,82)

# Tipificamos las variables numericas
```

```
AHnumer <- subset(AH2, select = numer)
AmesPre<- preProcess(AH2[,numer], method = c("center", "scale"))
Xnumer<-predict(AmesPre, newdata=AHnumer)
summary(Xnumer)

# Convertimos las variables categoricas en variables dummies

AHfactor <- subset(AH2, select = factor)
Xfactor<-model.matrix(SalePrice~.-1, data=AHfactor)

yames<-AH2[,82]

Xames<- cbind(Xnumer,Xfactor, yames)

#Nueva matriz X quitando variables que tienen varianza =0
nzv <- nearZeroVar(Xames, saveMetrics= TRUE)
nzv$zeroVar
Xames2<- Xames[, -(1:ncol(Xames))[nzv$zeroVar]]


# Definimos matriz excluyendo la variable yames
matriz <- as.matrix(Xames2[,-ncol(Xames2)])
matriz
matriz2<- as.data.frame(matriz)
matriz2


## MINIMOS CUADRADOS ORDINARIOS ##

OLS <- lm(yames~., matriz2, na.action = na.omit)
OLS
summary(OLS)

#Norma del vector de coeficientes del modelo optimo
beta<-  na.remove(as.vector(coef(OLS)))
Norm(beta[-1], p=1)

# Grafica de datos ajustados
predict.lm(OLS,newdata=matriz2)
yhat<- as.vector(predict.lm(OLS,newdata=matriz2))
yhat
plot(yhat, yames,xlab= "SalePrice ajustados", ylab="SalePrice")
abline(a=0, b=1, col="red")
title(main="OLS")


## Mal condicionamiento mediante el calculo de valores propios

#valores propios
m <- t(matriz)%*%matriz  #matriz 275x275
em<- eigen(m, symmetric=TRUE, only.values = TRUE)
```

```
plot(log(em$values))


# VALIDACION CRUZADA CON K=10 PARA RIDGE Y LASSO

# RIDGE

set.seed(1234)
cvGlmnet<- cv.glmnet(x = matriz, y = yames, alpha=0)
cvGlmnet
plot(cvGlmnet)
title(main="RIDGE",line=2.5)
cvGlmnet0<-cvGlmnet$lambda.min
cvGlmnet0 # landa optimo
cvGlmnet$cvm
min(cvGlmnet$cvm) #RSME

#modelo ridge con el landa optimo
ridge.model<- glmnet(x = matriz, y = yames, alpha=0, lambda=cvGlmnet0)
coef0<-coefficients(ridge.model)
coef0

#Norma del vector de coeficientes del modelo optimo
beta0<- as.vector(coef0)
Norm(beta0[-1], p=1)

#Rcuadrado
yhat0<- predict(ridge.model, newx=matriz)
Rsquared<- var(yhat0)/var(yames)
Rsquared


#LASSO

set.seed(1234)
cvGlmnet<- cv.glmnet(x = matriz, y = yames, alpha=1)
plot(cvGlmnet)
title(main="LASSO",line=2.5)
cvGlmnet2<-cvGlmnet$lambda.min
cvGlmnet2 #lambda optimo
cvGlmnet$cvm
min(cvGlmnet$cvm) #RSME


#modelo ridge con el landa optimo
lasso.model <-glmnet(x = matriz, y = yames, alpha=1, lambda=cvGlmnet2)
coef2<-coefficients(glmnet(x = matriz, y = yames, alpha=1, lambda=cvGlmnet2))
coef2
summary(lasso.model)

#Norma del vector de coeficientes del modelo optimo
```

```
beta2<- as.vector(coef2)
Norm(beta2[-1], p=1)

#Rcuadrado
yhat2<- predict(lasso.model, newx=matriz)
Rsquared<- var(yhat2)/var(yames)
Rsquared


# REGULARIZACION MEDIANTE KERNEL GAUSSIANO

memory.limit(size=15000)
krls.model<- krls(X = matriz, y = yames, whichkernel = "gaussian",print.level = 2,
tol=NULL,eigtrunc=10^(-6), derivative=FALSE, vcov=FALSE)
krls.model

predict(krls.model,newdata=matriz)
plot(predict(krls.model,newdata=matriz)$fit,yames)
abline(a=0, b=1, col="red")
title(main="KRLS")
```