FACULTAD DE MATEMÁTICAS

DEPARTAMENTO DE ESTADÍSTICA E INVESTIGACIÓN OPERATIVA

Trabajo Fin de Grado

MODELOS ADITIVOS GENERALIZADOS

Pablo Aguilar Barreiro

Grado en Estadística

Junio 2019

Dirigido por:

Joaquín García de las Heras

José Luis Pino Mejías

Índice general

Resumen

Este trabajo se centra en el estudio y aplicación del Modelo Aditivo Generalizado el cual intenta aprovechar las ventajas del enfoque paramétrico y no paramétrico de otros modelos. Por esta razón, primero realizamos un breve resumen del Modelo Lineal Generalizado. A continuación, introducimos el Modelo Aditivo Generalizado y su estructura, caracterizada por el uso de funciones suaves sobre las variables explicativas que permite relaciones no lineales entre éstas y la variable objetivo. Con el fin de obtener dichas funciones suaves, recopilamos diferentes técnicas, entre las que destaca el uso de los splines de regresión cúbicos, haciendo hincapié en las diferencias de las mismas. Seguidamente, introducimos el concepto de criptomoneda y comentamos los problemas que presenta su modelización estadística. Por último, haciendo uso del software estadístico R, comprobamos si la implementación de las técnicas descritas pueden ser beneficiosas a la hora de tomar decisiones relacionadas con la comercialización de las criptomonedas.

Abstract

This work is focused on the study and application of the Generalized Additive Model which tries to take advantage of the parametric and non-parametric approach of other models. For this reason, we first make a brief summary of the Generalized Linear Model. Next, we introduce the Generalized Additive Model and its structure, characterized by the use of smooth functions on the independent variables that allow non-linear relationships between them and the response variable. In order to obtain these smooth functions, we compile different techniques, among which the cubic regression splines stand out, emphasizing their differences. Next, we introduce the concept of cryptocurrency and discuss the problems presented by its statistical modeling. Finally, using the statistical software R, we check whether the implementation of the techniques described can be beneficial when making decisions related to the commercialization of cryptocurrencies.

Introducción

Este trabajo se ha estructurado en cuatro capítulos. En el primero, introducimos el concepto de modelo estadístico y, a continuación, describimos la estructura matemática que puede adquirir dicho modelo, así como los dos enfoques principales para su estudio: modelos paramétricos y no paramétricos, centrándonos en las ventajas y desventajas de cada uno. Más adelante, recopilamos algunos resultados conocidos del Modelo Lineal Generalizado, que forma parte del enfoque paramétrico, y del Modelo Aditivo Generalizado.

En el segundo capítulo, profundizamos sobre el Modelo Aditivo Generalizado y su estructura, caracterizada por el uso de funciones suaves sobre las variables explicativas que permite relaciones no lineales entre la variable objetivo y las explicativas. Para llevar a cabo el estudio de este modelo, es necesario obtener dichas funciones suaves. Con este fin, a lo largo de este capítulo, introduciremos diferentes técnicas, entre las que destaca el uso de los splines de regresión cúbicos. Seguidamente, se recogen una serie de técnicas que permiten el estudio de la posible interacción entre las variables explicativas.

En el siguiente capítulo, introducimos diferentes conceptos económicos que terminarán derivando en el estudio de la criptomoneda. Realizamos un breve comentario sobre su evolución desde que fue ideada y enumeramos las características que han convertido a la criptomoneda en un bien tan cotizado en los últimos años. No obstante, las criptomonedsa se caracterizan por la volatilidad en su precio, lo que la convierte en una inversión de riesgo para aquellas personas que decidan comercializar con ella. Seguidamente, comentamos las causas de dicha volatilidad, entre las que se encuentran la especulación y las casas de cambio.

En el último capítulo, para comprobar si el tratamiento de la criptomoneda, desde el punto de vista práctico, puede servirse del Modelo Aditivo Generalizado y de las técnicas descritas en los capítulos anteriores, realizamos un estudio sobre datos reales de la criptomoneda Ethereum y de tres casas de cambio usando el software R. Por un lado, el objetivo consistirá en analizar los distintos modelos obtenidos mediante las diferentes técnicas descritas anteriormente y, por otro lado, será predecir a traves de qué casa de cambio conviene más realizar una transacción de una determinada cantidad de criptomonedas.

Para finalizar, se recoge una revisión bibliográfica con aquellos libros, artículos, páginas webs, etc., usadas para la realización de este trabajo.

Capítulo 1

Modelo Lineal Generalizado

1.1. Introducción

Uno de los objetivos fundamentales de la Estadística es buscar, describir y predecir relaciones entre variables que describen fenómenos del mundo real. Una de las técnicas para alcanzar dicho objetivo es buscar una función o ecuación matemática que relacione las variables en estudio. Esto se traduce en construir un modelo estadístico que se encarge de describir la situación real.

Mediante la modelización estadística se pretende determinar si existe o no relación causal entre una **variable objetivo**, Y, y una serie de p **variables explicativas** $X_1, X_2, \ldots, X_p$. También se busca determinar cuál será el impacto sufrido por la variable objetivo o dependiente ante un cambio en las variables explicativas.

Esta relación se puede expresar matemáticamente como

$$Y = f(X_1, X_2, \ldots, X_p) + \epsilon \tag{1.1}$$

siendo f una función desconocida y definida sobre las variables explicativas que trata de modelizar la mejor relación posible entre Y y $X_1, X_2, \ldots, X_p$ y ϵ es el error aleatorio, el cual es independiente de las variables explicativas y, por hipótesis, su media es cero.

Al ser esta función f desconocida, será necesario aplicar unas determinadas técnicas para ajustar lo mejor posible dicha función y que la relación que se establezca entre Y y $X_1, X_2, \ldots, X_p$ sea lo más parecida posible a la realidad.

El objetivo principal de este trabajo será describir una serie de técnicas para llevar a cabo dicho ajuste, mostrar las diferencias entre ellas y aplicarlas a un fenómeno del mundo real que puede beneficiarse del estudio y aplicación de las mismas.

Sea cual sea el método a seguir partiremos del conjunto de entrenamiento para realizar la búsqueda de la función f. Este conjunto se obtiene a partir de una muestra aleatoria simple de n individuos sobre los que se realiza la medición de las variables involucradas en el estudio. Por tanto, el conjunto de entrenamiento está formado por $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, siendo $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ para $i = 1, 2, \ldots, n$.

Donde x_{ij} es el valor de la j-ésima variable explicativa para la i-esima observación e y_i el valor de la variable respuesta Y para la i-ésima observación, siendo $i = 1, 2, \ldots, n$ y $j = 1, 2, \ldots, p$.

Estas técnicas se pueden englobar en dos grupos: **métodos parámetricos** y **métodos no paramétricos**, tal y como se recoge en James, 2014, p.21. A continuación, se presentarán ambos enfoques haciendo hincapié en sus diferencias.

1.2. Modelos paramétricos

El modelo paramétrico sugiere que la función f tiene una forma funcional determinada. El planteamiento de los modelos paramétricos se puede dividir en dos pasos:

1. Se realiza una suposición acerca de la forma funcional de f. Por ejemplo, f se puede expresar como una combinación lineal de las variables explicativas y una serie de coeficientes $\beta_0, \beta_1, \ldots, \beta_p$

$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p \tag{1.2}$$

   Al hacer esta suposición, se realiza una restricción sobre la forma funcional de f, pero el problema de la búsqueda de f se simplifica considerablemente pues, en vez de tener que estimar una función p-dimensional $f(X_1, X_2, \ldots, X_p)$, se estiman los $p + 1$ coeficientes $\beta_0, \beta_1, \ldots, \beta_p$.

2. Tras seleccionar el modelo, se hace uso del conjunto entrenamiento para ajustarlo. En el caso del modelo lineal anterior (1.2), se estiman los p+1 coeficientes $\beta_0, \beta_1, \ldots, \beta_p$ de manera que

$$Y \approx \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \cdots + \hat{\beta}_p X_p$$

   La estimación de estos $p+1$ coeficientes se realiza mediante mínimos cuadrados.

Este enfoque se denota **paramétrico** pues al suponer una determinada forma funcional de f, el problema de ajustar una función p-dimensional se reduce a estimar una serie de parámetros, como $\beta_0, \beta_1, \ldots, \beta_p$ en (1.2), lo cual es mucho más sencillo.

A continuación, realizamos una breve introducción del **Modelo Lineal Generalizado**, que forma parte del enfoque paramétrico y que servirá como base para un modelo más general.

1.2.1. Modelo Lineal Generalizado

El planteamiento de métodos estadísticos en los que se trata de explicar el comportamiento de una o varias variables objetivos, a través de un conjunto de variables explicativas, requiere la elección de un modelo que describa la estructura de la relación entre las variables.

Generalmente, el modelo más utilizado es del tipo lineal en el que se modeliza la variable o variables objetivos, o alguna característica de ellas, a través de una combinación lineal de las variables explicativas.

El modelo lineal clásico consiste en expresar la esperanza condicionada de la variable objetivo Y como combinación lineal de las variables explicativas $X_1, X_2, \ldots, X_p$:

$$\begin{gathered} E[Y|X = x_i] = \mu_i \\ \mu_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon_i \end{gathered} \tag{1.3}$$

donde $\beta_0, \beta_1, \ldots, \beta_p$ son parámetros desconocidos y ϵ_i error aleatorio i.i.d $N(0, \sigma^2)$, $i = 1, 2, \ldots, n$.

Por tanto, el modelo lineal clásico consiste en expresar la esperanza condicionada de la variable objetivo como combinación lineal de las variables explicativas bajo hipótesis de normalidad y homocedasticidad. Esta modelización lineal clásica se puede extender a una familia más general (Nelder y Wedderburn, 1972) y ampliada por (McCullagh y Nelder, 1989) conocida como **Modelo Lineal Generalizado**.

Para especificar totalmente este modelo necesitamos tres componentes: distribucional, sistemática y estructural, tal y como se vio en la asignatura de Modelos Lineales. A continuación, comentaremos cada una de ellas.

Componente distribucional

En el Modelo Lineal Generalizado se asume que la distribución de la variable Y pertenece a la **Familia Exponencial**.

Se dice que una variable aleatoria pertenece a la Familia Exponencial si su función de probabilidad o de densidad, ya sea discreta o continua, presenta la forma:

$$P(Y = y;\, \theta,\, \phi) = f_Y(y;\, \theta,\, \phi) = exp\left\{ \frac{y\theta - b(\theta)}{\phi} + c(y, \phi) \right\}$$

siendo θ el parámetro natural, ϕ el parámetro de escala o dispersión y $b(.)$ y $c(.)$ son funciones específicas de cada elemento de la familia.

Para toda variable perteneciente a esta familia se verifica

$$E(Y) = \mu = \frac{\partial}{\partial \theta} b(\theta) \quad ; \quad V(Y) = \sigma^2 = \frac{\partial^2}{\partial \theta^2} b(\theta)\phi$$

En la siguiente tabla se exponen los elementos principales que caracterizan a algunas de las distribuciones más utilizadas de la familia exponencial.

Distribuciones	$\theta(\mu)$	$b(\theta)$	ϕ	μ	σ^2
Bernoulli $Be(\pi)$	$ln(\frac{\pi}{1-\pi})$	$ln(1+e^{\theta})$	1	$\pi = \frac{e^{\theta}}{1+e^{\theta}}$	$\pi(1-\pi)$
Poisson $Po(\lambda)$	$ln(\lambda)$	e^{θ}	1	$\lambda = e^{\theta}$	λ
Normal $N(\mu, \sigma^2)$	μ	$\frac{\theta^2}{2}$	σ^2	θ	ϕ
Exponencial $Exp(\lambda)$	μ	$ln(\theta)$	-1	$\frac{1}{\theta}$	$\frac{1}{\lambda^2}$
Gamma $Ga(p, \lambda)$	μ	$pln(\theta)$	-1	$\frac{p}{\theta}$	$\frac{p}{\theta^2}$

Componente sistemática

La componente sistemática recoge la variabilidad de la variable objetivo Y expresada mediante una combinación lineal de las variables explicativas $X_1, X_2, \ldots, X_p$ junto con los correspondientes parámetros β_0, $\beta_1, \ldots,$ β_p. Esta componente, también denominada predictor lineal, se representa mediante $z_i^t\beta$, donde $z_i^t = (1, x_{i1}, x_{i2}, \ldots, x_{ip})$ y $\beta^t = (\beta_0, \beta_1, \ldots, \beta_p)$ con $i = 1, 2, , \ldots, n$.

Componente estructural

En el modelo de regresión lineal el valor esperado de la variable objetivo Y se expresa mediante una combinación lineal de las variables explicativas, pero a la hora de llevarlo a la práctica esta relación no es adecuada, por lo que es necesario incluir una función que relacione dicho valor esperado con las variables explicativas. Esta función se denomina función enlace o vínculo y se representa mediante $g(\mu_i) = z_i^t\beta$.

La función inversa de la función enlace se denota por h y verifica

$$\mu_i = g^{-1}(z_i^t\beta) = h(z_i^t\beta)$$

La elección de la función enlace no siempre resulta obvia pues pueden existir diferentes funciones enlace que se puedan aplicar a un problema particular. Por tanto, es crucial elegir una función enlace que facilite la interpretación del modelo obtenido.

Para cada elemento de la familia exponencial existe una función enlace denominada función enlace canónica o natural que consiste en relacionar el parámetro natural directamente con el predictor lineal.

$$\theta_i = \theta(\mu_i) = z_i^t\beta \qquad g(\mu_i) = \theta(\mu_i)$$

A continuación, se especifican las funciones canónicas para algunas distribuciones de la Familia Exponencial.

Distribuciones	$\theta(\mu_i)$	Modelo
Bernoulli $Be(\pi)$	$ln(\frac{\pi}{1-\pi})$	$ln(\frac{\pi}{1-\pi}) = z_i^t\beta$
Poisson $Po(\lambda)$	$ln(\mu_i)$	$ln(\mu_i) = z_i^t\beta$
Normal $N(\mu, \sigma^2)$	μ_i	$\mu_i = z_i^t\beta$

1.3. Modelos no paramétricos

La principal desventaja de basarse en el enfoque paramétrico es que, normalmente, el modelo escogido no se asemeja a la verdadera forma funcional de f y, por tanto, todo estudio o análisis de datos que se lleve a cabo a partir de este ajuste conducirá a errores.

Para resolver este problema se puede optar por elegir modelos más flexibles que permitan trabajar con muchas más formas funcionales para la función f. Los modelos **no paramétricos** se basan en esta idea.

Al contrario que los métodos paramétricos, los no paramétricos no realizan ninguna suposición sobre la forma funcional de f antes de ajustarla (James, 2014, p.23). Al permitir que la forma funcional f pueda tomar cualquier forma dentro del enorme espacio de funciones posibles, se ofrece mucha más **flexibilidad** que la ofrecida por los métodos paramétricos.

Algunas de las técnicas que se usan para la búsqueda de f son las bases de funciones polinómicas, splines cúbicos naturales o splines de suavizado. Sobre estas técnicas hablaremos de manera más extensa en el capítulo siguiente.

Como se ha comentado anteriormente, hacer uso de los métodos paramétricos puede conllevar a que la forma funcional tomada para f no sea la correcta y esto conduciría a errores a la hora de realizar predicciones mediante el modelo obtenido. Para resolver este problema se puede optar por elegir modelos más flexibles que permitan trabajar con muchas más formas funcionales para la función f, pero habría que estimar un mayor número de parámetros.

Esto conllevaría a un fenómeno que se conoce como **sobreajuste** (James, 2014, p.22), por el cual, tanto los modelos paramétricos como los no paramétricos, se pueden ver afectados. Este hecho consiste en que el ajuste obtenido de la función f queda muy acoplado a características específicas de los datos que forman parte del conjunto de entrenamiento, usados para estimar los parámetros, y puede omitir información sistemática de relevancia. Por tanto, no se obtendrán predicciones adecuadas para observaciones que se encuentren fuera del rango del conjunto de entrenamiento.

Como hemos visto anteriormente, los métodos paramétricos reducen el problema de estimar la función f a estimar una serie de parámetros $\beta_0, \beta_1, \ldots, \beta_p$. Esto no ocurre con los métodos no paramétricos y, por tanto, se necesita un mayor número de observaciones (mayor que el necesario para los métodos paramétricos) para obtener un ajuste de f óptimo.

Algunas de las ventajas de los métodos paramétricos sobre los no paramétricos es que las técnicas de inferencia aplicadas son más potentes y se obtienen mejores interpretaciones sobre la relación entre la variable respuesta y las variables explicativas. La elección de la forma funcional de f conduce a un modelo determinado cuyos resultados deben poder ser interpretados. Por ejemplo, en el caso del modelo lineal clásico, (1.3), comprender la relación entre las variables explicativas $X_1, X_2, \ldots, X_p$ e Y no supone

mayor dificultad.

En cambio, al utilizar modelos no paramétricos se pueden llegar a obtener ajustes de f tan enrevesados que la interpretación de los resultados pasa a ser un problema y cuesta entender la relación entre cada variable explicativa con la variable respuesta.

Es aquí donde entra en juego lo que se conoce como **Modelo Aditivo Generalizado**. Este se basa en una mezcla de elementos de los métodos paramétricos y no paramétricos. Se considera una extensión del Modelo Lineal Generalizado, 1.2.1, que permite que las relaciones entre $X_1, X_2, \ldots, X_p$ e Y no deban ser lineales mientras se mantiene la aditividad. Sobre este modelo se profundizará más en el siguiente capítulo.

Capítulo 2

Modelo Aditivo Generalizado

2.1. Introducción

El **Modelo Aditivo Generalizado** es una extensión del Modelo Lineal Generalizado que fue introducido por Trevor Hastie and Robert Tibshirani en 1986 (Hastie y Tibshirani, 1986).

En la sección 1.2.1 hemos visto que el Modelo Lineal Generalizado asume que la influencia de las variables explicativas sobre la variable respuesta o una característica de la misma es de forma lineal, es decir,

$$g(\mu_i) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon_i$$

Desde otro punto de vista, puede ocurrir que el efecto de las variables explicativas sobre la variable dependiente tenga forma desconocida y, en ese caso, la estructura del modelo puede representarse de la siguiente forma:

$$g(\mu_i) = \mathbf{X}_i^* \boldsymbol{\theta} + f_1(x_{1i}) + f_2(x_{2i}) + f_3(x_{3i}) + \ldots, \quad i = 1, 2, , \ldots, n \tag{2.1}$$

donde $g()$ es la función enlace, como vimos en el apartado 1.2.1 , $\mu_i = E(Y_i)$ con Y_i variable respuesta e $Y_i \sim$ Distribución de la Familia Exponencial, $\mathbf{X}_i^*$ es la i-ésima fila de la matriz de diseño correspondiente a las covariables que definen las componentes paramétricas del modelo, f_j, $j = 1, 2, 3 \ldots$ son las funciones suaves[1] y $\boldsymbol{\theta}$ es el vector de coeficientes de regresión (Wood, 2006, p.121).

A diferencia del Modelo Lineal Generalizado, que restringe la relación entre la variable respuesta y la explicativa a la forma lineal, el Modelo Aditivo Generalizado permite que las funciones $f_j(.)$ puedan tomar cualquiera forma funcional, lo que proporciona más información acerca de la relación entre la variable explicativa y la variable objetivo.

El hecho de presentar esta ventaja sobre el Modelo Lineal Generalizado conlleva plantearse dos cuestiones que trataremos en este trabajo:

[1] Una función suave es aquella que tiene derivadas de todos los órdenes.

- Cómo construir estas funciones arbitrarias.
- Cómo suavizar las funciones obtenidas.

A lo largo de este capítulo presentaremos distintas formas de construir las funciones f_j. Este capítulo se centrará principalmente en representar estas funciones arbitrarias a partir de los splines cúbicos de regresión ya que son los más usados, pero también se comentarán otras técnicas como el uso de base de funciones polinómicas, splines de suavizado, B-Splines, P-Splines, Splines de regresión Thin Plate y tensores.

2.2. Bases de funciones

Con el objetivo de aclarar los conceptos que se desarrollarán a lo largo de este apartado, consideramos que, desde el punto de vista del modelo lineal clásico, es mejor comenzar teniendo en cuenta una sola variable explicativa X.

Formalmente se busca una función f que satisfaga:

$$y_i = f(x_i) + \epsilon_i, \quad i = 1, 2, , \ldots, n \tag{2.2}$$

donde y_i es la variable respuesta, x_i, la variable explicativa, f, una función suave y ϵ_i es el error aletario independiente de la variable explicativa e idénticamente distribuido según una distribución $N(0, \sigma^2)$ para $i = 1, 2, , \ldots, n$.

Con el objetivo de aplicar los métodos estadísticos usados hasta este momento, es necesario construir f de manera que (2.2) se convierta en un modelo lineal. Esto se puede conseguir definiendo una base de funciones b_j conocidas de dimensión q de la cual f (o una aproximación cercana) forma parte.

Para definir esta base de funciones, Wood, 2006, p.122, propone tomar funciones básicas y, mediante una combinación de éstas y un vector de parámetros β, se lleva a cabo la búsqueda de f que puede ser representada como sigue:

$$f(x) = \sum_{j=1}^{q} b_j(x)\beta_j \tag{2.3}$$

Al llevar a cabo la sustitución de (2.3) en (2.2), se obtiene el siguiente modelo lineal:

$$y_i = \sum_{j=1}^{q} b_j(x_i)\beta_j + \epsilon_i, \quad i = 1, 2, , \ldots, n$$

A continuación se expondrán distintas bases de funciones para llevar a cabo la construcción de f.

2.2.1. Base polinómica

Si se hace uso de la base polinómica, se ajusta la función f mediante un polinomio de grado considerable a lo largo de todo el recorrido de X.

En general, el modelo que se obtiene haciendo uso de una función polinómica de grado d es

$$y_i = \beta_1 + \beta_2 x_i + \beta_3 x_i^2 + \beta_4 x_i^3 + \cdots + \beta_{d+1} x_i^d + \epsilon_i, \quad i = 1, 2, , \ldots, n \tag{2.4}$$

Esta técnica se conoce como **regresión polinómica** cuyas funciones básicas son 1, x_i, x_i^2, x_i^3, $\ldots$, x_i^d y los coeficientes β_j se pueden estimar mediante mínimos cuadrados (James, 2014, p.266).

Para un grado alto del polinomio, este modelo, al representarlo, adopta una curva extremadamente no lineal. Por lo que, desde el punto de vista práctico, no se toma un grado del polinomio mayor que 3 o 4 ya que, para un grado alto, la representación del modelo se convierte en una curva muy flexible que puede adoptar comportamientos muy extraños, en especial en los extremos.

Supongamos que f es un polinomio de orden 4. Por tanto, se pueden tomar las siguientes funciones básicas de orden menor o igual que 4:

$$b_1(x) = 1, \quad b_2(x) = x, \quad b_3(x) = x^2, \quad b_4(x) = x^3 \quad y \quad b_5(x) = x^4$$

Por tanto, (2.3) quedaría

$$f(x) = \beta_1 + \beta_2 x + \beta_3 x^2 + \beta_4 x^3 + \beta_5 x^4$$

y (2.2) pasaría a ser

$$y_i = \beta_1 + \beta_2 x_i + \beta_3 x_i^2 + \beta_4 x_i^3 + \beta_5 x_i^4 + \epsilon_i, \quad i = 1, 2, , \ldots, n$$

2.2.2. Splines de regresión

Como hemos comentado anteriormente, la regresión polinómica ajusta la función f mediante un polinomio de grado considerable a lo largo de todo el rango de X. En cambio, tal y como se recoge en James, 2014, p.271, se introduce una alternativa que ofrece la posibilidad de realizar el ajuste a partir de diferentes polinomios de menor grado en distintas regiones del rango de X.

Por ejemplo, un polinomio cúbico definido a trozos puede llevar a cabo el ajuste de un modelo de regresión cúbico. La forma de este modelo es la siguiente

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \epsilon_i \tag{2.5}$$

donde los coeficientes β_0, β_1, β_2, β_3 son distintos en diferentes regiones del rango de X. Los puntos donde los coeficientes cambian de valor son conocidos como **nodos**.

Un polinomio cúbico definido a trozos sin ningún nodo es un polinomio cúbico estándar, como (2.5). Un polinomio cúbico definido a trozos con un nodo en el punto x^* se representa de la siguiente forma:

$$y_i = \begin{cases} \beta_{01} + \beta_{11} x_i + \beta_{21} x_i^2 + \beta_{31} x_i^3 + \epsilon_i & \text{si } x_i < x^* \\ \beta_{02} + \beta_{12} x_i + \beta_{22} x_i^2 + \beta_{32} x_i^3 + \epsilon_i & \text{si } x_i \geq x^* \end{cases}$$

Por lo que se ajustan dos funciones polinómicas distintas a los datos, una para las observaciones tales que $x_i < x^*$ y otra para aquellas observaciones tales que $x_i \geq x^*$.

El primer polinomio tiene como coeficientes β_{01}, β_{11}, β_{21}, β_{31} y el segundo β_{02}, β_{12}, β_{22}, β_{32}. Para ambos casos, los coeficientes se pueden estimar mediante mínimos cuadrados.

Mediante esta técnica, conocida como **Splines de regresión**, se divide el rango de X en distintas regiones, tal y como se observa en la figura 2.1. En cada región se ajusta una función polinómica sobre las observaciones correspondientes. Para asegurar que los polinomios se unan de manera suave en los extremos de cada región, es decir, en los nodos, se impone que los polinomios deban ser continuos en los nodos y la primera y segunda derivada deban ser continuas en cada nodo.

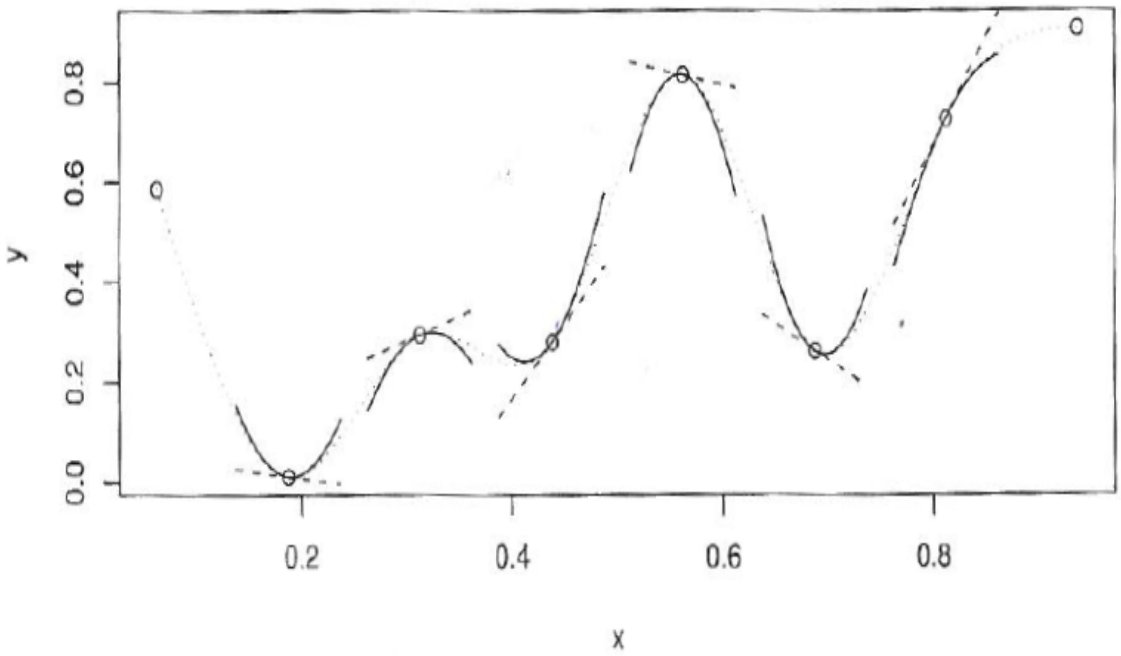

Figura 2.1: *Un spline cúbico queda representado por una curva construida mediante distintos polinomios cúbicos unidos de manera que la curva sea continua hasta su segunda derivada. Los puntos de unión (∘) son los nodos del spline. Cada polinomio tiene distintos coeficientes, pero en los nodos se igualarán los valores y las primeras dos derivadas con respecto a las zonas vecinas (Wood, 2006, p.124).*

Si el rango de X es dividido en suficientes regiones, se produce un ajuste extremadamente suave. En general, cuantos más nodos se usen, más flexible será el ajuste

realizado sobre f. Si se sitúan k nodos a lo largo del rango de X, se ajustan $k+1$ polinomios cúbicos.

Un spline de grado d es aquel definido mediante un polinomio de grado d definido a trozos cuyas primeras $d-1$ derivadas son continuas en cada nodo. Desde el punto de vista práctico, los splines más usados son los de grado 3, conocidos como **splines cúbicos**.

Bases de Splines Cúbicos

Tal y como se recoge en Wood, 2006, p.126, una de las bases de splines cúbicos más usadas está basada en $q-2$ nodos interiores $x_i^*, i = 1, \ldots, q-2$ y está generada por:

$$b_1(x) = 1, \quad b_2(x) = x, \quad b_{i+2} = R(x, x_i^*), \quad i = 1, \ldots, q-2,$$

siendo

$$R(x,z) = \frac{\left[\left(z-\frac{1}{2}\right)^2 - \frac{1}{12}\right]\left[\left(x-\frac{1}{2}\right)^2 - \frac{1}{12}\right]}{4} - \frac{\left[\left(|x-z|-\frac{1}{2}\right)^4 - \frac{1}{2}\left(|x-z|-\frac{1}{2}\right)^2 + \frac{7}{240}\right]}{24}$$

Haciendo uso de esta base para ajustar f se consigue que (2.2) se convierta en un modelo lineal $y = X\beta + \epsilon$, donde la i-ésima fila de la matriz de diseño es

$$X_i = \begin{bmatrix} 1 & x_i & R(x_i, x_1^*) & R(x_i, x_2^*) & \ldots & R(x_i, x_{q-2}^*) \end{bmatrix}$$

y, por tanto, los parámetros desconocidos pueden ser estimados mediante mínimos cuadrados.

Otra de las bases de funciones más usadas para representar splines cúbicos, propuesta por James, 2014, p.273, está basada en k nodos, $x_1^*, \ldots, x_k^*$, y está generada por:

$$b_1(x) = x, \quad b_2(x) = x^2, \quad b_3(x) = x^3,$$

$$b_4(x) = h(x, x_1^*), \quad b_5(x) = h(x, x_2^*), \quad b_6(x) = h(x, x_3^*), \quad \ldots \quad , b_{k+3}(x) = h(x, x_k^*),$$

donde

$$h(x, x_i^*) = (x - x_i^*)_+^3 = \begin{cases} (x - x_i^*)^3 & \text{si } x > x_i^* \\ 0 & \text{en caso contrario} \end{cases}$$

y, por tanto, (2.2) quedaría como

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_4 h(x, x_1^*) + \beta_5 h(x, x_2^*) \cdots + \beta_{k+3} h(x, x_k^*) + \epsilon_i$$

y los parámetros desconocidos pueden ser estimados al igual que para la base anterior.

Splines Naturales

Una deventaja de los splines de regresión es que pueden presentar una alta varianza en las regiones del recorrido de X que se encuentran a la izquierda del primer nodo definido y a la derecha del último.

Para solucionar esta cuestión, algunos autores, como James, 2014, p.274 o Wood, 2006, p.124, aconsejan que no se imponga la restricción de que las derivadas sean continuas en el primer y último nodo. Con esto se pretende conseguir que la función sea lineal en los extremos y, por tanto, que los nuevos splines produzcan ajustes más estables en los extremos del conjunto de datos. Estos splines reciben el nombre de **Splines Naturales**.

Lugar y Número de Nodos

Cuando se ajusta una función mediante splines, surgen dos cuestiones:

- Cuántos nodos es adecuado utilizar.
- Dónde colocar los nodos a lo largo del rango de X.

El ajuste es mucho más flexible en regiones donde haya muchos nodos pues en esas zonas los coeficientes de los polinomios van cambiando rápidamente (James, 2014, p.274).

Según los diferentes autores, como Durbán, 2018, p.18, existen diferentes criterios para determinar el número de nodos adecuados.

- Definir entre 3 y 7 nodos.
- Cuando el tamaño del conjunto entrenamiento es superior a 100 y X es continua, definir 5 nodos se considera una manera aceptable de asegurar tanto la flexibilidad como la precisión.
- Cuando el tamaño del conjunto entrenamiento es inferior a 30, se opta por definir 3 nodos.
- También se puede optar por probar con diferentes números de nodos y elegir aquel con el que se obtenga un mejor ajuste.

Otra opción para determinar el número de nodos es realizar **Validación Cruzada** (James, 2014, p.275). Esta técnica consiste en extraer el 10 % de observaciones del conjunto de entrenamiento, realizar el ajuste de f mediante splines con k nodos haciendo uso de los datos restantes y la predicción para los datos extraídos al principio. Este proceso se repite hasta que todas las observaciones hayan sido extraídas del conjunto al menos una vez y con distinto número de nodos. Mediante los residuos de los modelos se estudia la bondad del ajuste y se elegirá aquel k con el que se haya obtenido un mejor ajuste.

Tal y como propone Wood, 2006, p.124, existen diferentes criterios a la hora de dónde situar los nodos.

- Colocar más nodos en las zonas donde se crea que la función puede variar de manera más rápida y colocar menos nodos donde parezca más estable.
- Repartir los nodos de manera uniforme a lo largo del rango de X.
- Colocar los nodos en los cuantiles de X.

En resumen, el uso de splines de regresión proporciona, por lo general, mejores resultados que la regresión polinómica. Por un lado, los splines de regresión permiten colocar más nodos en zonas donde f parezca cambiar rápidamente y menos nodos en zonas donde f parezca más estable.

Por otro lado, para obtener un ajuste más flexible, la regresión polinómica aumenta el grado del polinomio, mientras que los splines de regresión aumentan el número de nodos y el grado del polinomio se mantiene fijo. La búsqueda de flexibilidad mediante regresión polinómica produce resultados no deseados en los extremos, mientras que con los splines cúbicos naturales se obtienen ajustes más razonables y estables (James, 2014, p.276).

2.2.3. Splines de suavizado

En el apartado 2.2.2 hemos introducido el concepto de splines de regresión que, con el objetivo de ajustar una función a un conjunto de datos lo más suavemente posible, se crean a partir de la definición de un conjunto de nodos, producen una secuencia de funciones básicas y se usa el método de mínimos cuadrados para estimar los coeficientes del modelo. En este apartado vamos a introducir un enfoque diferente, aunque con el mismo objetivo, que también produce splines.

A la hora de ajustar una curva a un conjunto de observaciones, se quiere encontrar una función f que realice el ajuste de manera adecuada, es decir, se quiere minimizar:

$$ECM = \sum_{i=1}^{n}(y_i - f(x_i))^2 \tag{2.6}$$

Si no se imponen restricciones sobre $f(x_i)$, siempre se puede hacer que (2.6) sea cero eligiendo f tal que $y_i = f(x_i)\ \forall\ i = 1, \ldots, n$ y que se interpole[2] todo el conjunto de entrenamiento. Dicha solución realizaría un ajuste demasiado cambiante y se produciría el efecto de **sobreajuste**, comentado en el apartado 1.3. Por lo que el objetivo principal va a ser encontrar una función f que minimice (2.6) y que a su vez realice un ajuste suave.

Para procurar que dicha función consiga el objetivo propuesto cumpliendo ambas condiciones, se busca una función f que minimice

$$\sum_{i=1}^{n}(y_i - f(x_i))^2 \ + \ \lambda \int_{-\infty}^{+\infty} f''(t)^2 dt \tag{2.7}$$

[2]Capacidad de predicción para nuevos datos que se encuentren dentro del rango de las observaciones del conjunto de entrenamiento.

donde λ es un número no negativo que recibe el nombre de **parámetro de suavización** (James, 2014, p.277).

La función f que minimiza (2.7) se conoce como **Spline de suavizado**. La expresión (2.7) está compuesta por una función de pérdida y una de penalización.

Función de pérdida: $\sum_{i=1}^{n}(y_i - f(x_i))^2$, mide la proximidad del ajuste realizado a los datos y se encarga de que f realice de manera adecuada el ajuste sobre el conjunto de datos.

Función de penalización: $\lambda \int_{-\infty}^{+\infty} f''(t)^2 dt$, penaliza la variabilidad de f. Dado que el término $f'(t)$ mide la pendiente de una función en t, el término $f''(t)$ mide cuánto está cambiando la pendiente.

Por lo tanto, cuando $f(t)$ es muy ondulada cerca de t, el término $f''(t)$ toma un valor alto, en valor absoluto, y, en caso contrario, toma un valor cercano a cero[3].

El término $\int_{-\infty}^{+\infty} f''(t)^2 dt$ es una medida del cambio total de $f'(t)$ a lo largo de todo su rango. Según la forma de la función f, esta integral tomará distintos valores:

- Si f es muy suave, $f'(t)$ será aproximadamente un valor constante y $\int_{-\infty}^{+\infty} f''(t)^2 dt$ tomará un valor muy pequeño.
- Si f es muy cambiante, $f'(t)$ variará mucho y $\int_{-\infty}^{+\infty} f''(t)^2 dt$ tomará un valor muy grande.

Por tanto, $\lambda \int_{-\infty}^{+\infty} f''(t)^2 dt$ penaliza la curvatura de la función e incentiva a f a realizar un ajuste suave.

Según el valor del parámetro de suavizado, se obtendrán distintos tipos de funciones f:

- Cuando λ toma un valor cercano a cero, $\lambda \int_{-\infty}^{+\infty} f''(t)^2 dt$ no produce ningún efecto. Por tanto, la función f será muy cambiante y se ajustará demasiado al conjunto de entrenamiento.
- Cuando $\lambda \to \infty$, se obtendrá una función f muy suavizada que puede ajustar a la mayoría de observaciones de manera errónea.

Un valor intermedio de λ asegura que se realice un ajuste que se acerque al conjunto de entrenamiento y que sea suave. Este parámetro equilibra el ajuste sobre todas

[3]La segunda derivada de una línea recta es cero. Una línea recta está totalmente suavizada

las observaciones y la suavidad del mismo.

Tal y como propone James, 2014, p.278, la función f que minimiza (2.7), $\hat{f}$, tiene las siguientes propiedades:

- Es un polinomio cúbico definido a trozos con tantos nodos como observaciones únicas haya.
- La primera y segunda derivada de dichos polinomios en cada nodo es coninua, pero no se impone la restricción de que las derivadas sean continuas en el primer y último nodo.

Como conclusión, esta función, $\hat{f}$, es un **spline cúbico natural** con nodos en cada observación única $x_1, \ldots, x_n$.

Elección del parámetro de suavización

Anteriormente hemos comentado la gran influencia del parámetro de suavizado en el ajuste de la función f. Ya se haya tomado un valor de λ alto o bajo, una elección errónea del mismo provocará un spline, $\hat{f}$, que no se adecúe de manera lo suficientemente precisa a la verdadera forma de la función f. Lo ideal sería tomar un valor de λ de manera que f y $\hat{f}$ sean lo más parecido posible (Wood, 2006, p.131-132). Para cuantificar la diferencia entre ambas funciones, se define la siguiente medida:

$$M = \frac{1}{n}\sum_{i=1}^{n}(\hat{f}_i - f_i)^2 \tag{2.8}$$

siendo $\hat{f}_i \equiv \hat{f}(x_i)$ y $f_i \equiv f(x_i)$.

Un criterio adecuado para determinar λ es que éste minimice (2.8). Al ser f desconocida, esta medida no es calculable, pero sí es posible estimar $E(M) + \sigma^2$ que es el error cuadrático medio a la hora de realizar predicciones.

Wood (2006) propone determinar el valor de λ mediante el siguiente procedimiento denominado **validación cruzada ordinaria**:

Sea $\hat{f}_i^{-i}$ el ajuste calculado para la observación x_i sin haberla tenido en cuenta en la construcción de $\hat{f}$, se define el valor de validación cruzada ordinario como

$$\nu_o = \frac{1}{n}\sum_{i=1}^{n}(\hat{f}_i^{-i} - y_i)^2. \tag{2.9}$$

Este valor se obtiene al dejar fuera del cálculo a una observación cada vez, ajustar el modelo sobre los datos restantes y calcular la diferencia cuadrada media entre el

dato no tenido en cuenta y su predicción. Sustituyendo $y_i = f_i + \epsilon_i$ en (2.9),

$$\begin{aligned} \nu_o &= \frac{1}{n}\sum_{i=1}^{n}(\hat{f}_i^{-i} - f_i - \epsilon_i)^2 \\ &= \frac{1}{n}\sum_{i=1}^{n}(\hat{f}_i^{-i} - f_i)^2 - 2(\hat{f}_i^{-i} - f_i)\epsilon_i + \epsilon_i^2 \end{aligned} \tag{2.10}$$

Como $E(\epsilon_i) = 0$ y ϵ_i y $\hat{f}_i^{-i}$ son independientes, al tomar valor esperado en (2.10) se obtiene

$$E(\nu_o) = \frac{1}{n}E\left(\sum_{i=1}^{n}(\hat{f}_i^{-i} - f_i)^2\right) + \sigma^2$$

Se puede afirmar que $\hat{f}^{-i} \approx f_i$ y si $n \to \infty$, se da la igualdad.

Igual ocurre con $E(\nu_o) \approx E(M) + \sigma^2$ y se da la igualdad cuando $n \to \infty$.

Por tanto, si lo que se quiere es minimizar M, un enfoque razonable sería elegir λ de manera que minimice ν_0.

Desde el punto de vista práctico, algunos autores, como Wood, 2006, p.132, consideran este proceso ineficiente y proponen el siguiente procedimiento que tiene otro valor de validación cruzada asociado:

$$\nu_o = \frac{1}{n}\sum_{i=1}^{n}\frac{(y_i - \hat{f}_i)^2}{(1 - A_{ii})^2}$$

donde $\hat{f}_i$ es el valor estimado obtenido del ajuste sobre todas las observaciones y A es una matriz influencia (Wood, 2006, p.173-175).

A su vez, para evitar problemas de cálculo de los elementos diagonales de la matriz A, en la práctica, se propone cambiar $1 - A_{ii}$ por $tr(I - A)/n$, dando lugar a lo que se conoce como **validación cruzada generalizada** cuyo valor asociado es

$$\nu_g = \frac{n\sum_{i=1}^{n}(y_i - \hat{f}_i)^2}{[tr(I - A)]^2}$$

2.2.4. Otras técnicas

Una vez presentados los splines de regresión y de suavizado, a continuación, pasamos a hacer una breve introducción sobre otras técnicas de construcción de las funciones f que, a pesar de basarse en otra metodología, siguen operando mediante splines.

B-Splines

Otra manera de representar splines cúbicos (y splines de mayor o menor orden) es haciendo uso de una base B-Spline (Wood, 2006, p.152-153).

La base B-Spline se caracteriza porque las funciones usadas son estrictamente locales, es decir, cada función base es distinta de cero en los intervalos definidos por $m+3$ nodos adyacentes, donde $m+1$ es el orden de la base, siendo $m=2$ para los splines cúbicos.

Para definir una base B-Spline de parámetro k, es necesario definir $k+m+1$ nodos, $x_1^* < x_2^* < \ldots < x_{k+m+1}^*$, donde el intervalo sobre el cual el spline sea evaluado se encuentre en $\left[x_{m+2}^*, x_k^*\right]$. Un spline de orden $m+1$ puede ser representado mediante

$$f(x) = \sum_{i=1}^{k} B_i^m(x)\beta_i$$

donde las funciones base del B-Spline se definen de manera recursiva como se muestra a continuación:

$$B_i^m(x) = \frac{x - x_i^*}{x_{i+m+1}^* - x_i^*} B_i^{m-1}(x) + \frac{x_{i+m+2}^* - x}{x_{i+m+2}^* - x_{i+1}^*} B_{i+1}^{m-1}(x) \quad i = 1, 2, \ldots, k$$

y

$$B_i^{-1}(x) = \begin{cases} 1 & \text{si } x_i^* < x \leq x_{i+1}^* \\ 0 & \text{en caso contrario} \end{cases}$$

En particular, para splines cúbicos con nodos $x^* = 0, 1, 2, 3$ (Pino, 2016), la base resultante es la siguiente:

$$\begin{aligned} B_1 &= \frac{x^2}{2}, & 0 \leq x \leq 1 \\ B_2 &= \frac{-2x^2 + 6x - 3}{2}, & 1 \leq x \leq 2 \\ B_3 &= \frac{(3-x)^2}{2}, & 2 \leq x \leq 3 \end{aligned}$$

A continuación, presentamos algunos gráficos[4] de las funciones base $B_i^m(x)$ con $m = 0, 1, 2$ (Brusch, 2016, p.25-27).

La base de orden 0, B_i^0, está definida entre los nodos x_i^* y x_{i+1}^*. Como se observa en la figura 2.2 se obtiene una función escalonada de valor 1 en $\left[x_i^*, x_{i+1}^*\right]$ y cero en caso contrario.

La base de orden 1, B_i^1, está definida entre tres nodos adyacentes x_i^* y x_{i+2}^*. Como se observa en la figura 2.3 se produce un polinomio a trozos de grado 1 en $\left[x_i^*, x_{i+2}^*\right]$ y

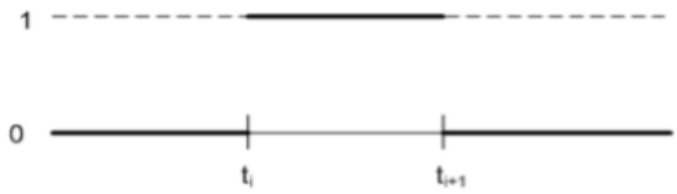

Figura 2.2: *Función base del B-Spline* B_i^0.

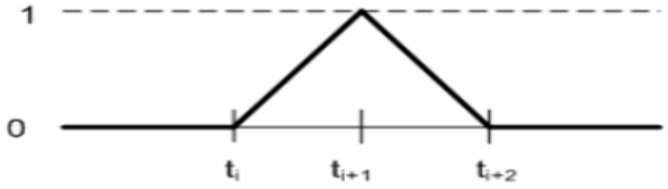

Figura 2.3: *Función base del B-Spline* B_i^1.

cero en caso contrario.

La base de orden 2, B_i^2, está definida entre cuatro nodos adyacentes x_i^* y x_{i+3}^*. Como se observa en la figura 2.4 se produce un polinomio cuadrático a trozos con una transición suave en $\left[x_i^*, x_{i+3}^*\right]$.

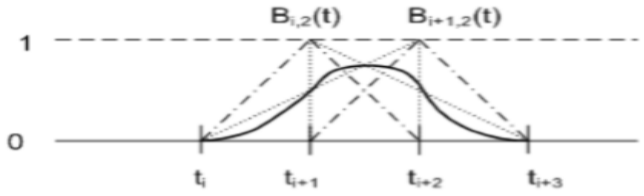

Figura 2.4: *Función base del B-Spline* B_i^2.

P-Splines

A lo largo de este capítulo hemos presentado dos enfoques de gran relevancia a la hora de construir las funciones no necesariamente lineales para modelizar la relación entre la variable objetivo Y y una serie de variables explicativas $X_1, X_2, \ldots, X_p$ y estos son los splines de regresión 2.2.2 y los de suavizado 2.2.3.

En el caso de los splines de regresión, estos se ajustan por mínimos cuadrados una vez se hayan seleccionado el número de nodos. Por otro lado, se estableció que los de suavizado definen tantos nodos como observaciones únicas haya. Esto conlleva que el número de parámetros sea bastante elevado si el número de observaciones es considerable y podría conducir a una implementación no eficiente.

[4]En las representaciones gráficas 2.2, 2.3 y 2.4 los nodos se representan mediante t_i.

El objetivo de los **P-splines**, recogidos a continuación, es combinar lo mejor de ambos tipos de splines. Hacen uso de un menor número de parámetros que los splines de suavizado y la selección de los nodos no es tan determinante como para los splines de regresión. Son splines de rango bajo, es decir, el tamaño de la base usada es mucho menor que la dimensión del conjunto de datos, al contrario que para los splines de suavizado.

Durbán, 2018, p.23-31, recomienda tomar menos de 40 nodos para asegurar una implementación computacional eficiente, sobre todo si se trabaja con una gran cantidad de datos.

Estos splines funcionan como suavizadores que usan una base de B-Spline, normalmente definidos para nodos uniformemente distribuidos, con una penalización diferente aplicada directamente a los parámetros, β, para controlar la ondulación de la función (Wood, 2006, p.153). Se utiliza una penalización basada en las diferencias de orden d entre los coeficienes adyacentes de las bases de B-Splines. Este tipo de penalización es más flexible ya que es independiente del grado del polinomio utilizado para construir los B-Splines.

Si se decide penalizar la diferencia de orden 2 entre los coeficientes adyacentes de la base de B-Spline (Wood, 2006, p.154), la penalización se definiría de la siguiente manera:

$$P = \sum_{i=1}^{k} (\beta_{i+1} - \beta_i)^2 = \beta_1^2 - 2\beta_1\beta_2 + 2\beta_2^2 - 2\beta_2\beta_3 + \cdots + \beta_k^2,$$

y P se podría expresar como sigue

$$P = \boldsymbol{\beta}^T \begin{bmatrix} 1 & -1 & 0 & . & . \\ -1 & 2 & -1 & . & . \\ 0 & -1 & 2 & . & . \\ . & . & . & . & . \\ . & . & . & . & . \end{bmatrix} \boldsymbol{\beta}.$$

2.3. Modelo Aditivo

Una vez presentadas las distintas técnicas para el estudio de la función f teniendo en cuenta una sola variable explicativa, es necesario explicar qué ocurre cuando hay más de una variable explicativa en el estudio.

El objetivo del **Modelo Aditivo** es expresar o modelizar la variable respuesta Y, o una característicasde la misma, a partir de la suma de las funciones suaves aplicadas a las distintas variables explicativas.

Por ejemplo, supongamos que trabajamos con dos variables explicativas, X y Z,

entonces la estructura del modelo aditivo es la siguiente:

$$y_i = f_1(x_i) + f_2(z_i) + \epsilon_i, \quad i = 1, \ldots, n \tag{2.11}$$

donde f_1 y f_2 son las funciones que permiten establecer una relación no lineal entre la variable respuesta y cada una de las explicativas y ϵ_i es el error aleatorio i.i.d según una $N(0, \sigma^2)$ para $i = 1, \ldots, n$ (Wood, 2006, p.133).

Hay que tener presente que modelizar la variable respuesta como la suma de funciones aplicadas a cada una de las variables en vez de una sola función aplicada a ambas variables impone una condición muy fuerte pues $f_1(x) + f_2(z)$ es un caso especial y restrictivo de la función definida sobre las dos variables $f(x, z)$.

No obstante, el aspecto positivo de definir cada función por separado es que cada variable explicativa mantiene la capacidad de interpretación del modelo lineal. Trabajar a partir de $f(x, z)$ proporcionaría mayor flexibilidad en el modelo, pero constituiría una traba a la hora de interpretar los resultados obtenidos.

Este modelo puede ser analizado mediante las técnicas vistas en 2.2 sobre cada función definida y el grado de suavizado puede ser seleccionado mediante validación cruzada al igual que en el modelo univariante.

2.4. Interacción entre variables

Cuando se trabaja con al menos dos variables explicativas, las técnicas descritas hasta ahora no tienen en cuenta la posible interacción entre ellas. A continuación, recogemos una breve introducción sobre técnicas que sí tienen en cuenta la posible interacción entre las variables explicativas.

Thin Plate Splines

Tal y como se recoge en Wood, 2006, p.154, los splines Thin Plate son considerados como una solución general al problema de construir una función suave de más de una variable explicativa.

Se considera el problema de estimar una función suave a partir de n observaciones $(y_i, \mathbf{x}_i)$, $i = 1, 2, \ldots, n$, de manera que

$$y_i = k(\mathbf{x}_i) + \epsilon_i, \quad i = 1, 2, \ldots, n$$

Los splines Thin Plate estiman k mediante una función $\hat{f}$ que minimiza

$$\|\mathbf{y} - \mathbf{f}\|^2 + \lambda J_{mp}(f) \tag{2.12}$$

donde $\mathbf{y}$ es un vector compuesto por las observaciones y_i, $i = 1, 2, \ldots, n$, p el número de variables explicativas, m el orden de las derivadas, $\mathbf{f} = [f(\mathbf{x}_1)\, f(\mathbf{x}_2)\, \ldots\, f(\mathbf{x}_n)]^t$, λ es el parámetro de suavización que controla el balance entre el ajuste del modelo

y la suavidad de f. La función $J_{mp}(f)$ es una función de penalización que mide la ondulación de f y se define como sigue:

$$J_{mp}(f) = \int \dots \int_{R^p} \sum_{\nu_1+\dots+\nu_p=m} \frac{m!}{\nu_1! + \dots + \nu_p!} \left(\frac{\partial^m f}{\partial x_1^{\nu_1} \dots \partial x_p^{\nu_p}} \right)^2 dx_1 \dots dx_p. \quad (2.13)$$

En el caso de dos variables explicativas, cuya ondulación está medida mediante las segundas derivadas, la penalización toma la siguente forma:

$$J_{22} = \iint \left(\frac{\partial^2 f}{\partial x_1^2} \right)^2 + 2 \left(\frac{\partial^2 f}{\partial x_1 \partial x_2} \right)^2 \left(\frac{\partial^2 f}{\partial x_2^2} \right)^2 dx_1 dx_2.$$

Para el estudio general del problema es necesario elegir m de manera que $2m > p$ aunque desde el punto de vista práctico Wood, 2006, p.154, aconseja elegir m tal que $2m > p + 1$. Sujeto a estas restricciones, se puede demostrar que la función que minimiza (2.12) es

$$\hat{f}(\mathbf{x}) = \sum_{i=1}^{n} \delta_i \, \eta_{mp} \left(\|\mathbf{x} - \mathbf{x}_i\| \right) \quad + \quad \sum_{j=1}^{M} \alpha_j \phi_j(\mathbf{x}) \quad (2.14)$$

donde $\boldsymbol{\delta}$ y $\boldsymbol{\phi}$ son vectores de coeficientes que posteriormente habrá que estimar, $\boldsymbol{\delta}$ está sujeto a la restricción $\mathbf{T}^t\boldsymbol{\delta} = 0$ donde $T_{ij} = \phi_j(x_i)$. La definición de la expresión η_{md} está recogida en Wood, 2006, p.156.

Sea la matriz $\mathbf{E}$ cuyos elementos son $E_{ij} \equiv \eta_{mp}(\|\mathbf{x}_i - \mathbf{x}_j\|)$, el problema a la hora de ajustar el spline Thin Plate pasa a ser:

$$minimizar_{\delta,\alpha} \quad \|\mathbf{y} - \mathbf{E}\boldsymbol{\delta} - \mathbf{T}\boldsymbol{\alpha}\|^2 + \lambda \boldsymbol{\delta}^t \mathbf{E} \boldsymbol{\delta} \quad s.a \quad \mathbf{T}^t\boldsymbol{\delta} = 0. \quad (2.15)$$

Un estudio más profundo de esta técnica se recoge en Wood, 2006, p.155-156.

A lo largo de este proceso no se ha escogido la posición de los nodos ni ninguna función base. Los splines Thin Plate pueden trabajar con cualquier número de variables explicativas. Aunque los problemas comentados anteriormente parecen haber sido solucionados, los splines thin plate cuentan con una desventaja: tienen tantos parámetros como número de datos y, por tanto, su uso conlleva un gran coste computacional, sobre todo en el caso multidimensional.

Por ello, Wood, 2006, p.157, introduce los **Splines de regresión Thin Plate**. Estos son una versión basada en los anteriores en los que el número de parámetros es mucho menor al número de datos. Estos splines evitan el problema de la selección del lugar de los nodos, son computacionalmente eficientes y pueden ser contruidos mediante funciones de cualquier número de variables explicativas.

Un estudio más profundo de esta técnica se recoge en Wood, 2006, p.158-160.

Tensores

En algunas ocasiones resulta complicado interpretar distintas variables explicativas, en relación a la una respecto a las otras, cuando forman parte de la misma función suave y tienen distintas unidades de medida. Por este motivo, Wood, 2006, p.162, introduce una nueva técnica que puede ser usada para construir funciones suaves de cualquier número de variables explicativas, conocida como **tensores**.

A continuación, se expone un ejemplo en el que se quiere construir una función suave de tres variables explicativas $X, Z\, y\, V$.

Primero, suponemos que contamos con bases de bajo rango disponible para representar las funciones suaves $f_X, f_Z\, y\, f_V$ para cada variable explicativa por separado. Se denotan mediante:

$$f_X(x) = \sum_{i=1}^{I} \alpha_i a_i(x), \quad f_Z(z) = \sum_{l=1}^{L} \delta_l d_l(z) \quad y \quad f_V(v) = \sum_{k=1}^{K} \beta_k b_k(v)$$

donde $\alpha_i,\ \delta_l\ y\ \beta_k$ son los parámetros y $a_i(x),\ d_l(z)\ y\ b_k(v)$ las funciones base conocidas.

Es necesario considerar cómo la función suave de X, f_X, puede transformarse en una función suave de $X\, y\, Z$ (Wood, 2006, p.163). Esto requiere que f_X varíe suavemente con Z y esto se consigue permitiendo que los parámetros, α_i, varíen suavemente con Z. Haciendo uso de la base ya disponible para representar la función suave de Z, se expresa

$$\alpha_i(z) = \sum_{l=1}^{L} \delta_{il} d_l(z)$$

y se obtiene

$$f_{XZ}(x, z) = \sum_{i=1}^{I} \sum_{l=1}^{L} \delta_{il} d_l(z) a_i(x)$$

Para crear una función suave de $X, Z\, y\, V$ se realiza el mismo proceso. Esto requiere que f_{XZ} varíe suavemente con V y esto se consigue permitiendo que los parámetros de f_{XZ} varíen suavemente con V. Por tanto, se obtiene

$$\delta_{il}(v) = \sum_{k=1}^{K} \beta_{ilk} b_k(v)$$

$$f_{XZV}(x, z, v) = \sum_{i=1}^{I} \sum_{l=1}^{L} \sum_{k=1}^{K} \beta_{ilk} b_k(v) d_l(z) a_i(x)$$

Habiendo obtenido esta función, es necesario tener alguna medida de la ondulación de f_{XZV} si se quiere que la base sea útil para representar funciones suaves en el contento del Modelo Aditivo Generalizado.

Un estudio más profundo de los tensores se recoge en Wood, 2006, p.165-167.

2.5. Modelo Aditivo Generalizado

Al igual que el Modelo Lineal Generalizado, visto en 1.2.1, es una ampliación del Modelo Lineal, el **Modelo Aditivo Generalizado** es una ampliación del Modelo Aditivo.

Como vimos al comienzo del capítulo, 2.1, este modelo pretende analizar la relación entre una variable respuesta Y, que puede seguir cualquiera de las distribuciones de la Familia Exponencial, y una serie de variables explicativas mediante el ajuste de funciones suaves y no obligatoriamente lineales sobre cada variable explicativa individualmente. La estructura del modelo es similar a (2.1).

Este modelo permite ajustar una función f_j no lineal sobre la variable explicativa X_j para modelizar relaciones no lineales entre Y y $X_1, X_2, \ldots, X_p$, hecho que la regresión lineal no es capaz de llevar a cabo. Con los ajustes no lineales pueden obtenerse predicciones más precisas de la variable respuesta.

Al ser el modelo aditivo, se puede estudiar el efecto de cada variable explicativa X_j sobre la variable respuesta Y o una característica suya individualmente mientras se mantienen fijas el resto de variables explicativas. Este modelo también trae consigo una limitación y es que está restringido a la aditividad.

Cuando se trabaja con al menos dos variables explicativas, técnicas como la regresión polinómica, los plines de regresión o los de suavizado no tienen en cuenta la posible interacción entre las variables explicativas. Para estudiar las interacciones entre un número determinado de variables, se pueden usar los Splines de regresión Thin Plate y los Tensores 2.4.

A continuación, describimos la estructura del Modelo Aditivo Generalizado teniendo en cuenta la posible interacción de variables explicativas.

Como se vio al comienzo del capítulo, el **Modelo Aditivo Generalizado** modeliza una variable respuesta, y_i, usando un modelo cuya estructura, (2.1), pasa a ser:

$$g(\mu_i) = \mathbf{X}_i^* \boldsymbol{\theta} + f_1(x_{1i}) + f_2(x_{2i}, x_{3i}) + f_3(x_{4i}) + \ldots, \quad i = 1, 2, , \ldots, n \tag{2.16}$$

donde $g()$ es la función enlace, como vimos en el apartado 1.2.1, $\mu_i = E(Y_i)$ con Y_i variable respuesta e $Y_i \sim$ Distribución de la Familia Exponencial, $\mathbf{X}_i^*$ es la i-ésima fila de la matriz de diseño correspondiente a las covariables que definen las componentes paramétricas del modelo, f_j, $j = 1, 2, 3 \ldots$ son las funciones suaves y θ es el vector de coeficientes de regresión (Wood, 2006, p.167).

Para ajustar dicho modelo se puede especificar una base para cada función f_j haciendo uso de las distintas técnicas expuestas a lo largo de este capítulo.

Capítulo 3

Criptomoneda

3.1. Introducción

En general, por dinero digital se entiende cualquier medio de intercambio monetario que se realiza por un medio electrónico como, por ejemplo, cuando se hace una transferencia de una cuenta bancaria a otra, cuando se paga con tarjeta una compra en una tienda o cuando se compra un producto por Internet. En estos casos, cuando se realiza un pago de dinero sin que haya un intercambio físico de moneda o billetes, se habla de dinero digital.

Existe también la moneda virtual que se caracteriza porque sólo existe en formato digital. Por ejemplo, actualmente en muchos videojuegos existe una divisa particular con la que se pueden adquirir diferentes productos para hacer más ameno el videojuego. Este es el caso del FIFA, videojuego de fútbol, en el que el usuario paga una cierta cantidad de dinero en euros, dólares, ..., recibe la cuantía correspondiente de monedas particulares del juego y, con ellas, puede comprar sobres de cartas que incluyen diferentes jugadores de fútbol para formar su propio equipo.

Por tanto, como las monedas virtuales no existen físicamente, todas las monedas virtuales son digitales, pero no viceversa (Economipedia). Por ejemplo, el dinero que se tiene en una cuenta bancaria del banco es moneda digital, pero no virtual.

Las **criptomonedas**, como el Bitcoin, Ethereum, Litecoin, Dogecoin o Ripple son un tipo de moneda virtual que funciona como cualquier otra divisa, es decir, se pueden usar para comprar o vender bienes y servicios.

El origen de la criptomoneda proviene del movimiento Cypherpunk en la década de los 80. Este movimiento defiende que la criptografía sea mundialmente utilizada como herramienta de cambio social y político. En 1990 David Chaun creó Digicash, un sistema centralizado de dinero electrónico que permitía transacciones caracterizadas por una mayor seguridad y anonimato. En la misma década, Adam Black propone Hashcash, un sistema basado en prueba de trabajo para controlar el spam y los ataques de denegación de servicio (Oroyfinanzas).

No fue hasta el año 2009 cuando apareció la primera criptomoneda completamente

descentralizada, el Bitcoin, que fue creada por una persona o grupo de personas bajo el seudónimo de Satoshi Nakamoto. La innovación, llevada a cabo por los mismos, que impulsó el Bitcoin y el resto de criptomonedas posteriores se conoce como cadena de bloques o, en inglés, **blockchain**.

Las características fundamentales de las criptomonedas son:

- Hace uso de la **criptografía** para que los pagos y cobros se realicen de manera segura. Por criptografía se entiende la técnica de escribir con procedimientos o claves secretas de manera que lo escrito sólo sea inteligible para aquella persona que sepa descifrarlo.
- **Descentralización**. No está regulada por ningún Gobierno, banco central ni ninguna institución monetaria. Esto provoca que puedan ser utilizadas para transacciones ilegales.
- No hay intermediarios. Se establece un contacto directo entre vendedor y comprador, lo que se conoce como **peer to peer**.
- Las transacciones realizadas son **irreversibles**, es decir, una vez se efectúa el pago, no se puede cancelar.
- Son intercambiables por otras **divisas** como euros, dólares, libras, ...
- No es necesario revelar tu **identidad** al hacer negocios.

El blockchain es un libro de cuentas similar al de un banco, pero con copias del mismo en ordenadores de todos los que participan en la cadena, incluyendo datos de cantidad, fecha, operación y participantes. Estos aspectos se actualizan automáticamente con cada transacción y una vez registradas, los registros son inalterables. Por tanto, se obtiene un registro fiel y verificable de todas las operaciones (Criptomoneda.Ninja).

Así pues, el blockchain se encarga de guardar cada una de las transacciones realizadas en conjuntos de datos conocidos como bloques. Estos bloques están conectados cronológicamente, es decir, al crearse un bloque nuevo, se enlaza con el que se ha creado inmediatamente anterior y se enlazará con el que se cree inmediatamente posterior. Los bloques que se originan están almacenados en millones de ordenadores conectados a la red del blockchain. Cada uno de estos ordenadores se conoce como nodo de la red blockchain.

Para poder falsificar una operación habría que cambiar los registros de todos los ordenadores que guardan una copia. Esta es una operación inviable en la práctica, lo que supone un aspecto fundamental para las criptomonedas.

Conforme se van creando los bloques, es necesario que se vayan validando. Cualquier persona se puede ofrecer como voluntaria para realizar este trabajo que se verá recompensado, en el caso del Bitcoin, con un pago en esa moneda. Estas personas, conocidas como mineros, realizan dos comprobaciones consultando la cadena de bloques. Primero, comprueban que tú has recibido anteriormente el dinero que quieres transferir y,

posteriormente, comprueban que la transacción está firmada con la clave privada de cada propietario. Si todo se cumple correctamente, comienzan a validar los bloques.

3.2. Problemas con criptomonedas

Uno de los problemas que ha acompañado a las criptomonedas es la volatilidad de los precios, tal y como se puede observar en los gráficos 3.1 y 3.2, aspecto que aleja a posibles usuarios. Esta volatilidad proviene de la especulación que consiste en la compra de bienes, en este caso criptomonedas, para revender a un mayor precio y se produce por la incertidumbre existente sobre el precio de las criptomonedas.

Una moneda debe transmitir una sensación de fiabilidad, es decir, los usuarios deben tener la certeza de que con el dinero que tienen hoy podrán comprar aproximadamente las mismas cosas mañana, dentro de una semana o dentro de un año. Este aspecto no lo cumplen las criptomonedas como, por ejemplo, es el caso del Bitcoin tal y como se pueden observar en los gráficos 3.1 y 3.2. Su valor puede llegar a cotas muy altas y descender vertiginosamente en pocos días.

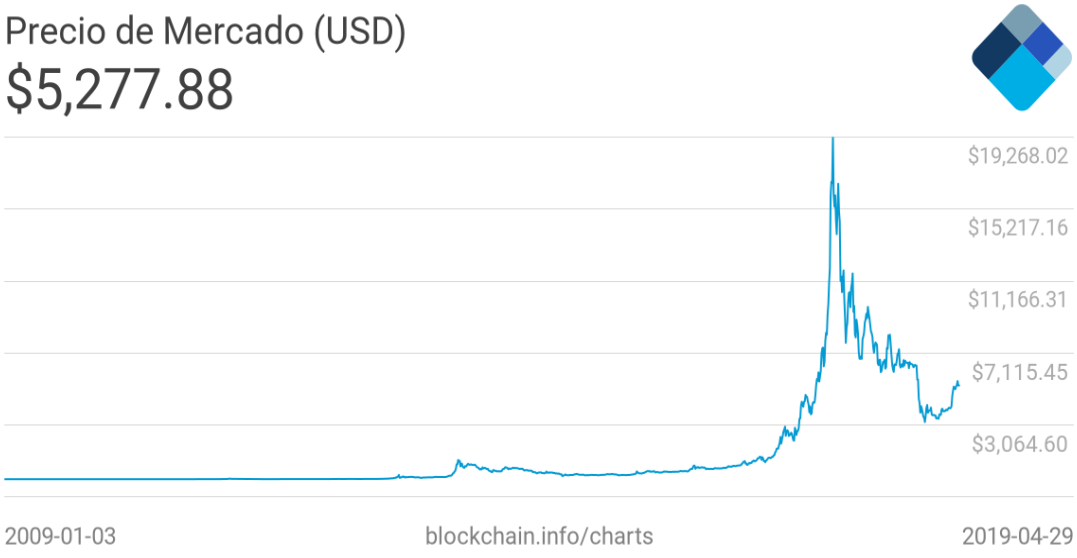

Figura 3.1: *Evolución del precio de mercado del bitcoin.*

En principio, el pago que se realiza al minero es la única comisión del sistema blockchain. Normalmente las transacciones no se realizan directamente desde el propio sistema blockchain, sino que se realizan a través de plataformas especializadas que se conocen como **casas de cambio**. Estas casas suministran aplicaciones para adquirir criptomonedas. Al no haber actualmente mucha competencia en el sector, las comisiones de las casas de cambios son más elevadas que las propias del sistema blockchain.

A medida que las criptomonedas fueron ganando popularidad, el mercado se ha ido llenando de estas casas, pero no todas son igual de fiables ni aplican la misma comisión a cambio de usarlas (Coinmarketcap).

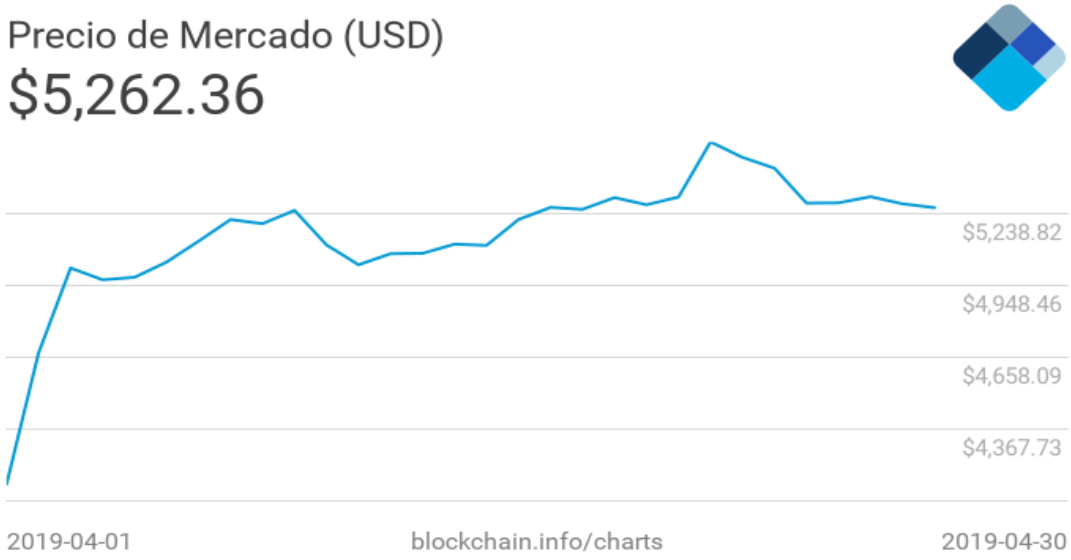

Figura 3.2: *Evolución del precio de mercado del bitcoin durante el mes de abril de 2019.*

Actualmente existe un alto número de este tipo de plataformas de calidad constatada para la comercialización de criptomonedas. No obstante, al existir mucha más demanda que oferta, las plataformas se aprovechan de esto para ofrecer sus servicios a cambio de altas comisiones.

Nuestro trabajo estudia la aplicación de las técnicas estudiadas en el segundo capítulo para llevar a cabo la selección de la mejor casa de cambio a través de la cual realizar la transacción pertinente.

Capítulo 4

Aplicación MAG a las criptomonedas

4.1. Objetivo

Como se ha visto en el capítulo anterior, el precio de las criptomonedas se caracteriza por ser volátil. Su valor puede llegar a cotas muy altas y descender vertiginosamente en pocos días. Al igual que se puede perder mucho dinero si se toma una mala decisión en el momento más desafortunado, se puede llegar a ganar mucho dinero si se vende una cantidad determinada de criptomonedas en el momento preciso.

Por este motivo, cualquier modelo para el estudio de la relación entre el precio de las criptomonedas y distintas variables explicativas como, por ejemplo, la cantidad de criptomonedas de la transacción o el momento de la compra, y que sea capaz de predecir una posible situación futura, será muy útil para tomar la decisión de si comprar o no en un momento determinado y a través de qué casa de cambio efectuar la transacción.

Debido a la sabida volatilidad del precio de la criptomoneda, se entiende que aplicar el **Modelo Lineal Generalizado** puede no ser de mucha ayuda para conseguir dicho objetivo, pues este se limita a relacionar la variable objetivo, en este caso el precio de la criptomoneda, con las distintas variables explicativas mediante una combinación lineal de éstas. Comprobaremos que, efectivamente, este modelo no conduce a soluciones adecuadas para la selección de casas de cambio.

En los anteriores capítulos hemos introducido una alternativa al Modelo Lineal Generalizado que es el **Modelo Aditivo Generalizado**. Este permite relacionar la variable objetivo o una característica suya con cada variable explicativa de manera no necesariamente lineal mediante distintas funciones suaves f_j y bajo la condición de aditividad. En el segundo capítulo hemos expuesto distintas técnicas para la búsqueda de dichas funciones f_j, tales como la regresión polinómica, splines cúbicos de regresión, splines de suavizado, B-Splines, P-Splines, splines de regresión Thin Plate o tensores. Todas estas técnicas, que hemos introducido de manera teórica, se pueden trasladar a la práctica mediante el software estadístico **R**.

El objetivo a conseguir será demostrar la utilidad del Modelo Aditivo Generalizado y de las distintas técnicas asociadas para modelizar la relación, aparentemente no lineal, tal y como podemos observar en los gráficos 4.1 y 4.2, entre las variables involucradas en el estudio. Además, una vez propuesto el mejor modelo para elegir la casa de cambio cuyo precio predicho sea más bajo, realizaremos una simulación para mostrar como podríamos aplicarlo en la práctica.

4.2. Descripción de los datos

La base de datos original está formada por 200188 observaciones referidas a la criptomoneda **Ethereum**, tomadas durante tres semanas desde el 22 de noviembre de 2018 hasta el 12 de diciembre del mismo año. Las casas de cambio involucradas en este estudio son casas reales anonimizadas.

```
load("datos.RData")
str(datos)

## 'data.frame': 200188 obs. of  14 variables:
##  $ _id     : chr  "5bf67bf64154d84bb82661ac" "5bf67bf64154d84bb82661ab" "5bf67bf6
##  $ t_id    : num  5275791 5275792 5275790 5275788 5275785 ...
##  $ house   : chr  "5bf67a3abf23fc4b4565be83" "5bf67a3abf23fc4b4565be83" "5bf67a3a
##  $ price   : num  115 115 115 115 115 ...
##  $ quantity: num  3.3921 0.2001 5.327 15.7011 0.0465 ...
##  $ fee     : num  0.003 0.003 0.003 0.003 0.003 0.003 0.003 0.003 0.003 0.003 ...
##  $ date    : chr  "2018-11-22T09:50:11.628Z" "2018-11-22T09:50:12.561Z" "2018-11-
##  $ type    : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ total   : num  390.18 23.02 613.05 1803.65 5.35 ...
##  $ __v     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ casa    : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ pxq     : num  391.35 23.09 614.9 1809.08 5.37 ...
##  $ pneto   : num  115 115 115 115 115 ...
##  $ time    : chr  "2018-11-22 09:50:11.628" "2018-11-22 09:50:12.561" "2018-11-22
```

Como podemos observar, la base de datos cuenta con 14 variables. Para este estudio usaremos como variable objetivo la variable **Precio** (price) que es el precio en dólares de una unidad de criptomoneda en el momento de la transacción. Las variables explicativas del estudio son las siguientes:

- **Cantidad** (quantity): unidades de criptomonedas comercializadas en la transacción.
- **Casa** (casa): casa de cambio involucrada en la compraventa.
- **Tiempo**: número de segundos transcurridos en el años 2018 hasta llegar a la fecha correspondiente de la transacción. Esta variable será calculada a través de la variable **Time** de la base de datos original.
- **Lagprecio**: precio correspondiente a la transacción anterior. Esta variable se obtendrá a través de la variable **Price** de la base de datos original.

Primero, cargamos las librerías necesarias para llevar a cabo tanto la selección del subconjunto y el tratamiento de los datos (tidyverse, lubridate) como la modelización entre las variables (splines, mgcv).

```
library(lubridate)
library(tidyverse)
library(splines)
library(mgcv)
```

Para comenzar, seleccionamos las variables Precio (price), Cantidad (quantity), Fecha (time) y Casa (casa).

```
datos1 <- datos %>% select(price, quantity, time, casa)

str(datos1)

## 'data.frame': 200188 obs. of  4 variables:
##  $ price   : num  115 115 115 115 115 ...
##  $ quantity: num  3.3921 0.2001 5.327 15.7011 0.0465 ...
##  $ time    : chr  "2018-11-22 09:50:11.628" "2018-11-22 09:50:12.561" "2018-11-22
##  $ casa    : num  1 1 1 1 1 1 1 1 1 1 ...
```

Observamos que las variables Precio, Cantidad y Casa son de tipo numérico y Fecha de tipo carácter. Esta última incluye tanto la fecha como la hora en la que se produjo la transacción.

A continuación, realizamos las siguientes modificaciones en el conjunto de datos:

- Se transforma la variable Casa para pasarla a factor. En este estudio contamos con datos de tres casas de cambio.
- Se renombran las variables Precio y Cantidad.
- Se crea una nueva variable, Tiempo, a partir de la variable Fecha, que representa el número de segundos transcurridos en el años 2018 hasta llegar a la fecha correspondiente.
- Se crea una nueva variable, Lagprecio, a partir de la variable Precio. Se desplazan las observaciones un lugar hacia la derecha, por lo que corresponde al precio de la transacción anterior.

```
datos2 <- datos1 %>%
  mutate(fecha = as.Date(time), casa=as.factor(casa)) %>%
  rename(precio=price, cantidad=quantity) %>%
  select(-time)

datos2$tiempo<-as.numeric(as.POSIXct(datos1$time))
datos2$lagprecio<-lag(datos2$precio)
```

```
str(datos2)

## 'data.frame': 200188 obs. of  6 variables:
##  $ precio   : num  115 115 115 115 115 ...
##  $ cantidad : num  3.3921 0.2001 5.327 15.7011 0.0465 ...
##  $ casa     : Factor w/ 3 levels "1","2","3": 1 1 1 1 1 1 1 1 1 1 ...
##  $ fecha    : Date, format: "2018-11-22" "2018-11-22" ...
##  $ tiempo   : num  1.54e+09 1.54e+09 1.54e+09 1.54e+09 1.54e+09 ...
##  $ lagprecio: num  NA 115 115 115 115 ...
```

A continuación, presentamos algunos gráficos (4.1, 4.2) que describen la evolución del precio a lo largo de las tres semanas según las casas y la relación entre el precio de venta y cantidad de criptomonedas comprada.

```
datos2 %>% ggplot(aes(y=precio, x=tiempo, group=casa)) +
  geom_line(aes(colour=casa))+
        theme(strip.text = element_text(size=40),
                    axis.title.x = element_text(vjust=-0.8,size=18),
                    axis.title.y = element_text(vjust=-0.2,size=18),
                    axis.text = element_text(size=18),
                    axis.text.x = element_text(angle=0, vjust=0.5),
        panel.margin = unit(1, "lines"))
```

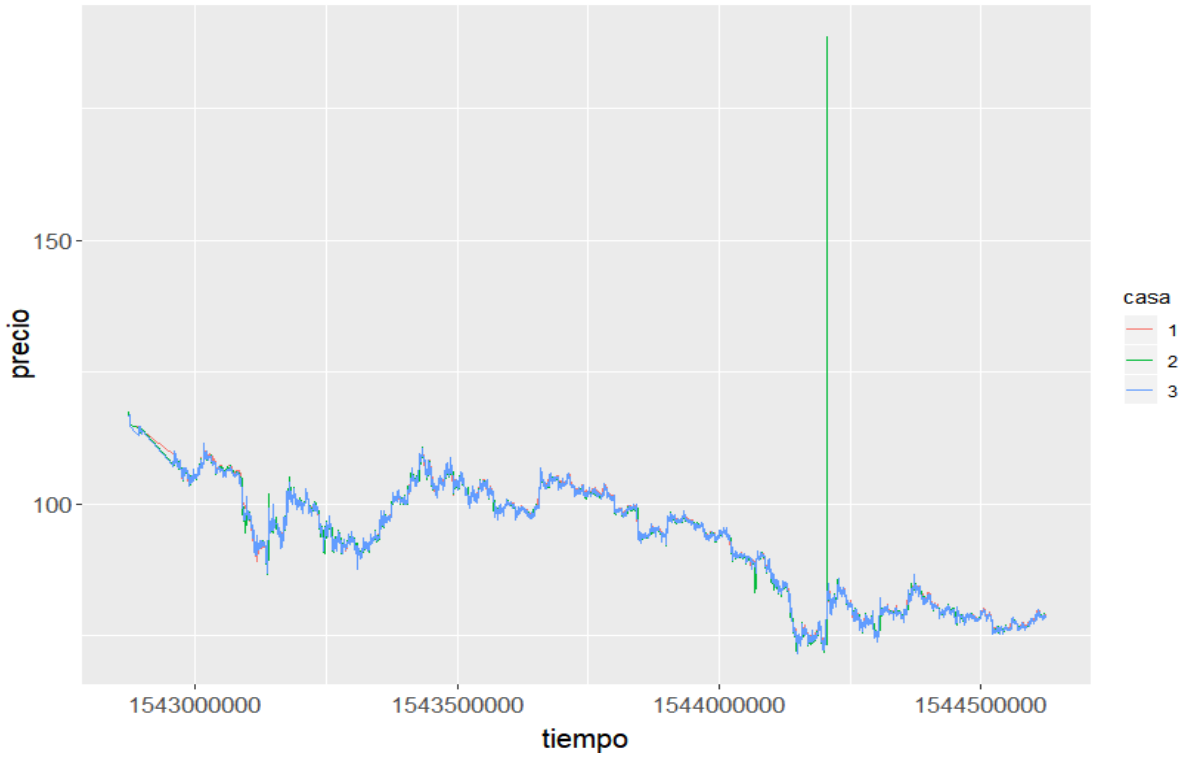

Figura 4.1: *Evolución del precio a lo largo de las 3 semanas según las casas*

Observamos que las tres casas de cambio involucradas mantienen sus precios semejantes, salvo la Casa 1 que, al principio, aumenta su precio un poco por encima de las otras dos y la Casa 2 que en determinados momentos incrementa o disminuye sus precios con respecto a las otras dos casas. Cabe destacar que, el día 7 de diciembre, la Casa 2 sufre un incremento considerable de los precios superando los 175 dólares, pero en el mismo día vuelve al nivel de precios habitual.

Al principio, el precio rondaba los 120 dólares y, a lo largo de las tres semanas, ha ido sufriendo constantes subidas y bajadas hasta terminar un poco por encima de los 75 dólares. Esto demuestra la sabida volatilidad del precio de las criptomonedas.

La evolución del precio de la criptomoneda a lo largo del tiempo también puede estudiarse mediante alguno de los modelos de series temporales, como el ARIMA, pero éste no es el objetivo de este trabajo.

```
datos2 %>%ggplot(aes(y=precio, x=cantidad, group=casa)) +
  geom_point(aes(colour=casa)) +
  facet_wrap(~casa)
```

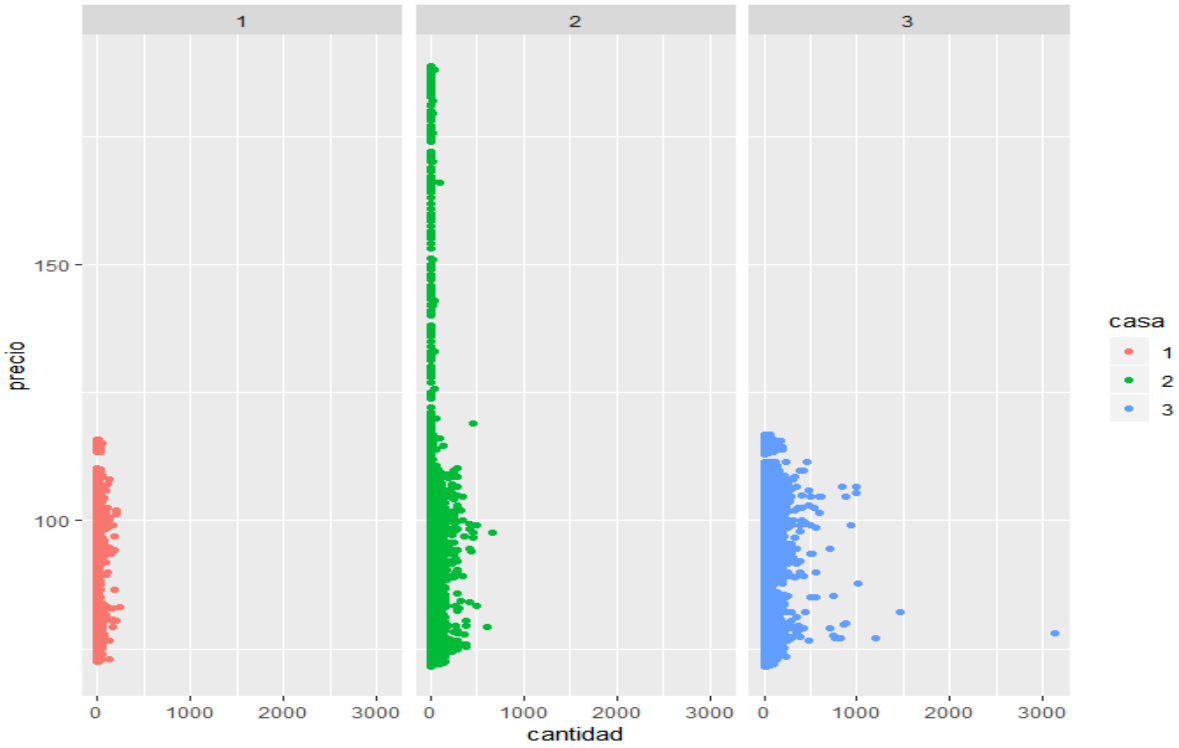

Figura 4.2: *Cantidad de criptomonedas en la transacción*

Observamos que la Casa 1 se caracteriza por realizar transacciones de una cantidad pequeña de criptomonedas al nivel de precios más habitual, mientras que la Casa 3 efectúa compraventas de una mayor cantidad de criptomonedas que las otras dos casas por un precio similar. En cambio, la Casa 2 se caracteriza por haber realizado transacciones de una suma considerable, pero no debido a la cantidad comercializada

pues no es superior a la de la Casa 3.

Una vez obtenida la base de datos correctamente diseñada, se divide entre el conjunto de entrenamiento, usado para construir los modelos, y el conjunto test, usado para evaluar posteriormente dichos modelos. El conjunto de entrenamiento está formado por las observaciones correspondientes a los primeros 20 días, es decir, del 21/11/2018 hasta el 11/12/2018. En cambio, el conjunto test está formado por las observaciones correspondientes al último día disponible, el 12/12/2018.

```
datos_train <- datos2 %>% filter(fecha!="2018-12-12")
datos_test <- datos2 %>% filter(fecha=="2018-12-12")
```

A continuación, realizamos un **estudio descriptivo** del conjunto entrenamiento.

```
summary(datos_train)
```

```
##      precio          cantidad         casa           fecha
##  Min.   : 71.57   Min.   :   0.0000   1: 15715   Min.   :2018-11-22
##  1st Qu.: 80.89   1st Qu.:   0.4765   2: 42034   1st Qu.:2018-11-27
##  Median : 94.01   Median :   1.8800   3:137799   Median :2018-12-02
##  Mean   : 92.23   Mean   :   7.0516              Mean   :2018-12-02
##  3rd Qu.:101.61   3rd Qu.:   5.0000              3rd Qu.:2018-12-07
##  Max.   :188.77   Max.   :3134.2355              Max.   :2018-12-11
##
##      tiempo            lagprecio
##  Min.   :1.543e+09   Min.   : 71.57
##  1st Qu.:1.543e+09   1st Qu.: 80.89
##  Median :1.544e+09   Median : 94.01
##  Mean   :1.544e+09   Mean   : 92.23
##  3rd Qu.:1.544e+09   3rd Qu.:101.61
##  Max.   :1.545e+09   Max.   :188.77
##                      NA's   :1
```

Apreciamos que el precio máximo, asociado a la Casa 2 (4.1), es mayor que el doble de la media y que la cantidad máxima, asociada a la Casa 3 (4.2), supera con mucha diferencia al resto. Por último, la Casa 3 es la casa de cambio que más transacciones ha realizado, seguida de la segunda y primera casa, respectivamente.

Seleccionamos las observaciones de la variable Precio en el conjunto test para realizar posteriormente la validación del modelo.

```
precios_reales <- datos_test %>% select(precio)
```

Para evaluar la bondad del ajuste de los modelos realizamos la suma de cuadrados de la diferencia entre los precios reales del 12/12/2018 y los precios predichos para ese mismo día. Para realizar dicho proceso creamos la siguiente función:

```r
bondad = function(a, b){
  bondad = sum((a - b)^2)
  cat("Bondad de ajuste:", bondad)
}
```

Con este conjunto de datos, a continuación, construiremos los modelos a partir de las distintas variables y aplicando las técnicas vistas en el segundo capítulo.

4.3. Modelos

Ante la gran variedad de posibles modelos a estudiar y las técnicas vistas en el segundo capítulo, a continuación, expondremos los modelos de mayor utilidad y con los que se obtengan predicciones de mayor acierto, suponiendo conocida la variable Lagprecio del conjunto test. No obstante, en el **anexo** de este trabajo, podremos encontrar una gran variedad de modelos sobre los que se aplican las diferentes técnicas descritas en el segundo capítulo.

A partir de algunos modelos se obtendrán predicciones adecuadas y fiables, mientras que otros modelos carecerán de estos resultados. Para los primeros modelos se calcularán las predicciones suponiendo desconocida la variable Lagprecio del conjunto test. Esto se hará para evidenciar la necesidad de ajustar el modelo tantas veces como sea posible en función del tiempo de captura de la información.

4.3.1. Precio ∼ Lagprecio + Tiempo. Regresión polinómica

A continuación, vamos a estudiar un modelo que relaciona el precio de la criptomoneda y el precio inmediatamente anterior y el momento de la transacción aplicando regresión polinómica.

Mediante la función **poly** se realiza regresión polinómica indicando el grado del polinomio. Mediante el siguiente bucle, en el que se construyen distintos modelos de regresión polinómica de grado 2, 3, 4 y 5, obtendremos el modelo con el que trabajaremos más adelante. Se elegirá el de mejor grado de bondad de ajuste.

```r
for (i in 2:5){
  for (j in 2:5){
  fit <- lm(precio ~ poly(tiempo,i) + poly(lagprecio,j),
            data=datos_train[-1,])
  predic1 = predict(fit, newdata=list(tiempo=datos_test$tiempo,
                                      lagprecio=datos_test$lagprecio))
  cat(". Polinomio Grado: Tiempo", i, " Lagprecio ",
      j, bondad(predic1, precios_reales), " ","\n")
  }
}
```

```
## Bondad de ajuste: 359.7643. Polinomio Grado: Tiempo 2  Lagprecio  2
```

```
## Bondad de ajuste: 254.2219. Polinomio Grado: Tiempo 2  Lagprecio  3
## Bondad de ajuste: 244.5325. Polinomio Grado: Tiempo 2  Lagprecio  4
## Bondad de ajuste: 248.6654. Polinomio Grado: Tiempo 2  Lagprecio  5
## Bondad de ajuste: 263.6619. Polinomio Grado: Tiempo 3  Lagprecio  2
## Bondad de ajuste: 241.5727. Polinomio Grado: Tiempo 3  Lagprecio  3
## Bondad de ajuste: 244.2842. Polinomio Grado: Tiempo 3  Lagprecio  4
## Bondad de ajuste: 245.7616. Polinomio Grado: Tiempo 3  Lagprecio  5
## Bondad de ajuste: 325.9791. Polinomio Grado: Tiempo 4  Lagprecio  2
## Bondad de ajuste: 398.6155. Polinomio Grado: Tiempo 4  Lagprecio  3
## Bondad de ajuste: 286.4181. Polinomio Grado: Tiempo 4  Lagprecio  4
## Bondad de ajuste: 282.2116. Polinomio Grado: Tiempo 4  Lagprecio  5
## Bondad de ajuste: 433.8722. Polinomio Grado: Tiempo 5  Lagprecio  2
## Bondad de ajuste: 304.8913. Polinomio Grado: Tiempo 5  Lagprecio  3
## Bondad de ajuste: 244.497. Polinomio Grado: Tiempo 5  Lagprecio  4
## Bondad de ajuste: 283.6591. Polinomio Grado: Tiempo 5  Lagprecio  5
```

Por tanto, trabajaremos definiendo un polinomio de grado 3 para cada una de las variables. Este modelo se construye como sigue:

```
fit <- lm(precio ~ poly(tiempo,3) + poly(lagprecio,3),
          data=datos_train[-1,])
summary(fit)

##
## Call:
## lm(formula = precio ~ poly(tiempo, 3) + poly(lagprecio, 3), data = datos_train[-1
##     ])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -80.496  -0.117  -0.013   0.098 103.645
##
## Coefficients:
##                         Estimate Std. Error   t value Pr(>|t|)
## (Intercept)           9.223e+01  2.938e-03 31391.419  < 2e-16 ***
## poly(tiempo, 3)1      2.080e+01  2.481e+00     8.385  < 2e-16 ***
## poly(tiempo, 3)2      2.819e+00  1.521e+00     1.853   0.0638 .
## poly(tiempo, 3)3     -7.395e+00  1.325e+00    -5.580  2.4e-08 ***
## poly(lagprecio, 3)1   4.965e+03  2.479e+00  2002.627  < 2e-16 ***
## poly(lagprecio, 3)2 -1.499e+02  1.501e+00   -99.830  < 2e-16 ***
## poly(lagprecio, 3)3 -9.677e+01  1.349e+00   -71.710  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.299 on 195540 degrees of freedom
## Multiple R-squared:  0.9867,Adjusted R-squared:  0.9867
## F-statistic: 2.42e+06 on 6 and 195540 DF,  p-value: < 2.2e-16
```

Si establecemos el nivel de significación al 10 % ($\alpha = 0.1$), podemos afirmar que las componentes del polinomio de grado 3 para cada variable son significativas.

```
plot(fit)
```

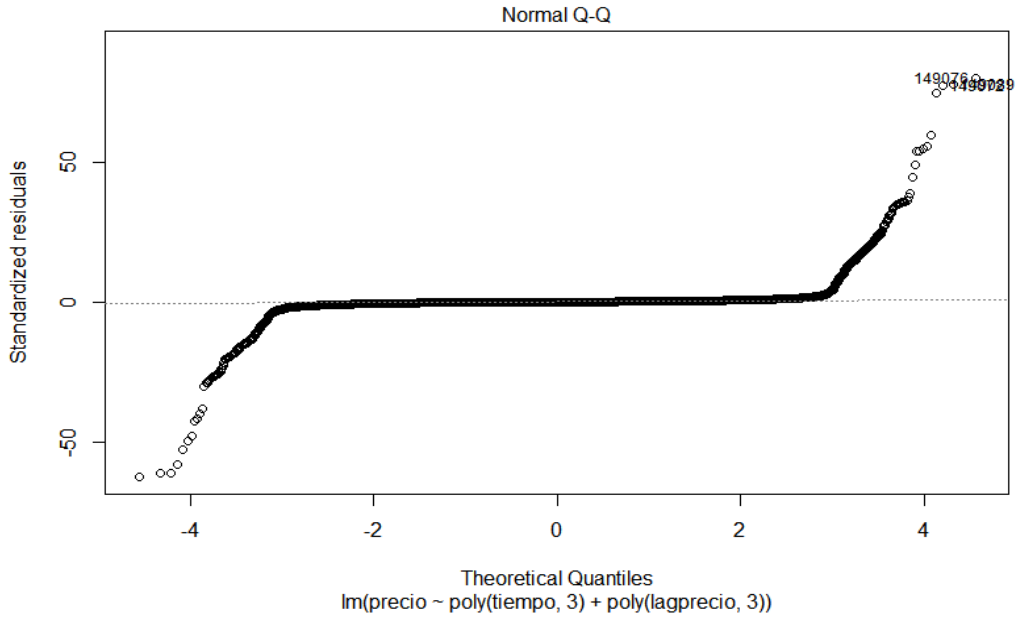

Figura 4.3: *Normalidad*

En el gráfico 4.3 observamos que la normalidad no se mantiene en las colas. En el gráfico 4.4 comprobamos la existencia de valores atípicos en la base de datos. Vamos a estudiar a qué se debe esta circunstancia.

```
datos_train[149071,]

##        precio cantidad casa      fecha     tiempo lagprecio
## 149071   87.1    0.315    2 2018-12-07 1544206551    187.67

datos_train[149082,]

##        precio cantidad casa      fecha     tiempo lagprecio
## 149082  87.56 2.252524    2 2018-12-07 1544206554       185

datos_train[149088,]

##        precio cantidad casa      fecha     tiempo lagprecio
## 149088  86.95   11.006    2 2018-12-07 1544206547    183.65
```

Como podemos observar en el resumen descriptivo del conjunto entrenamiento, la variable Lapgrecio tiene un valor máximo de 188,77. Por tanto, estas tres observaciones cuentan con un valor muy cercano al máximo de la variable Lagprecio, lo que puede propiciar su consideración como valores atípicos.

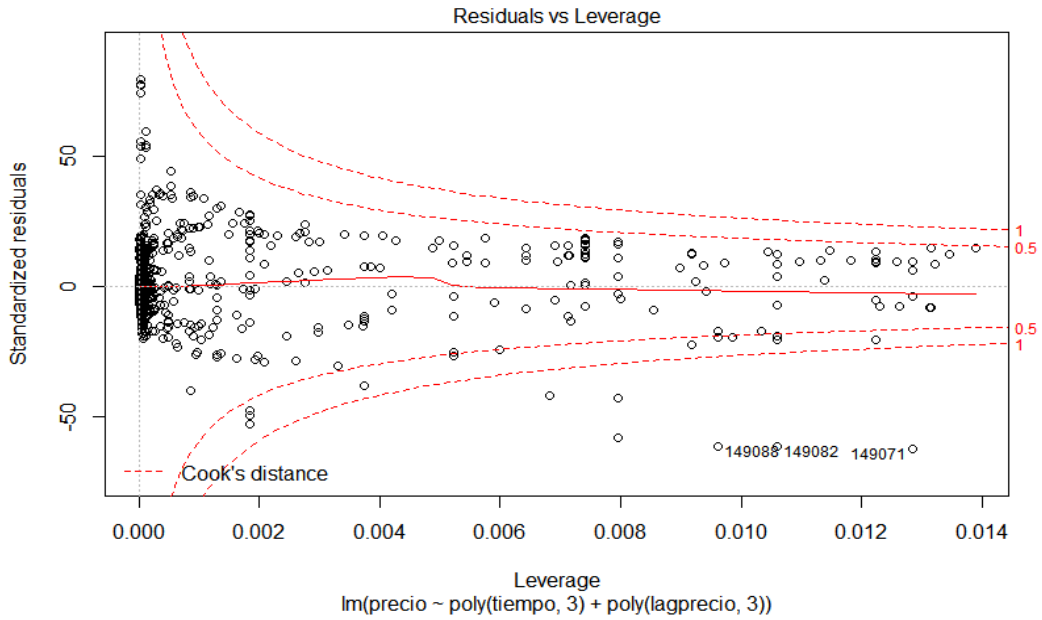

Figura 4.4: *Valores atípicos*

Predicción supuesto Lagprecio conocido

A continuación, obtenemos las predicciones de los precios para el día 12/12/2018 suponiendo conocida la variable Lagprecio del conjunto test.

```
predic1 = predict(fit, newdata=list(tiempo=datos_test$tiempo,
                                    lagprecio=datos_test$lagprecio))
bondad(predic1, precios_reales)

## Bondad de ajuste: 241.5727
```

Realizamos la representación gráfica (4.5) de los precios reales comparados con los precios predichos anteriormente.

```
plot(precios_reales$precio, predic1, ylab="Predicciones",
     xlab="Precios reales", main="Comparación. Precios reales y Predicción")
```

Por lo general, observamos que la nube de puntos se encuentra concentrada en la diagonal del gráfico. Este resultado hace suponer que las predicciones obtenidas son adecuadas.

Pedicción supuesto Lagprecio desconocido

A continuación, obtenemos las predicciones de los precios para el día 12/12/2018 suponiendo desconocida la variable Lagprecio del conjunto test. Para calcular la predicción del primer dato del día 12/12/2018 necesitamos el valor de la variable Lagprecio para la última entrada del día anterior. Esta primera predicción también es el valor de la variable Lagprecio necesario para realizar la segunda predicción, y así sucesivamente.

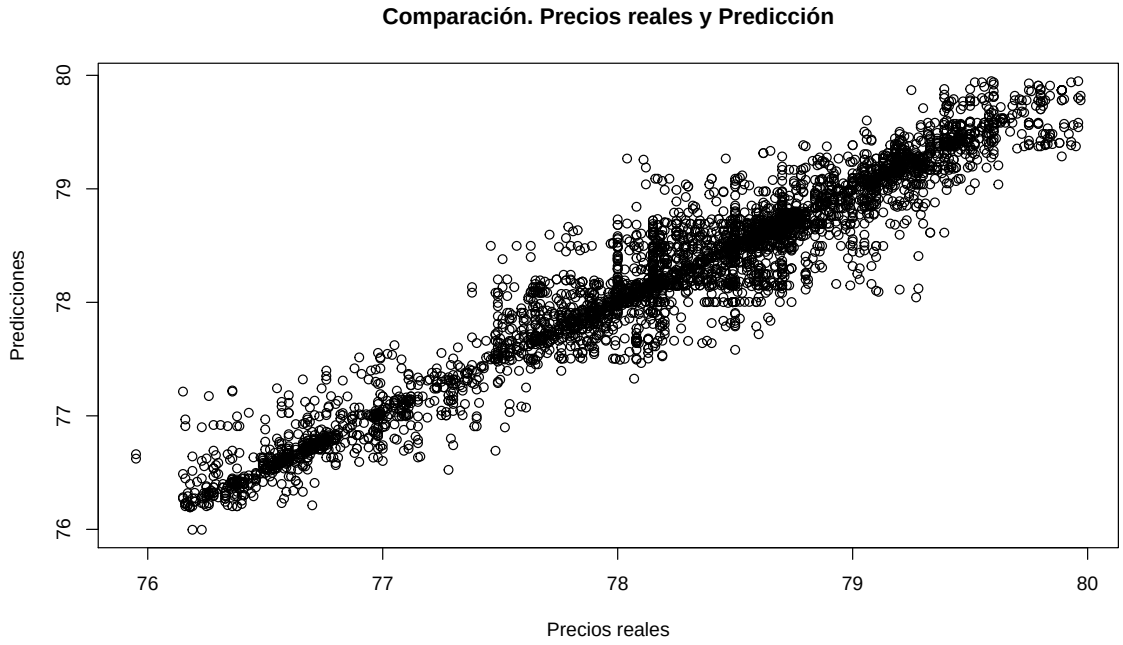

Figura 4.5: Precio $\sim$ *Lagprecio* + *Tiempo*. *Regresión polinómica*. *Lagprecio conocido*

```
lagprecio_test = rep(NA,length(datos_test$precio)+1)
predic2 = rep(NA,length(datos_test$precio))

lagprecio_test[1] = datos_train$precio[length(datos_train$precio)]
lagprecio_test[2] = predict(fit,
                            newdata=list(lagprecio=lagprecio_test[1],
                                         tiempo=datos_test$tiempo[1]))
predic2[1] = lagprecio_test[2]

for (i in 2:nrow(datos_test)){
  predic2[i] = predict(fit,
                       newdata=list(lagprecio=lagprecio_test[i],
                                    tiempo=datos_test$tiempo[i]))
  lagprecio_test[i+1]= predic2[i]
}
bondad(predic2, precios_reales)

## Bondad de ajuste: 4692.854
```

Como era de esperar, obtenemos un peor grado de bondad de ajuste.

Realizamos la representación gráfica (4.6) de los precios reales comparados con los precios predichos anteriormente

```
plot(precios_reales$precio, predic2, ylab="Predicciones",
     xlab="Precios reales", main="Comparación. Precios reales y Predicción")
```

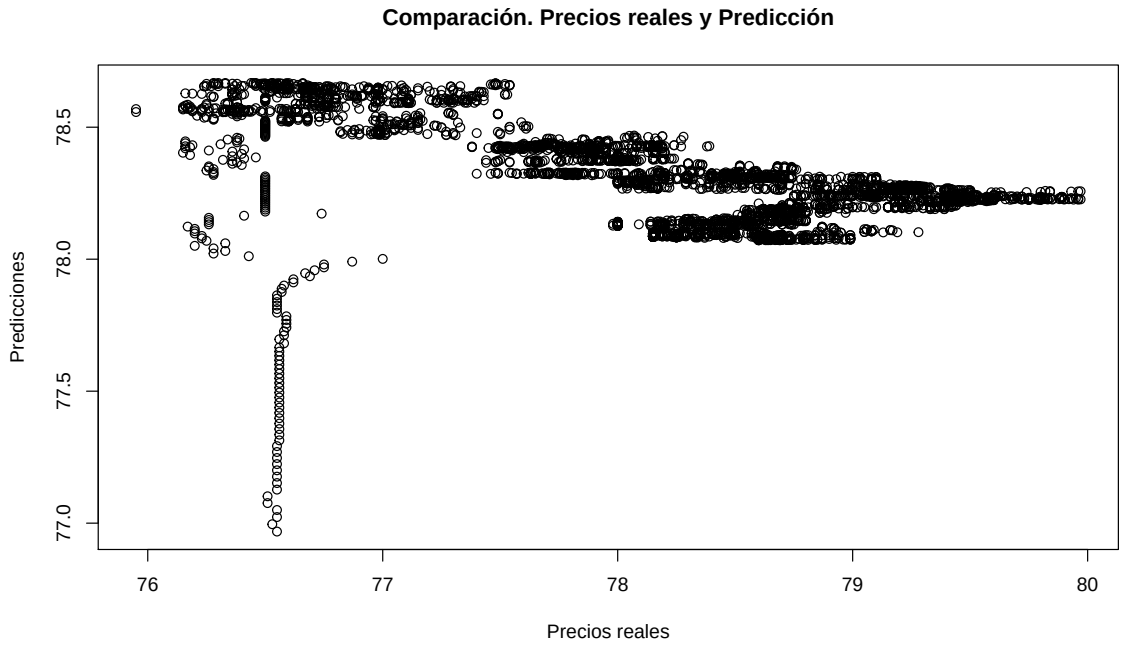

Figura 4.6: Precio $\sim$ *Lagprecio* + *Tiempo. Regresión polinómica. Lagprecio desconocido*

En este gráfico apreciamos el empeoramiento de las predicciones con respecto a las anteriores. Por ejemplo, para los precios entre 76 y 78, se obtienen, mayoritariamente, predicciones por encima de los 78 dólares. Para los precios superiores a 79, se obtienen predicciones por debajo de 78,5 dólares.

Una vez obtenidas ambas predicciones, realizamos la representación gráfica (4.7) de las dos a lo largo del día 12/12/2018.

```
par(mfrow =c(1,2))

a = data.frame(precio = precios_reales, predic1,
               casa=datos_test$casa, time=datos_test$tiempo)
plot(a$time, a$precio, ylim=c(min(a$predic1,  a$precio),
                              max(a$predic1, a$precio)), lwd=2,
     xlab="Tiempo", ylab="Precio",
     main="Lagprecio conocido")
points(a$time, a$predic1, col =" blue", lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"), pch=c(1,1))

b = data.frame(precio = precios_reales, predic2,
               casa=datos_test$casa, time=datos_test$tiempo)
```

```
plot(b$time, b$precio, ylim=c(min(b$predic2,  b$precio),
                              max(b$predic2, b$precio)), lwd=2,
     xlab="Tiempo", ylab="Precio",
     main="Lagprecio desconocido")
points(b$time, b$predic2, col =" blue", lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"), pch=c(1,1))
```

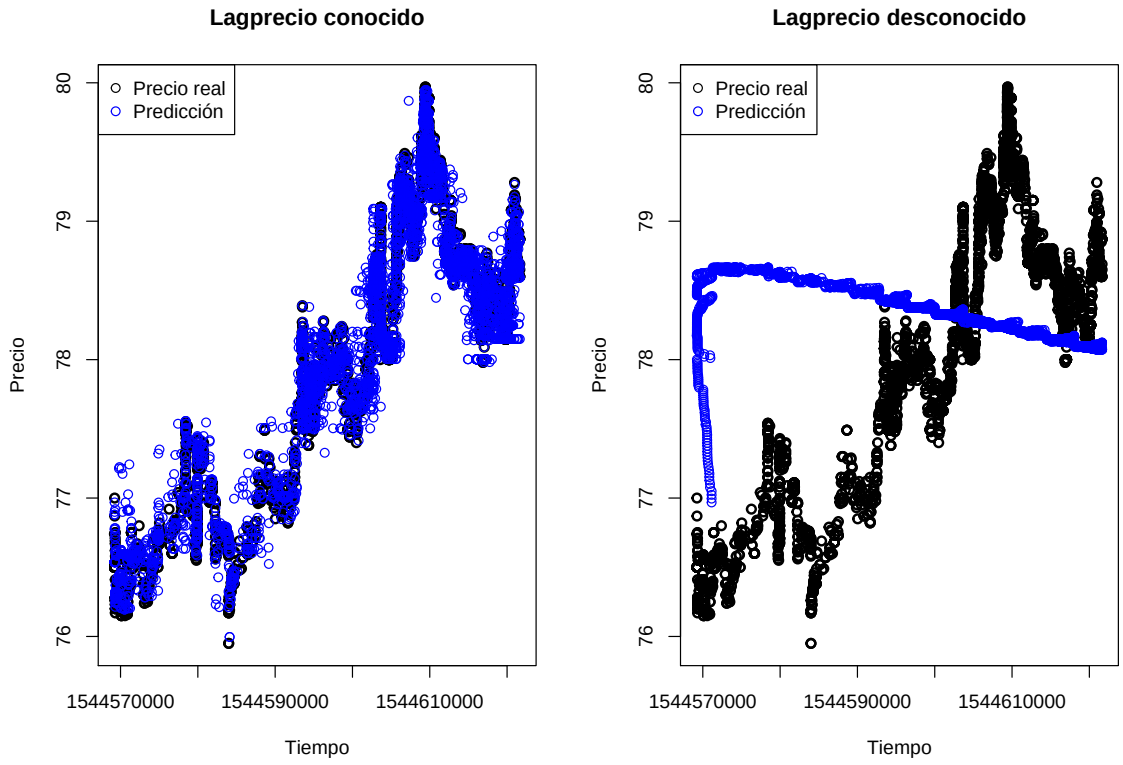

Figura 4.7: Precio $\sim$ *Lagprecio* + *Tiempo*. *Regresión polinómica*. *Predicciones*

```
par(mfrow =c(1,1))
```

Observamos que, supuesto Lagprecio desconocido, la evolución de las predicciones a lo largo del día 12/12/2018 no se asemeja a la evolución de los precios reales, pues, al comiezo del día, aumenta casi 2 dólares el precio y el resto del día va disminuyendo lentamente hasta quedarse por encima de los 78. En cambio, el precio real, a lo largo del día, sufre constantes subidas y bajadas, pero llega a alcanzar los 80 dólares y termina por encima de los 79. Estas cotas no son alcanzadas por la predicción supuesto Lagprecio desconocido.

Tal y como podemos observar en los gráficos 4.8, 4.9 y 4.10, si representamos los resultados obtenidos mediante ambas predicciones, diferenciando entre las tres casas de cambio, se manifiesta con mayor claridad la pérdida de calidad de las predicciones realizadas suponiendo desconocida la variable Lagprecio del conjunto test.

```
par(mfrow =c(1,2))

  d <- a %>% filter(casa==1)
  plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
                                max(d$predic1, d$precio)),
       pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
  title(paste0("Lagprecio conocido Casa 1"))
  points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
  legend("topleft", legend = c("Precio real","Predicción"),
         col=c("black","blue"),  pch=c(1,1))

  c <- b %>% filter(casa==1)
  plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                                max(c$predic2, c$precio)),
       lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
  title(paste0("Lagprecio desconocido Casa 1"))
  points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
  legend("topleft", legend = c("Precio real","Predicción"),
         col=c("black","blue"),  pch=c(1,1))
```

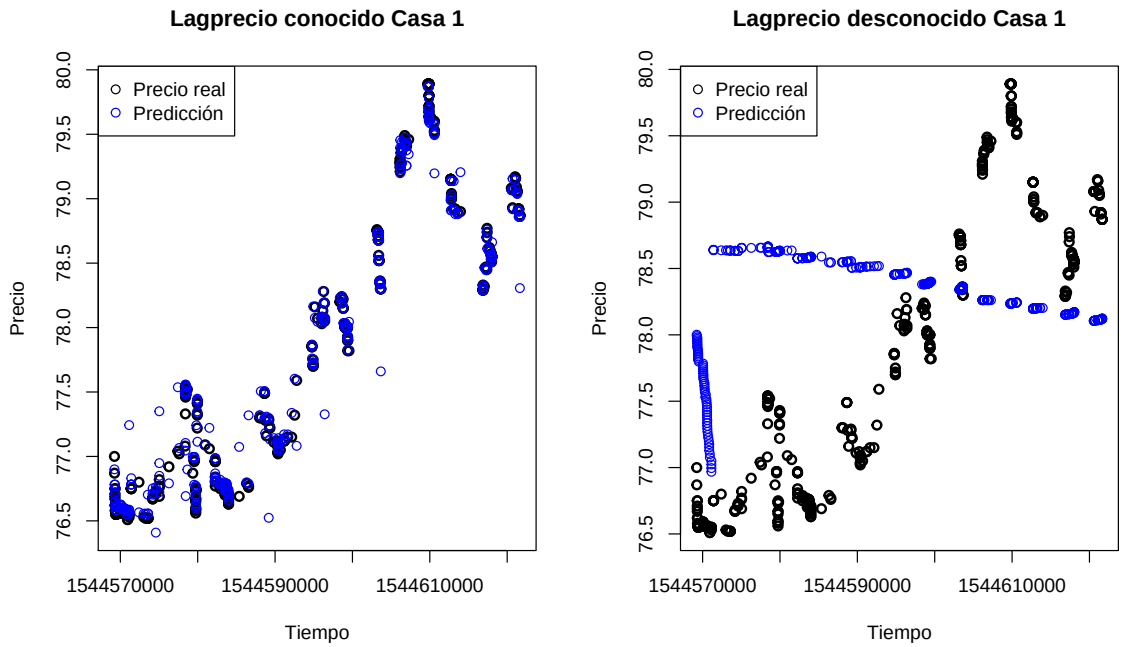

Figura 4.8: Precio $\sim$ *Lagprecio* + *Tiempo*. *Regresión polinómica*. *Casa* 1

```
par(mfrow =c(1,1))
```

```
par(mfrow =c(1,2))
d <- a %>% filter(casa==2)
plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
                              max(d$predic1, d$precio)),
     pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio conocido Casa 2"))
points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))

c <- b %>% filter(casa==2)
plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                              max(c$predic2, c$precio)),
     lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio desconocido Casa 2"))
points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))
```

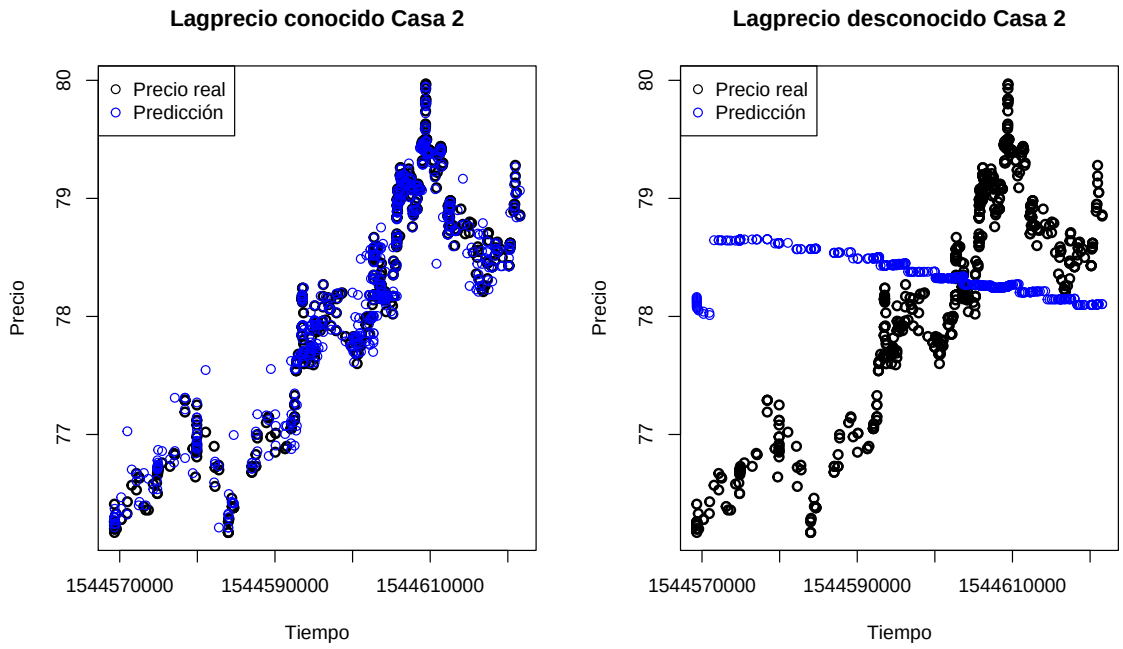

Figura 4.9: Precio $\sim$ *Lagprecio* + *Tiempo*. *Regresión polinómica*. *Casa* 2

```
par(mfrow =c(1,1))
```

```
par(mfrow =c(1,2))

d <- a %>% filter(casa==3)
plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
                              max(d$predic1, d$precio)),
     pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio conocido Casa 3"))
points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))

c <- b %>% filter(casa==3)
plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                              max(c$predic2, c$precio)),
     lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio desconocido Casa 3"))
points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))
```

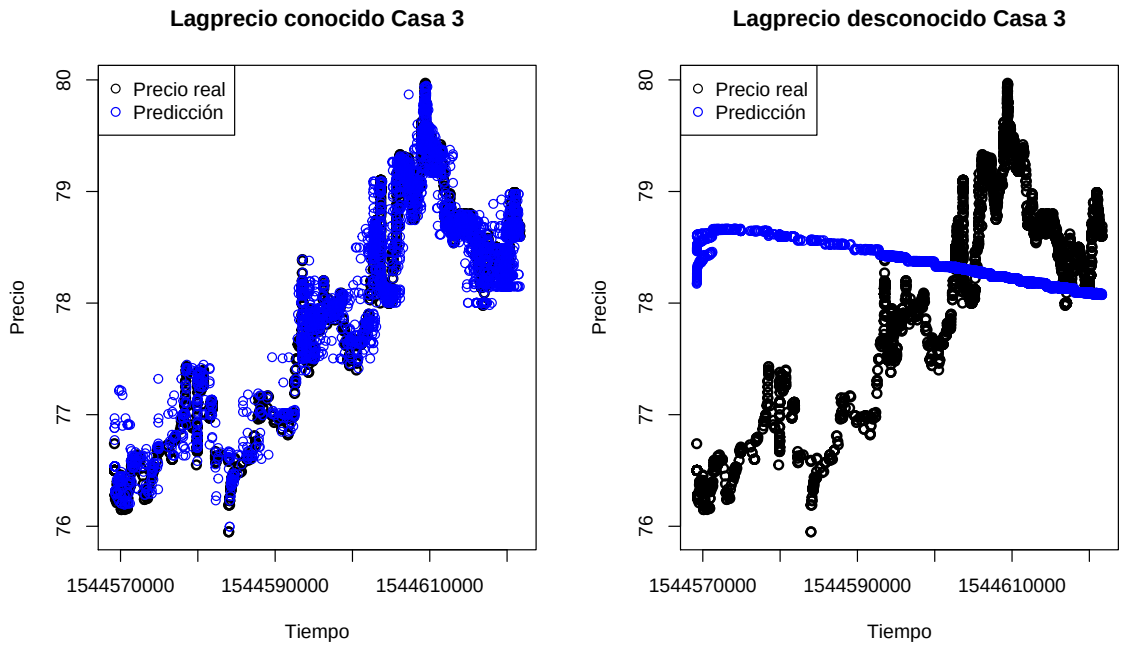

Figura 4.10: Precio $\sim$ *Lagprecio* + *Tiempo*. *Regresión polinómica*. *Casa* 3

```
par(mfrow =c(1,1))
```

A continuación, realizamos un gráfico evolutivo (4.11) del precio de la criptomoneda incluyendo ambas predicciones.

```r
plot(datos_train$tiempo, datos_train$precio,
     xlim=c(min(datos_train$tiempo), max(datos_test$tiempo)),
     type="l", main="Evolución del precio",
     xlab="Tiempo", ylab="Precio")
lines(datos_test$tiempo, predic1, col ="green", lwd=2)
lines(datos_test$tiempo, predic2, col ="red", lwd=2)
legend("topleft",
       legend = c("Lagprecio conocido","Lagprecio desconocido"),
       col=c("green","red"), lty=1)
```

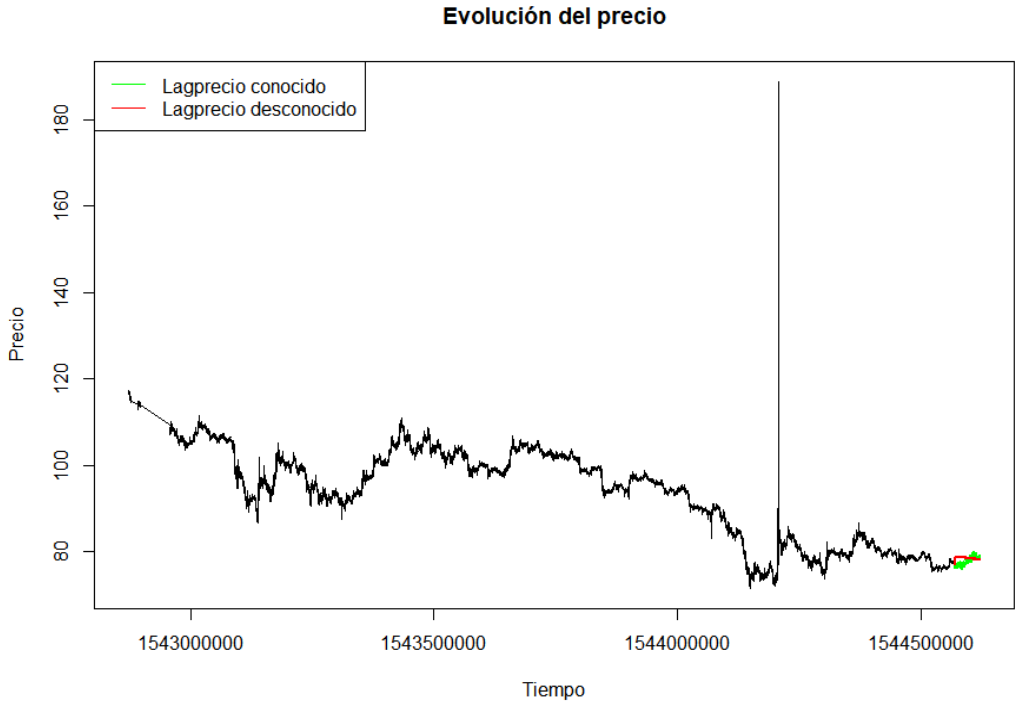

Figura 4.11: *Precio* $\sim$ *Lagprecio* + *Tiempo. Regresión polinómica. Evolución del precio*

Observamos que, supuesto Lagprecio conocido, el precio sufre una leve bajada al comienzo del día, pero va aumentando hasta alcanzar el nivel del precios del día anterior y refleja las constantes subidas y bajadas del precio. Por otro lado, la predicción supuesto Lapgrecio desconocido no refleja las numerosas variaciones sufridas por el precio y, tras sufrir una subida de más de 1.5 dólares, va disminuyendo hasta finalizar el día.

4.3.2. Precio $\sim$ Lagprecio. Spline natural

Queremos estudiar la relación entre el precio de la criptomoneda y el precio inmediatamente anterior y, esta vez, construimos el modelo haciendo uso de los splines naturales.

Mediante la función **ns** se aplican splines naturales y con el argumento **df** se indican los grados de libertad sobre la variable Lagprecio. Si $df = 4$, se trabaja con 3 nodos. Si $df = 5$, se trabaja con 4 nodos y así sucesivamente.

Mediante el siguiente bucle en el que construimos distintos modelos aplicando splines naturales con grados de libertad de 2 a 7, obtendremos el modelo con el que trabajaremos a continuación. Elegiremos aquel con mejor grado de bondad de ajuste.

```
for (i in 2:7){
  fit <- lm(precio ~ ns(lagprecio, df=i), data=datos_train)
  predic1 = predict(fit, newdata=list(lagprecio=datos_test$lagprecio))
  cat(". Spline natural con", i, "grados de libertad",
      bondad(predic1, precios_reales), " ","\n")
}
```

```
## Bondad de ajuste: 262.2437. Spline natural con 2 grados de libertad
## Bondad de ajuste: 240.5785. Spline natural con 3 grados de libertad
## Bondad de ajuste: 253.2913. Spline natural con 4 grados de libertad
## Bondad de ajuste: 242.4333. Spline natural con 5 grados de libertad
## Bondad de ajuste: 242.6725. Spline natural con 6 grados de libertad
## Bondad de ajuste: 243.4976. Spline natural con 7 grados de libertad
```

Trabajaremos con el modelo construido a partir de la aplicación de splines naturales con 3 grados de libertad y, por tanto, 2 nodos.

```
attr(ns(datos_train$lagprecio,df=3), "knots")
```

```
## 33.33333% 66.66667%
##      84.9      99.0
```

Los nodos están situados en los valores 84,9 y 99 de la variable Lapgrecio, correspondientes al percentil 33 y 66. En el gráfico expuesto a continuación (4.12) podemos observar la localización de los nodos en el recorrido de la variable Lagprecio.

```
plot(datos_train$lagprecio, datos_train$precio)
abline(v=c(84.9, 99), col="red")
```

Este modelo se construye como sigue:

```
fit=lm(precio~ns(lagprecio,df=3),data=datos_train)
summary(fit)
```

```
##
## Call:
## lm(formula = precio ~ ns(lagprecio, df = 3), data = datos_train)
##
## Residuals:
```

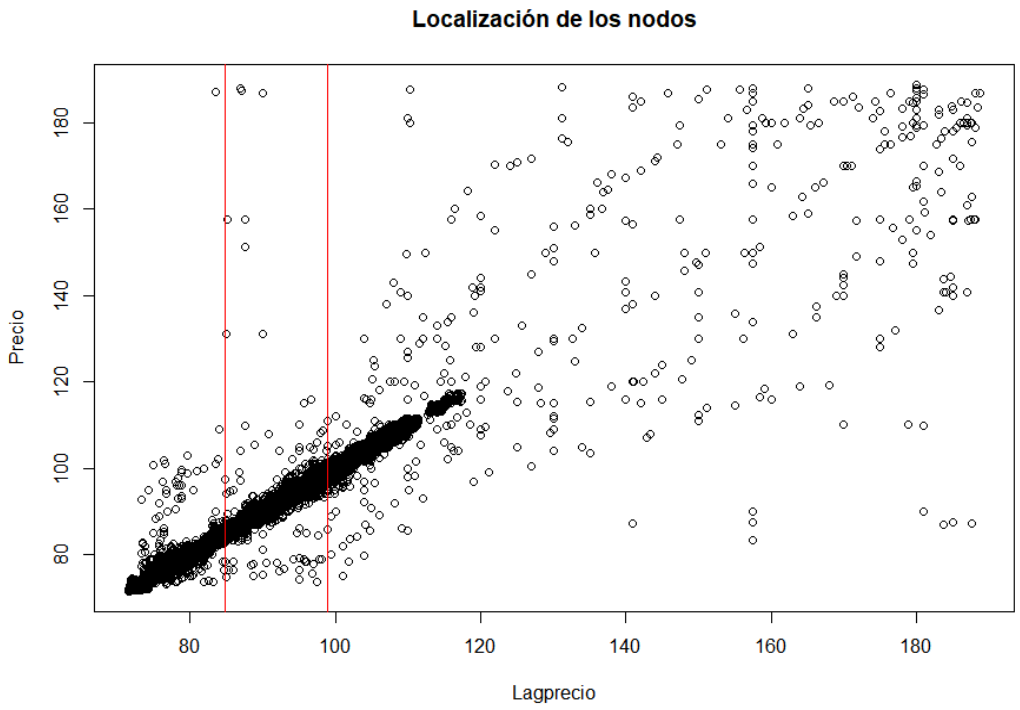

Figura 4.12: *Precio* $\sim$ *Lagprecio. Splines naturales. Localización de nodos*

```
##     Min      1Q  Median      3Q     Max
## -85.821  -0.129  -0.016   0.113 103.730
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              71.79343    0.01142  6284.1   <2e-16 ***
## ns(lagprecio, df = 3)1 48.60379    0.02325  2090.5   <2e-16 ***
## ns(lagprecio, df = 3)2 90.84029    0.07005  1296.8   <2e-16 ***
## ns(lagprecio, df = 3)3 95.93332    0.12392   774.1   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.312 on 195543 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.9865,Adjusted R-squared:  0.9865
## F-statistic: 4.747e+06 on 3 and 195543 DF,  p-value: < 2.2e-16
```

Observamos que las tres componentes del spline para la variable Lagprecio son significativas.

Predicción supuesto Lagprecio conocido

A continuación, obtenemos las predicciones de los precios para el día 12/12/2018 suponiendo conocida la variable Lagprecio del conjunto test.

```
predic1 = predict(fit, newdata=list(lagprecio=datos_test$lagprecio))
bondad(predic1, precios_reales)

## Bondad de ajuste: 240.5785
```

Realizamos la representación gráfica (4.13) de los precios reales comparados con los precios predichos anteriormente

```
plot(precios_reales$precio, predic1, ylab="Predicciones",
     xlab="Precios reales",
     main="Comparación. Precios reales y Predicción")
```

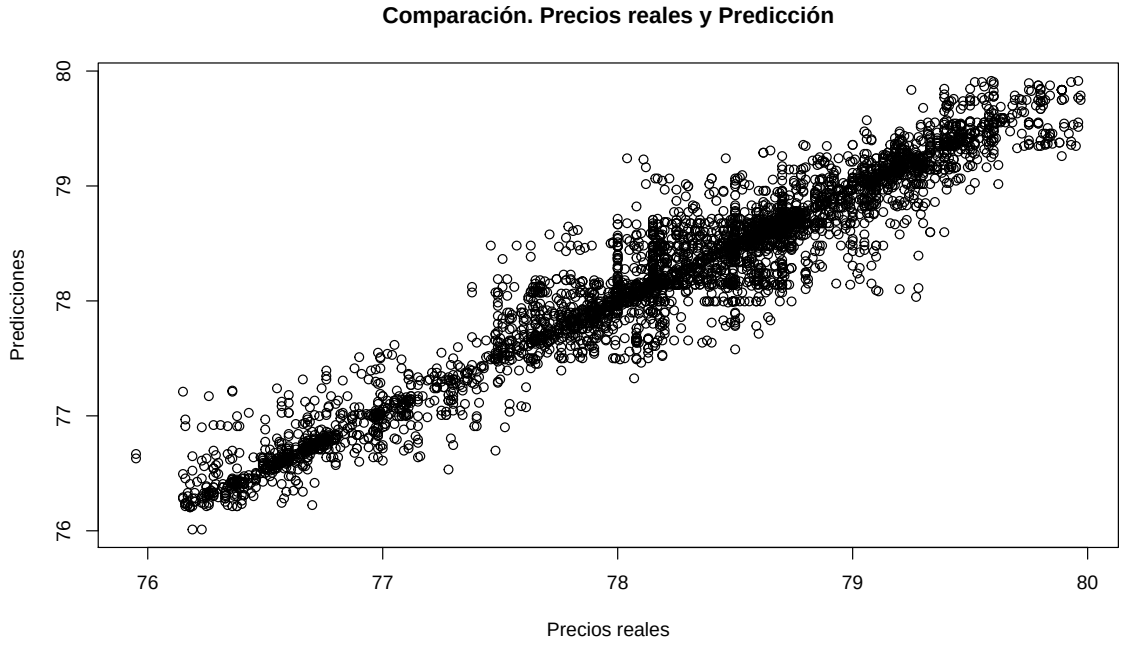

Figura 4.13: Precio $\sim$ *Lagprecio. Splines naturales. Lagprecio conocido*

Por lo general, observamos que la nube de puntos se encuentra concentrada en la diagonal del gráfico lo que hace pensar que las predicciones obtenidas son adecuadas.

Predicción supuesto Lagprecio desconocido

A continuación, obtenemos las predicciones de los precios para el día 12/12/2018 suponiendo desconocida la variable Lagprecio del conjunto test. El cálculo se realiza de igual manera que para el modelo anterior.

```
lagprecio_test = rep(NA,length(datos_test$precio)+1)
predic2 = rep(NA,length(datos_test$precio))

lagprecio_test[1] = datos_train$precio[length(datos_train$precio)]
```

```
lagprecio_test[2] = predict(fit, newdata=list(lagprecio=lagprecio_test[1]))
predic2[1] = lagprecio_test[2]

for (i in 2:nrow(datos_test)){
  predic2[i] = predict(fit, newdata=list(lagprecio=lagprecio_test[i]))

  lagprecio_test[i+1]= predic2[i]
}
bondad(predic2, precios_reales)

## Bondad de ajuste: 4156.316
```

Como era de esperar, obtenemos un peor grado de bondad de ajuste.

Realizamos la representación gráfica (4.14) de los precios reales comparados con los precios predichos anteriormente.

```
plot(precios_reales$precio, predic2, ylab="Predicciones",
    xlab="Precios reales",
    main="Comparación. Precios reales y Predicción")
```

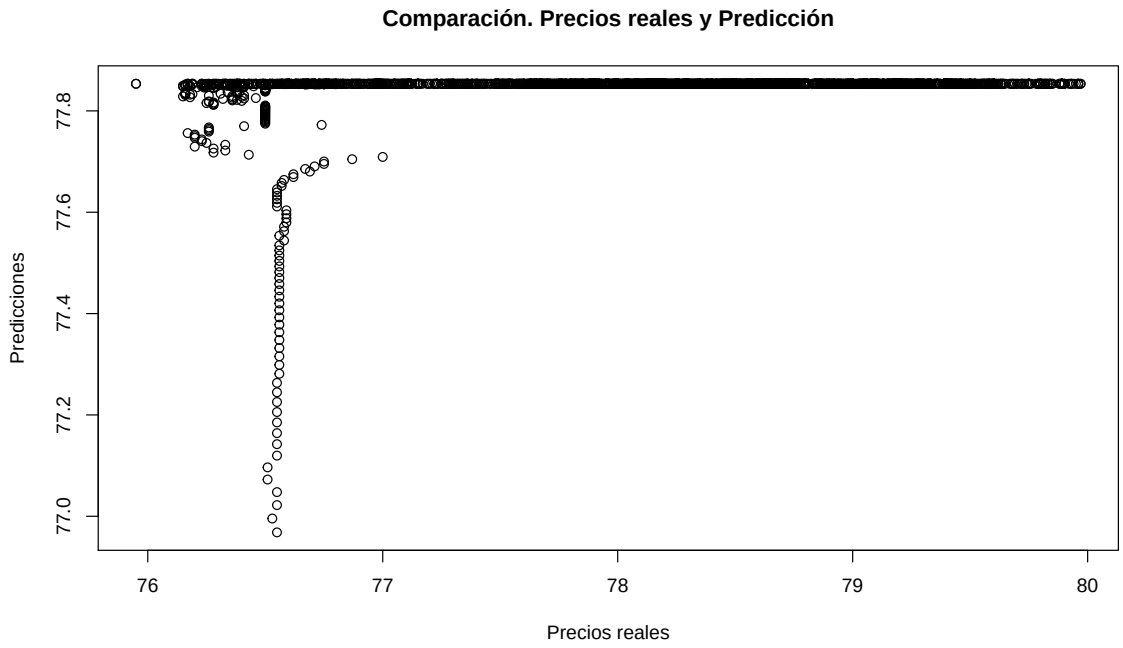

Figura 4.14: Precio $\sim$ *Lagprecio. Splines naturales. Lagprecio desconocido*

En este gráfico apreciamos el deterioro de las predicciones con respecto a las anteriores. Para las criptomonedas cuyos precios se encuentran entre 76 y 77 dólares, obtenemos predicciones superiores a los 77 dólares. Además, para las criptomonedas cuyos precios oscilan entre 78 y 80 dólares, obtenemos predicciones inferiores a los 78

dólares.

Una vez obtenidas ambas predicciones, realizamos la representación gráfica (4.15) de las dos a lo largo del día 12/12/2018.

```
par(mfrow =c(1,2))

a = data.frame(precio = precios_reales, predic1,
               casa=datos_test$casa, time=datos_test$tiempo)
plot(a$time, a$precio, ylim=c(min(a$predic1,  a$precio),
                              max(a$predic1, a$precio)), lwd=2,
     xlab="Tiempo", ylab="Precio",
     main="Lagprecio conocido")
points(a$time, a$predic1, col =" blue", lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"), pch=c(1,1))

b = data.frame(precio = precios_reales, predic2,
               casa=datos_test$casa, time=datos_test$tiempo)
plot(b$time, b$precio, ylim=c(min(b$predic2,  b$precio),
                              max(b$predic2, b$precio)), lwd=2,
     xlab="Tiempo", ylab="Precio",
     main="Lagprecio desconocido")
points(b$time, b$predic2, col =" blue", lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"), pch=c(1,1))
```

```
par(mfrow =c(1,1))
```

Al igual que para el anterior modelo, supuesto Lagprecio desconocido, la evolución de las predicciones a lo largo del día 12/12/2018 no se asemeja a la evolución de los precios reales ya que, al comienzo del día, aumenta el precio casi 1 dólar y el resto del día se mantiene constante por debajo de los 78 dólares. En cambio, el precio real, a lo largo del día, sufre constantes subidas y bajadas llegando a alcanzar los 80 dólares y terminando por encima de los 79. Estas cotas no son alcanzadas por la predicción supuesto Lagprecio desconocido.

Tal y como podemos observar en los gráficos 4.16, 4.17 y 4.18, si representamos los resultados obtenidos mediante ambas predicciones, diferenciando entre las tres casas de cambio, se manifiesta con mayor claridad la pérdida de calidad de las predicciones realizadas suponiendo desconocida la variable Lagprecio del conjunto test.

```
par(mfrow =c(1,2))

  d <- a %>% filter(casa==1)
  plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
```

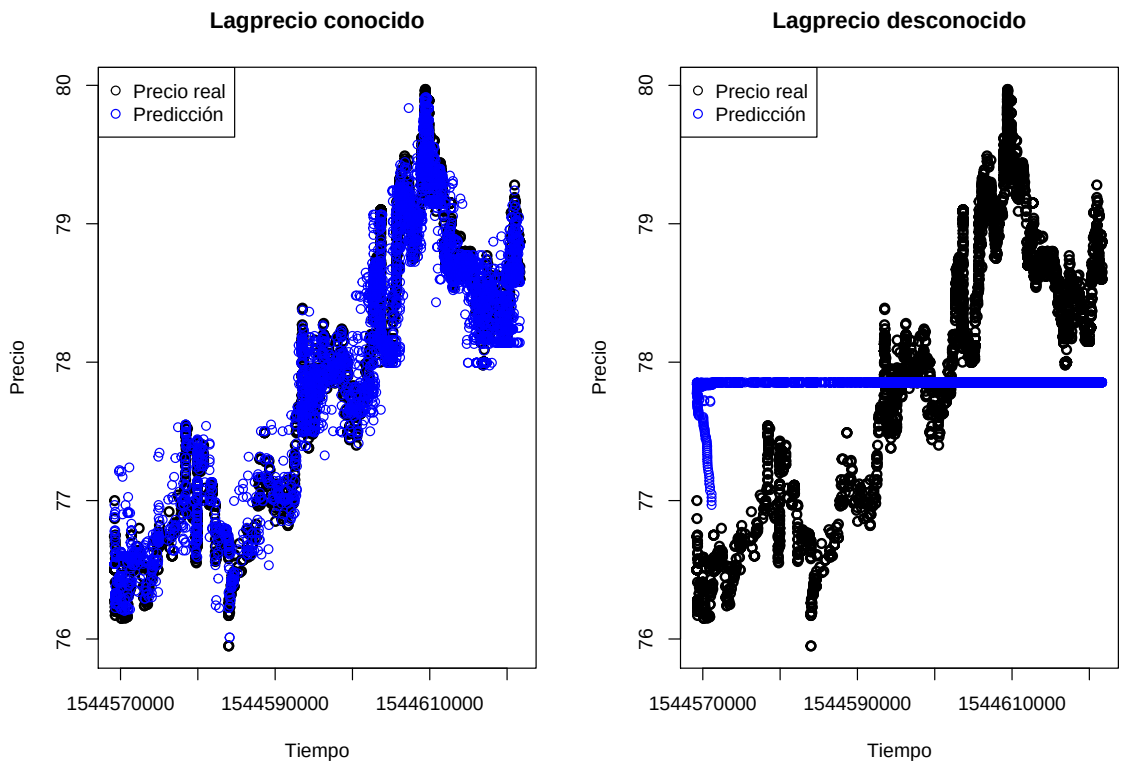

Figura 4.15: Precio $\sim$ *Lagprecio. Splines naturales. Predicciones*

```
                              max(d$predic1, d$precio)),
     pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
  title(paste0("Lagprecio conocido Casa 1"))
  points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
  legend("topleft", legend = c("Precio real","Predicción"),
         col=c("black","blue"),  pch=c(1,1))

  c <- b %>% filter(casa==1)
  plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                              max(c$predic2, c$precio)),
     lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
  title(paste0("Lagprecio desconocido Casa 1"))
  points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
  legend("topleft", legend = c("Precio real","Predicción"),
         col=c("black","blue"),  pch=c(1,1))
```

```
par(mfrow =c(1,1))
```

```
par(mfrow =c(1,2))
d <- a %>% filter(casa==2)
```

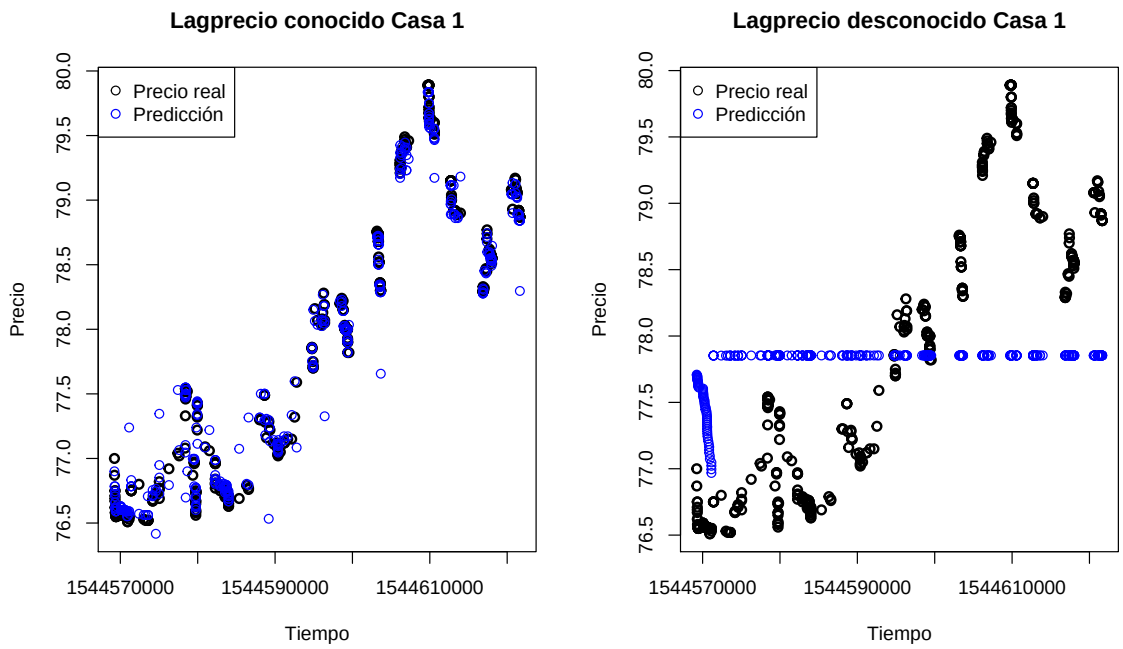

Figura 4.16: Precio $\sim$ *Lagprecio. Splines naturales. Casa* 1

```
plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
                              max(d$predic1, d$precio)),
     pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio conocido Casa 2"))
points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))

c <- b %>% filter(casa==2)
plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                              max(c$predic2, c$precio)),
     lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio desconocido Casa 2"))
points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))
```

```
par(mfrow =c(1,1))
```

```
par(mfrow =c(1,2))

d <- a %>% filter(casa==3)
plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
```

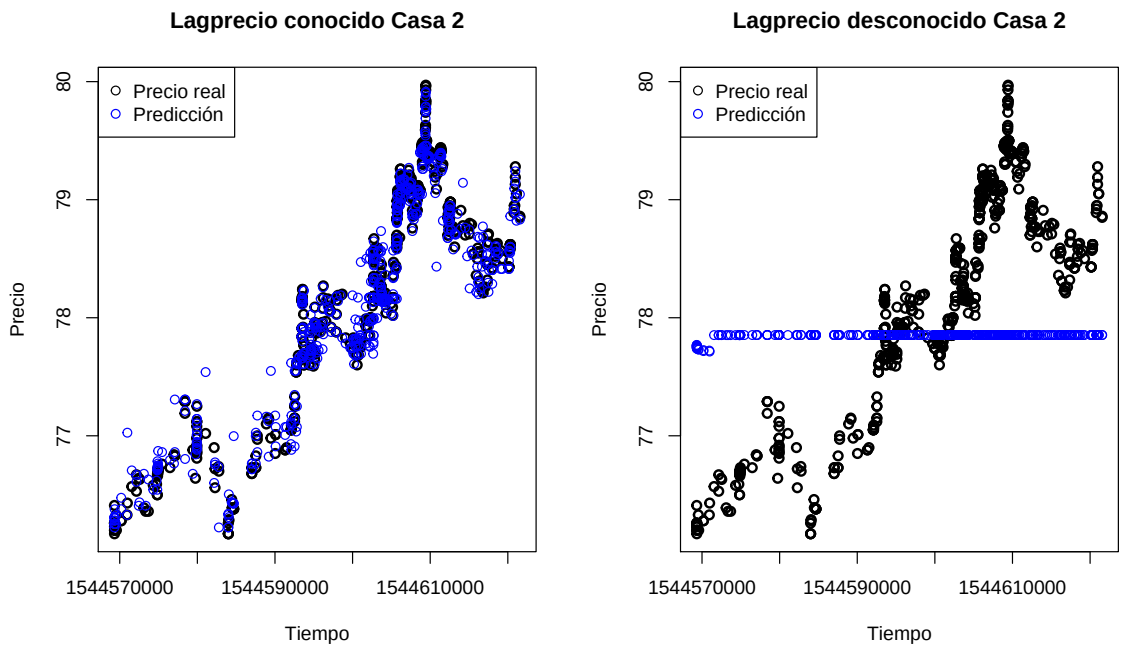

Figura 4.17: Precio $\sim$ *Lagprecio. Splines naturales. Casa* 2

```
                                max(d$predic1, d$precio)),
     pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio conocido Casa 3"))
points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))

c <- b %>% filter(casa==3)
plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                                max(c$predic2, c$precio)),
     lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio desconocido Casa 3"))
points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))
```

```
par(mfrow =c(1,1))
```

A continuación, realizamos un gráfico evolutivo (4.19) del precio de la criptomoneda incluyendo ambas predicciones.

```
plot(datos_train$tiempo, datos_train$precio,
     xlim=c(min(datos_train$tiempo), max(datos_test$tiempo)),
     type="l", main="Evolución del precio",
```

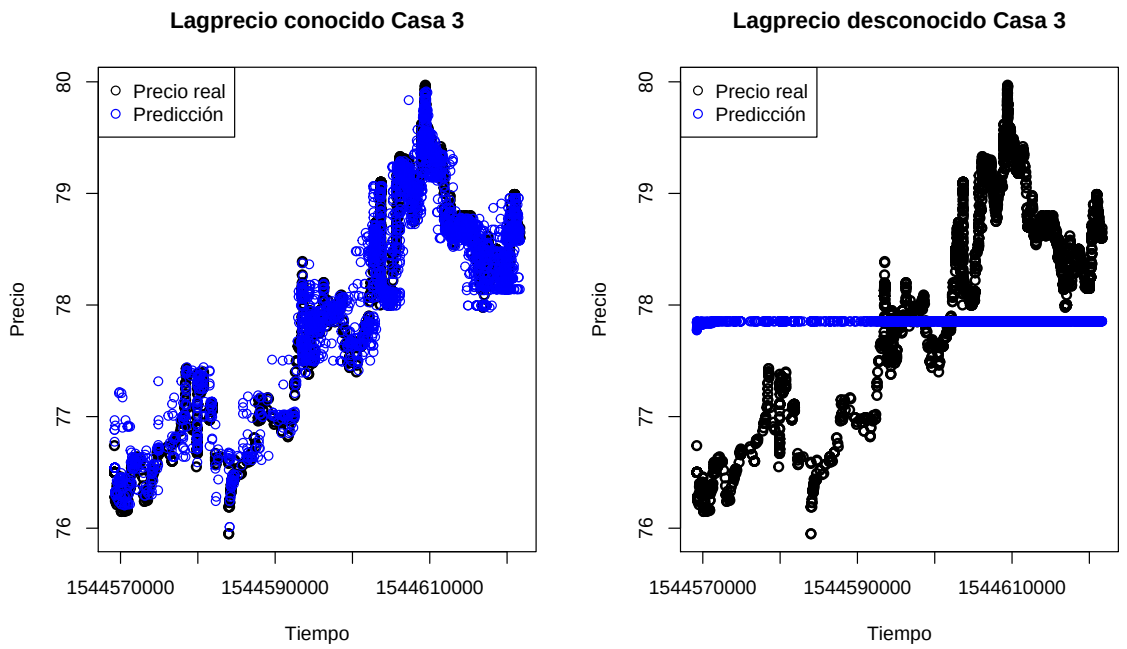

Figura 4.18: Precio $\sim$ *Lagprecio. Splines naturales. Casa* 3

```
     xlab="Tiempo", ylab="Precio")
lines(datos_test$tiempo, predic1, col ="green", lwd=2)
lines(datos_test$tiempo, predic2, col ="red", lwd=2)
legend("topleft", legend = c("Lagprecio conocido",
                             "Lagprecio desconocido"),
       col=c("green","red"), lty=1)
```

Observamos que, supuesto Lagprecio conocido, el precio sufre una leve bajada al comienzo del día, pero va subiendo hasta superar el nivel del precios del día anterior y refleja las constantes subidas y bajadas del precio. Por otro lado, la predicción supuesto Lapgrecio desconocido no refleja las numerosas variaciones sufridas por el precio y, tras sufrir una subida de 1 dólar, se mantiene constante hasta finalizar el día.

4.3.3. Precio $\sim$ Lagprecio + Casa + Cantidad + Tiempo

A continuación, vamos a estudiar la influencia de las casas de cambio a lo largo del tiempo aplicando sobre las variables Lagprecio, Cantidad y Tiempo algunas de las técnicas basadas en splines descritas en el segundo capítulo, pero no aplicaremos ninguna de estas técnicas sobre la variable Casa al ser una variable categórica.

Mediante la función **gam** se ajusta un Modelo Aditivo Generalizado y a partir de la función **s** se aplica la técnica de suavizado. Con el argumento **bs** se indica la base de funciones a usar para cada variable. En este trabajo trabajaremos con las siguientes bases:

1. Con **tp** se usan Splines de regresión Thin Plate.

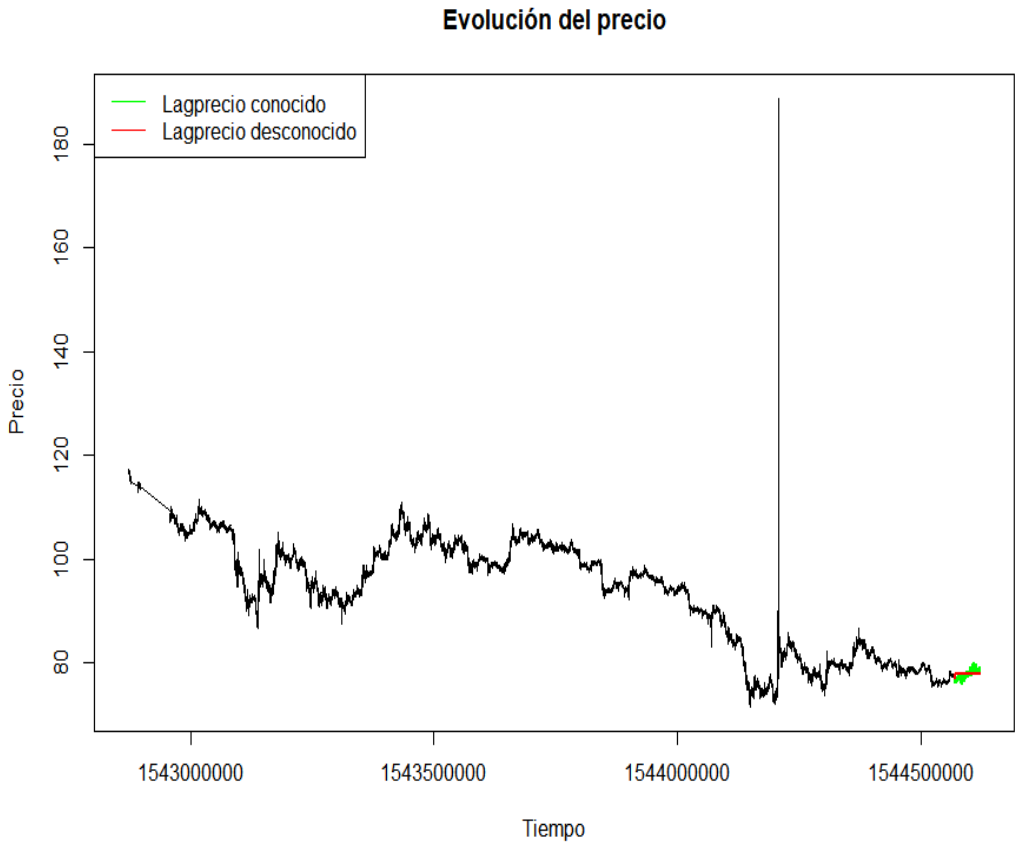

Figura 4.19: *Precio ∼ Lagprecio. Splines naturales. Evolución del precio*

2. Con **ps** se usan P-splines.

3. Con **cr** se usan Splines cúbicos de regresión.

En este modelo aplicamos Splines cúbicos de regresión sobre Tiempo y Lagprecio y P-splines sobre Cantidad.

```
fit=gam(precio ~ s(tiempo,bs="cr") +
                 s(cantidad,bs="ps") +
                 s(lagprecio,bs="cr") +
                 casa, data=datos_train)
summary(fit)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## precio ~ s(tiempo, bs = "cr") + s(cantidad, bs = "ps") + s(lagprecio,
##     bs = "cr") + casa
##
```

```
## Parametric coefficients:
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept) 92.22865    0.01041 8859.793  < 2e-16 ***
## casa2        0.03370    0.01225    2.751  0.00594 **
## casa3       -0.00788    0.01099   -0.717  0.47330
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                edf Ref.df         F p-value
## s(tiempo)    8.981  9.000 1.034e+02 < 2e-16 ***
## s(cantidad)  4.235  4.706 3.952e+00 0.00193 **
## s(lagprecio) 8.952  8.999 2.483e+05 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.987   Deviance explained = 98.7%
## GCV = 1.6826  Scale est. = 1.6824    n = 195547
```

Observamos que las variables Lagprecio, Tiempo y Cantidad son significativas.

Al ser la variable Casa una variable cualitativa con 3 categorías, se crean dos variables dummy tomando como referencia la Casa 1. Por un lado, la variable dummy correspondiente a la Casa 2 con respecto a la Casa 1 es significativa y la estimación del coeficiente asociado es mayor que cero, lo que conlleva a pensar que la Casa 2 cuenta con precios más altos que la Casa 1. No obstante, la variable dummy correspondiente a la Casa 3 con respecto a la Casa 1 no es significativa.

Predicción supuesto Lagprecio conocido

A continuación, obtenemos las predicciones de los precios para el día 12/12/2018 suponiendo conocida la variable Lagprecio del conjunto test.

```
predic1 = predict(fit,
                  newdata=list(tiempo=datos_test$tiempo,
                               cantidad=datos_test$cantidad,
                               lagprecio=datos_test$lagprecio,
                               casa=datos_test$casa))
bondad(predic1, precios_reales)

## Bondad de ajuste: 278.8949
```

Realizamos la representación gráfica (4.20) de los precios reales comparados con los precios predichos anteriormente

```
plot(precios_reales$precio, predic1, ylab="Predicciones",
     xlab="Precios reales",
     main="Comparación. Precios reales y Predicción")
```

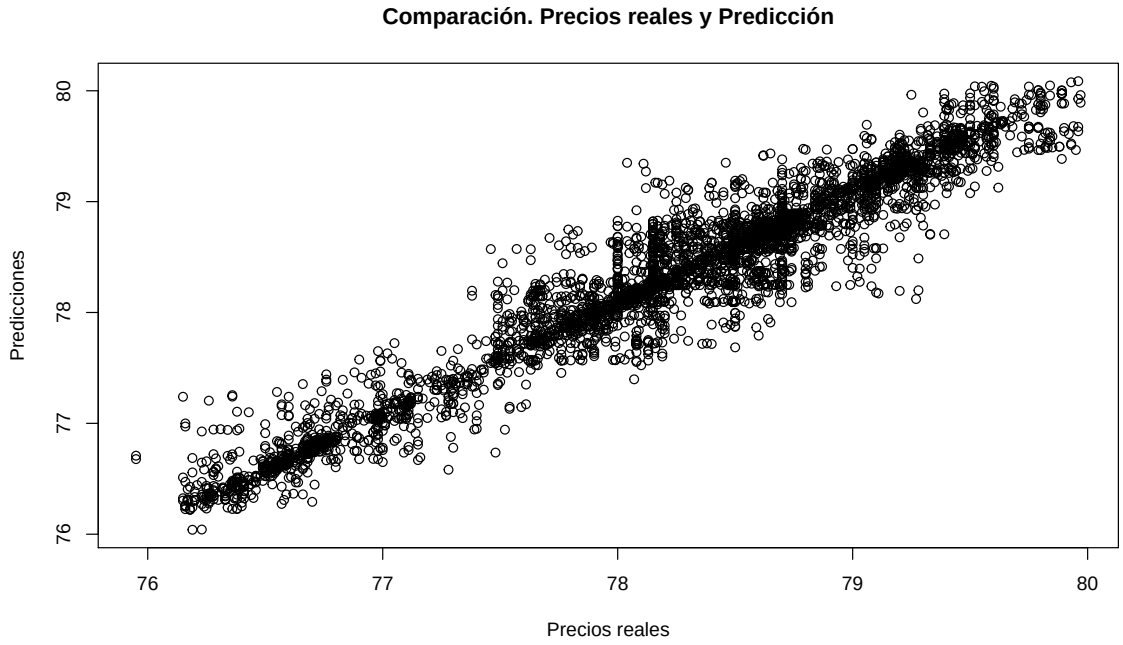

Figura 4.20: Precio $\sim$ *Lagprecio* + *Casa* + *Cantidad* + *Tiempo. Suavizado. Lagprecio conocido*

Por lo general, observamos que la nube de puntos se encuentra concentrada en la diagonal del gráfico lo que hace pensar que las predicciones realizadas son adecuadas y apreciamos mayor dispersión que para los dos primeros modelos.

Predicción supuesto Lagprecio desconocido

A continuación, obtenemos las predicciones de los precios para el día 12/12/2018 suponiendo desconocida la variable Lagprecio del conjunto test. El cálculo se realiza de igual manera que para los dos modelos anteriores.

```
lagprecio_test = rep(NA,length(datos_test$precio)+1)
predic2 = rep(NA,length(datos_test$precio))

lagprecio_test[1] = datos_train$precio[length(datos_train$precio)]
lagprecio_test[2] = predict(fit,
                            newdata=list(lagprecio=lagprecio_test[1],
                                         tiempo=datos_test$tiempo[1],
                                         cantidad=datos_test$cantidad[1],
                                         casa=datos_test$casa[1]))
predic2[1] = lagprecio_test[2]

for (i in 2:nrow(datos_test)){
  predic2[i] = predict(fit,
                       newdata=list(lagprecio=lagprecio_test[i],
                                    tiempo=datos_test$tiempo[i],
```

```
                                    cantidad=datos_test$cantidad[i],
                                    casa=datos_test$casa[i]))
  lagprecio_test[i+1]= predic2[i]
}
bondad(predic2, precios_reales)

## Bondad de ajuste: 16210842
```

Obtenemos un grado de bondad de ajuste mucho mayor que el obtenido para las predicciones supuesto Lapgrecio conocido.

Realizamos la representación gráfica (4.21) de los precios reales comparados con los precios predichos anteriormente.

```
plot(precios_reales$precio, predic2, ylab="Predicciones",
     xlab="Precios reales",
     main="Comparación. Precios reales y Predicción")
```

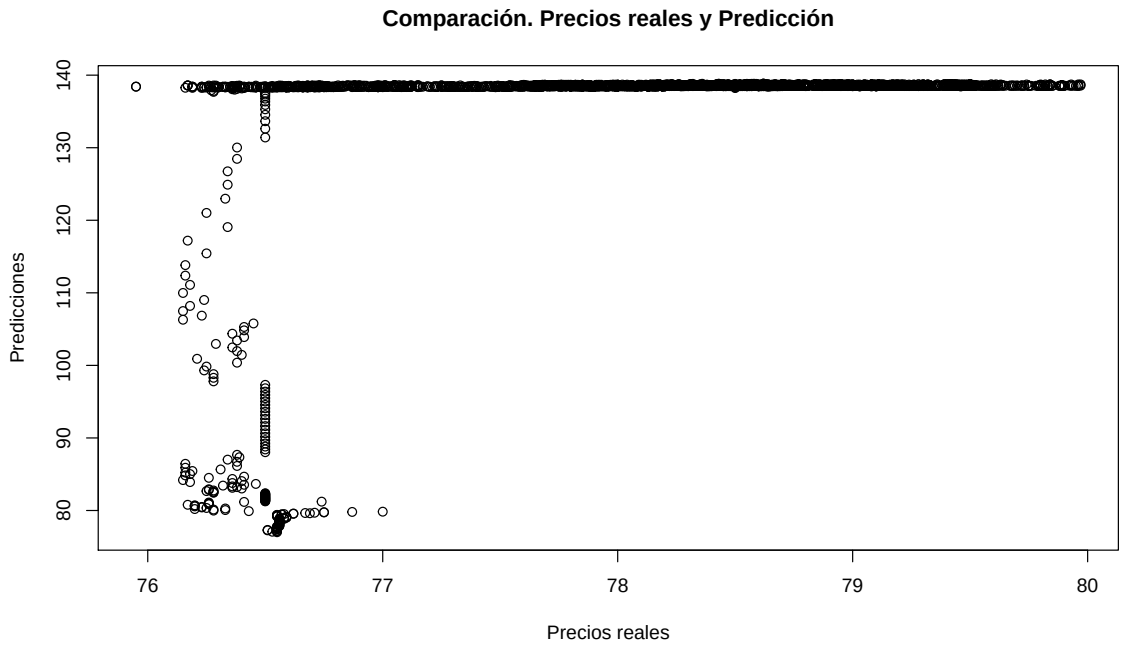

Figura 4.21: Precio $\sim$ *Lagprecio* + *Casa* + *Cantidad* + *Tiempo. Suavizado. Lagprecio desconocido*

En este gráfico apreciamos el deterioro de las predicciones con respecto a las obtenidas supuesto Lagprecio conocido. Para la inmensa mayoría de precios reales, se obtienen predicciones muy por encima de su valor real. El recorrido de las predicciones abarca desde por debajo de los 80 hasta los 138 dólares.

Una vez obtenidas ambas predicciones, realizamos la representación gráfica (4.22) de las dos a lo largo del día 12/12/2018.

```
par(mfrow =c(1,2))

a = data.frame(precio = precios_reales, predic1,
               casa=datos_test$casa, time=datos_test$tiempo)
plot(a$time, a$precio, ylim=c(min(a$predic1,  a$precio),
                              max(a$predic1, a$precio)), lwd=2,
     xlab="Tiempo", ylab="Precio",
     main="Lagprecio conocido")
points(a$time, a$predic1, col =" blue", lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"), pch=c(1,1))

b = data.frame(precio = precios_reales, predic2,
               casa=datos_test$casa, time=datos_test$tiempo)
plot(b$time, b$precio, ylim=c(min(b$predic2,  b$precio),
                              max(b$predic2, b$precio)), lwd=2,
     xlab="Tiempo", ylab="Precio",
     main="Lagprecio desconocido")
points(b$time, b$predic2, col =" blue", lwd=1)
legend("center", legend = c("Precio real","Predicción"),
       col=c("black","blue"), pch=c(1,1))
```

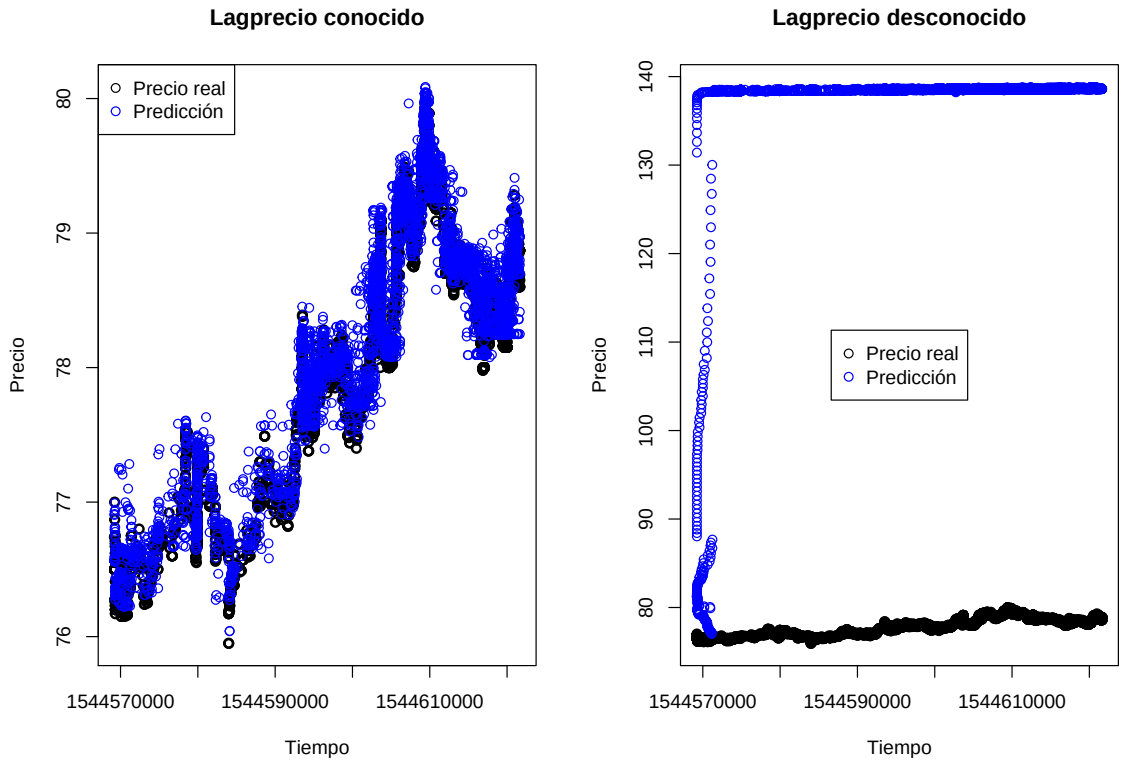

Figura 4.22: Precio $\sim$ *Lagprecio* + *Casa* + *Cantidad* + *Tiempo*. *Suavizado*. *Predicciones*

```
par(mfrow =c(1,1))
```

Al comienzo del día, las predicciones, supuesto Lagprecio desconocido, sufren una subida considerable hasta rondar los 140 dólares. Los precios permanecen en este nivel durante la mayor parte del día y al final del mismo. Como podemos observar, no refleja las subidas y bajadas de los precios reales.

Tal y como podemos observar en los gráficos 4.23, 4.24 y 4.25, si representamos los resultados obtenidos mediante ambas predicciones, diferenciando entre las tres casas de cambio, se manifiesta con mayor claridad la pérdida de calidad de las predicciones realizadas suponiendo desconocida la variable Lagprecio del conjunto test.

```
par(mfrow =c(1,2))

  d <- a %>% filter(casa==1)
  plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
                                max(d$predic1, d$precio)),
       pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
  title(paste0("Lagprecio conocido Casa 1"))
  points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
  legend("topleft", legend = c("Precio real","Predicción"),
         col=c("black","blue"),  pch=c(1,1))

  c <- b %>% filter(casa==1)
  plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                                max(c$predic2, c$precio)),
       lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
  title(paste0("Lagprecio desconocido Casa 1"))
  points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
  legend("center", legend = c("Precio real","Predicción"),
         col=c("black","blue"),  pch=c(1,1))
```

```
par(mfrow =c(1,1))
```

```
par(mfrow =c(1,2))
d <- a %>% filter(casa==2)
plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
                              max(d$predic1, d$precio)),
     pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio conocido Casa 2"))
points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))

c <- b %>% filter(casa==2)
```

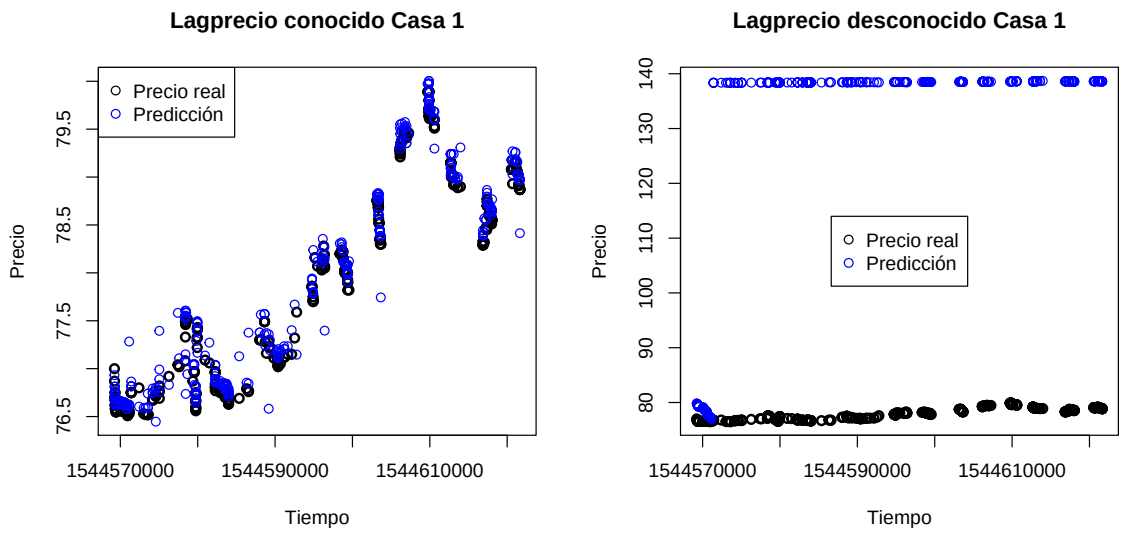

Figura 4.23: Precio $\sim$ *Lagprecio* + *Casa* + *Cantidad* + *Tiempo*. *Suavizado*. *Casa* 1

```
plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                              max(c$predic2, c$precio)),
     lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio desconocido Casa 2"))
points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
legend("center", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))
```

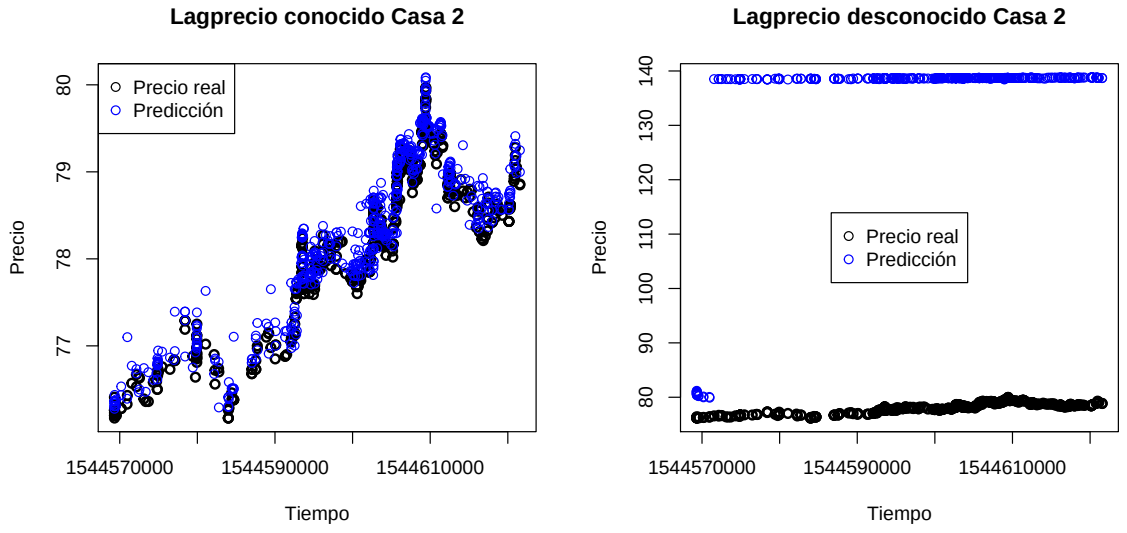

Figura 4.24: Precio $\sim$ *Lagprecio* + *Casa* + *Cantidad* + *Tiempo*. *Suavizado*. *Casa* 2

```
par(mfrow =c(1,1))
```

```
par(mfrow =c(1,2))

d <- a %>% filter(casa==3)
plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
                              max(d$predic1, d$precio)),
     pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio conocido Casa 3"))
points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))

c <- b %>% filter(casa==3)
plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                              max(c$predic2, c$precio)),
     lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio desconocido Casa 3"))
points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
legend("center", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))
```

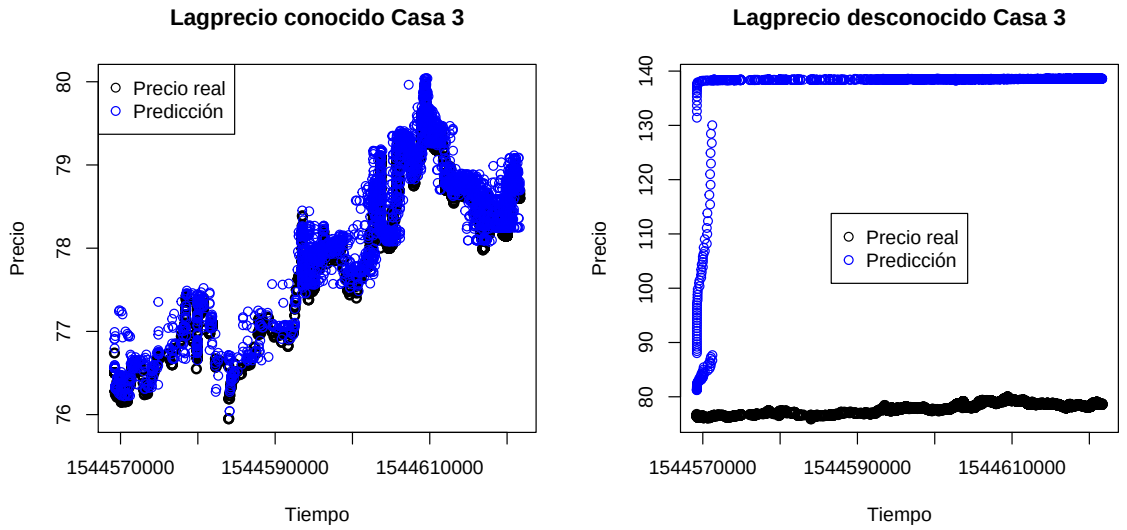

Figura 4.25: Precio $\sim$ *Lagprecio* + *Casa* + *Cantidad* + *Tiempo*. *Suavizado*. *Casa* 3

```
par(mfrow =c(1,1))
```

A continuación, realizamos un gráfico evolutivo (4.26) del precio de la criptomoneda incluyendo ambas predicciones.

```
plot(datos_train$tiempo, datos_train$precio,
     xlim=c(min(datos_train$tiempo), max(datos_test$tiempo)),
     type="l", main="Evolución del precio",
     xlab="Tiempo", ylab="Precio")
lines(datos_test$tiempo, predic1, col ="green", lwd=2)
lines(datos_test$tiempo, predic2, col ="red", lwd=2)
legend("topleft", legend = c("Lagprecio conocido",
                             "Lagprecio desconocido"),
       col=c("green","red"), lty=1)
```

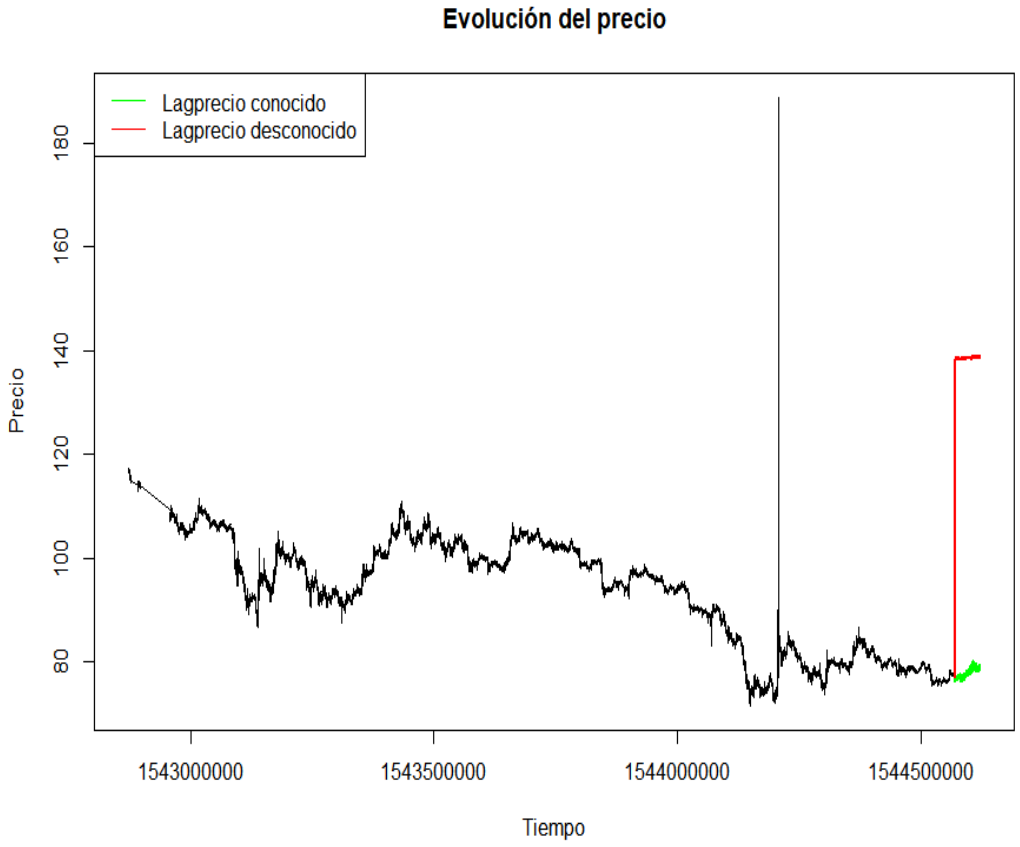

Figura 4.26: *Precio ∼ Lagprecio + Casa + Cantidad + Tiempo. Suavizado. Evolución del precio*

Observamos que, supuesto Lagprecio conocido, el precio sufre una leve bajada al comienzo del día, pero va subiendo hasta superar el nivel del precios del día anterior y refleja las constantes subidas y bajadas del precio. Por otro lado, los precios, supuesto Lapgrecio desconocido, no reflejan las numerosas variaciones sufridas por el precio, sufren una subida considerable hasta rozar los 140 dólares y se mantienen prácticamente constantes hasta el final del día.

4.3.4. Precio ∼ Lagprecio + Cantidad ∗ Tiempo ∗ Casa

A continuación, vamos a construir un modelo que estudia la posible interacción entre las variables Cantidad, Tiempo y Casa aplicando tensores. En el modelo también se incluye la variable Lagprecio.

Mediante la función **gam** se ajusta un Modelo Aditivo Generalizado y a partir de la función **te** se aplica la técnica de Tensores y de suavizado a la vez. Con **bs** se indica la base de funciones a usar para cada variable. En este modelo aplicamos Splines cúbicos de regresión sobre Tiempo y splines de regresión Thin Plate sobre Cantidad. No aplicamos ninguna técnica sobre la variable Lagprecio.

```
fit=gam(precio ~ lagprecio +
          te(tiempo, cantidad, bs=c("cr", "tp"), by=casa),
        data=datos_train)
summary(fit)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## precio ~ lagprecio + te(tiempo, cantidad, bs = c("cr", "tp"),
##     by = casa)
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.1820800  0.0544491   58.44   <2e-16 ***
## lagprecio   0.9654758  0.0005894 1637.99   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                                edf Ref.df     F p-value
## te(tiempo,cantidad):casa1  7.764  8.352  89.5  <2e-16 ***
## te(tiempo,cantidad):casa2 13.988 14.616 114.9  <2e-16 ***
## te(tiempo,cantidad):casa3  9.025  9.070 278.9  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.986   Deviance explained = 98.6%
## GCV = 1.8158  Scale est. = 1.8155    n = 195547
```

Obtenemos que todas las variables son significativas.

Predicción supuesto Lagprecio conocido

A continuación, obtenemos las predicciones de los precios para el día 12/12/2018 suponiendo conocida la variable Lagprecio del conjunto test.

```
predic1 = predict(fit,
                  newdata=list(tiempo=datos_test$tiempo,
                               cantidad=datos_test$cantidad,
                               lagprecio=datos_test$lagprecio,
                               casa=datos_test$casa))
bondad(predic1, precios_reales)

## Bondad de ajuste: 285.0183
```

Realizamos la representación gráfica (4.27) de los precios reales comparados con los precios predichos anteriormente

```
plot(precios_reales$precio, predic1, ylab="Predicciones",
     xlab="Precios reales",
     main="Comparación. Precios reales y Predicción")
```

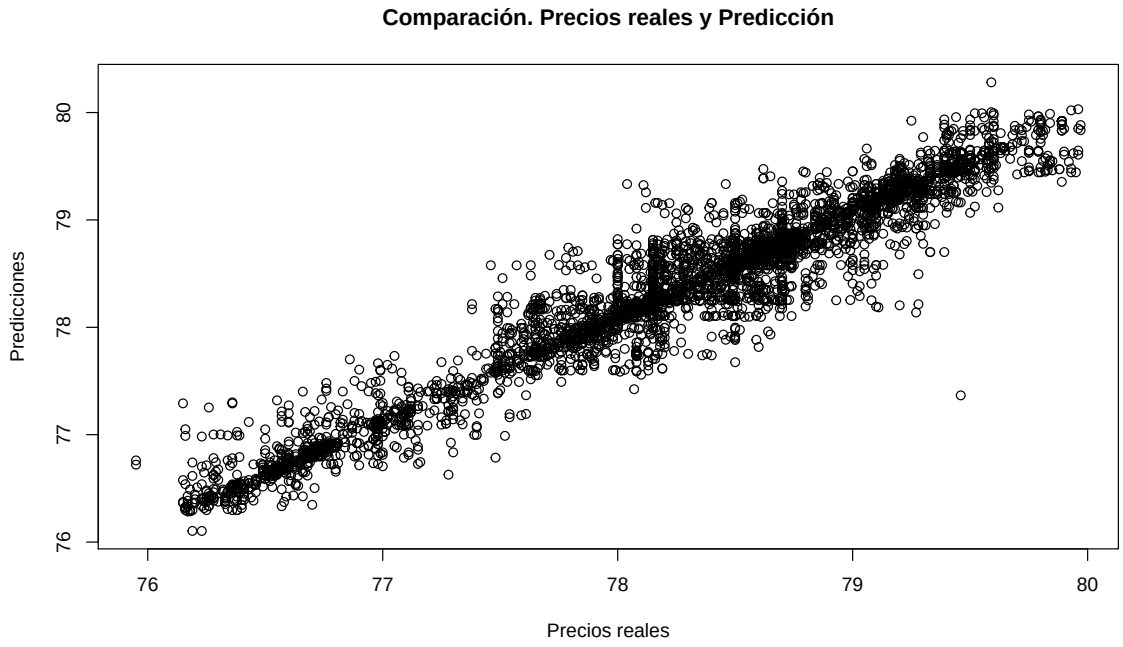

Figura 4.27: Precio $\sim$ *Lagprecio* + *Cantidad* $*$ *Tiempo* $*$ *Casa*. *Tensores*. *Lagprecio conocido*

Por lo general, observamos que la nube de puntos se encuentra concentrada en la diagonal del gráfico lo que hace pensar que las predicciones realizadas son adecuadas y apreciamos mayor dispersión que para los dos primeros modelos.

Predicción supuesto Lagprecio desconocido

A continuación, obtenemos las predicciones de los precios para el día 12/12/2018 suponiendo desconocida la variable Lagprecio del conjunto test. El cálculo se realiza de igual manera que para los modelos anteriores.

```
lagprecio_test = rep(NA,length(datos_test$precio)+1)
predic2 = rep(NA,length(datos_test$precio))

lagprecio_test[1] = datos_train$precio[length(datos_train$precio)]
lagprecio_test[2] = predict(fit,
                            newdata=list(lagprecio=lagprecio_test[1],
                                         cantidad=datos_test$cantidad[1],
                                         tiempo=datos_test$tiempo[1],
                                         casa=datos_test$casa[1]))
predic2[1] = lagprecio_test[2]

for (i in 2:nrow(datos_test)){
  predic2[i] = predict(fit,
                       newdata=list(lagprecio=lagprecio_test[i],
                                    cantidad=datos_test$cantidad[i],
                                    tiempo=datos_test$tiempo[i],
                                    casa=datos_test$casa[i]))
  lagprecio_test[i+1]= predic2[i]
}
bondad(predic2, precios_reales)

## Bondad de ajuste: 33989.58
```

Obtenemos un grado de bondad de ajuste menor al obtenido para el modelo anterior.

Realizamos la representación gráfica (4.28) de los precios reales comparados con los precios predichos anteriormente.

```
plot(precios_reales$precio, predic2, ylab="Predicciones",
     xlab="Precios reales",
     main="Comparación. Precios reales y Predicción")
```

En este gráfico apreciamos un desfase en los precios de 2 a 4 dólares. Para la inmensa mayoría de criptomonedas se obtienen predicciones de su precio por encima de su valor real, pero concentradas entre 80 y 82 dólares.

Realizamos la representación gráfica (4.29) de los precios reales comparados con los precios predichos anteriormente.

```
par(mfrow =c(1,2))

a = data.frame(precio = precios_reales, predic1,
               casa=datos_test$casa, time=datos_test$tiempo)
plot(a$time, a$precio, ylim=c(min(a$predic1,  a$precio),
                              max(a$predic1, a$precio)), lwd=2,
     xlab="Tiempo", ylab="Precio",
```

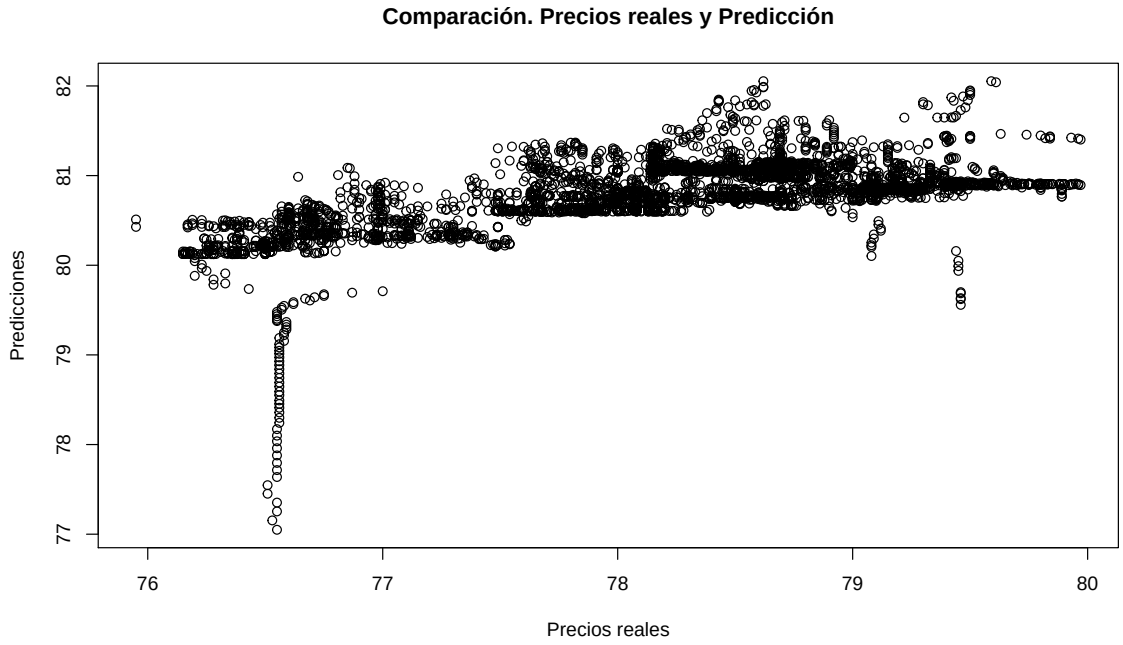

Figura 4.28: Precio $\sim$ *Lagprecio* + *Cantidad* $*$ *Tiempo* $*$ *Casa*. *Tensores*. *Lagprecio desconocido*

```
     main="Lagprecio conocido")
points(a$time, a$predic1, col =" blue", lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"), pch=c(1,1))

b = data.frame(precio = precios_reales, predic2,
               casa=datos_test$casa, time=datos_test$tiempo)
plot(b$time, b$precio, ylim=c(min(b$predic2,  b$precio),
                              max(b$predic2, b$precio)), lwd=2,
     xlab="Tiempo", ylab="Precio",
     main="Lagprecio desconocido")
points(b$time, b$predic2, col =" blue", lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"), pch=c(1,1))
```

```
par(mfrow =c(1,1))
```

Al comienzo del día, las predicciones, supuesto Lagprecio desconocido, sufren una subida de alrededor de 3 dólares hasta superar los 80 dólares. No obstante, a lo largo del día estas predicciones reflejan las variaciones sufridas por el precio real y se aprecia el desfase de 2 a 4 dólares.

Tal y como podemos observar en los gráficos 4.30, 4.31 y 4.32, si representamos los resultados obtenidos mediante ambas predicciones, diferenciando entre las tres casas

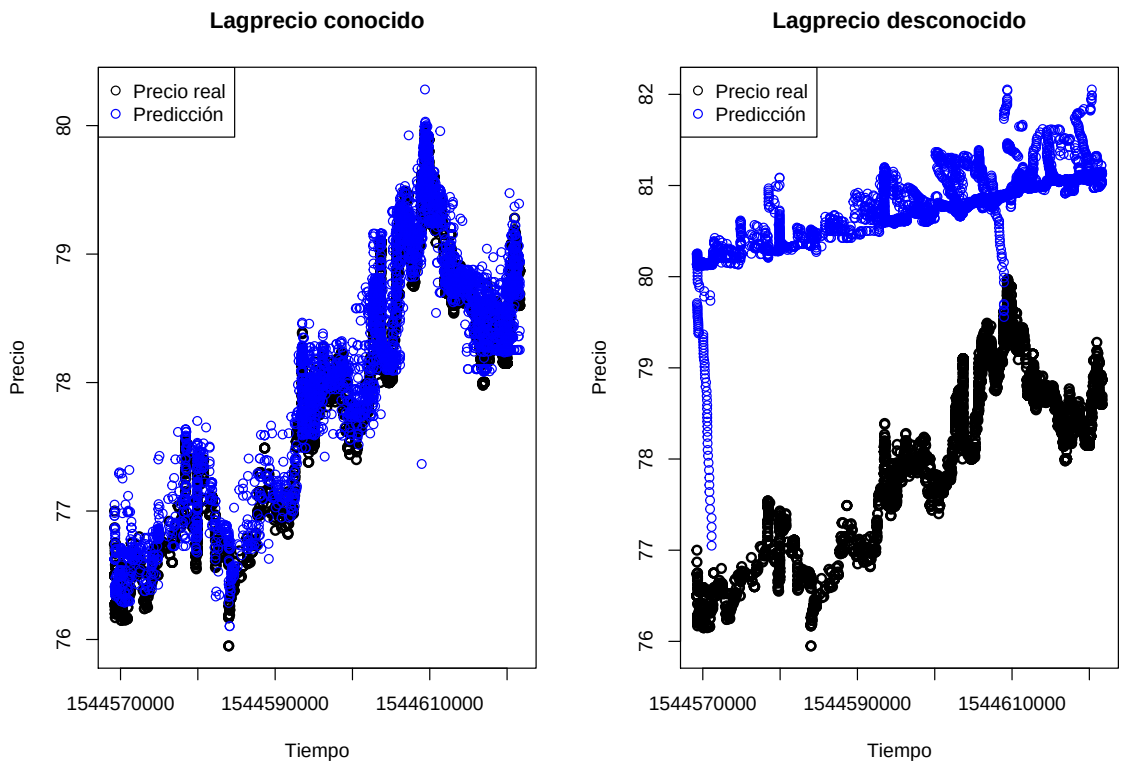

Figura 4.29: Precio $\sim$ *Lagprecio* + *Cantidad* $*$ *Tiempo* $*$ *Casa*. *Tensores*. *Predicciones*

de cambio, se manifiesta con mayor claridad la pérdida de calidad de las predicciones realizadas suponiendo desconocida la variable Lagprecio del conjunto test.

```
par(mfrow =c(1,2))

  d <- a %>% filter(casa==1)
  plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
                                max(d$predic1, d$precio)),
       pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
  title(paste0("Lagprecio conocido Casa 1"))
  points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
  legend("topleft", legend = c("Precio real","Predicción"),
         col=c("black","blue"),  pch=c(1,1))

  c <- b %>% filter(casa==1)
  plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                                max(c$predic2, c$precio)),
       lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
  title(paste0("Lagprecio desconocido Casa 1"))
  points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
  legend("topleft", legend = c("Precio real","Predicción"),
         col=c("black","blue"),  pch=c(1,1))
```

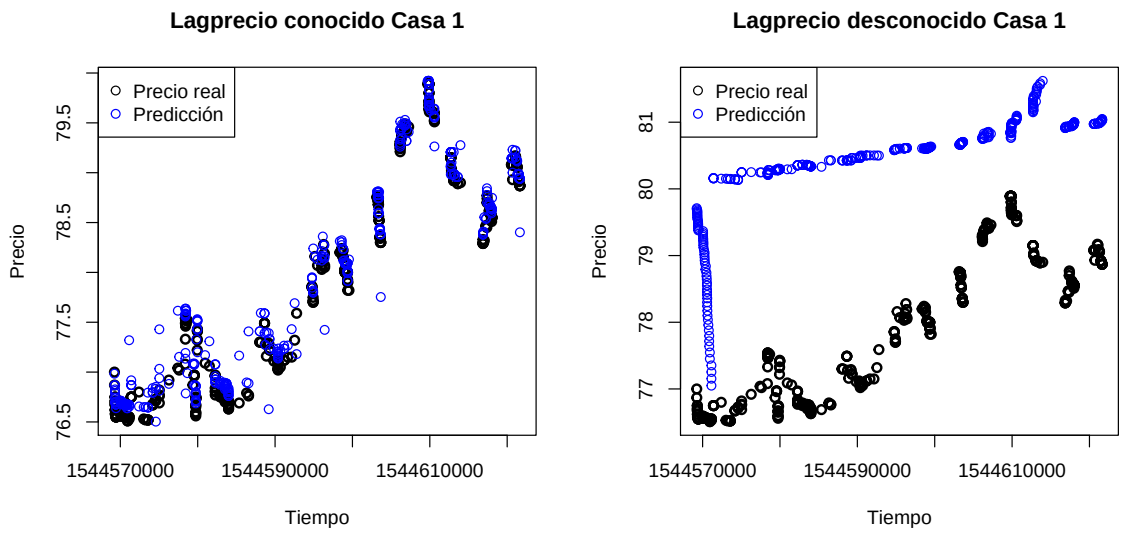

Figura 4.30: Precio $\sim Lagprecio + Cantidad * Tiempo * Casa.Tensores.Casa\,1$

```
par(mfrow =c(1,1))
```

```
par(mfrow =c(1,2))
d <- a %>% filter(casa==2)
plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
                              max(d$predic1, d$precio)),
     pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio conocido Casa 2"))
points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))

c <- b %>% filter(casa==2)
plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                              max(c$predic2, c$precio)),
     lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio desconocido Casa 2"))
points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))
```

```
par(mfrow =c(1,1))
```

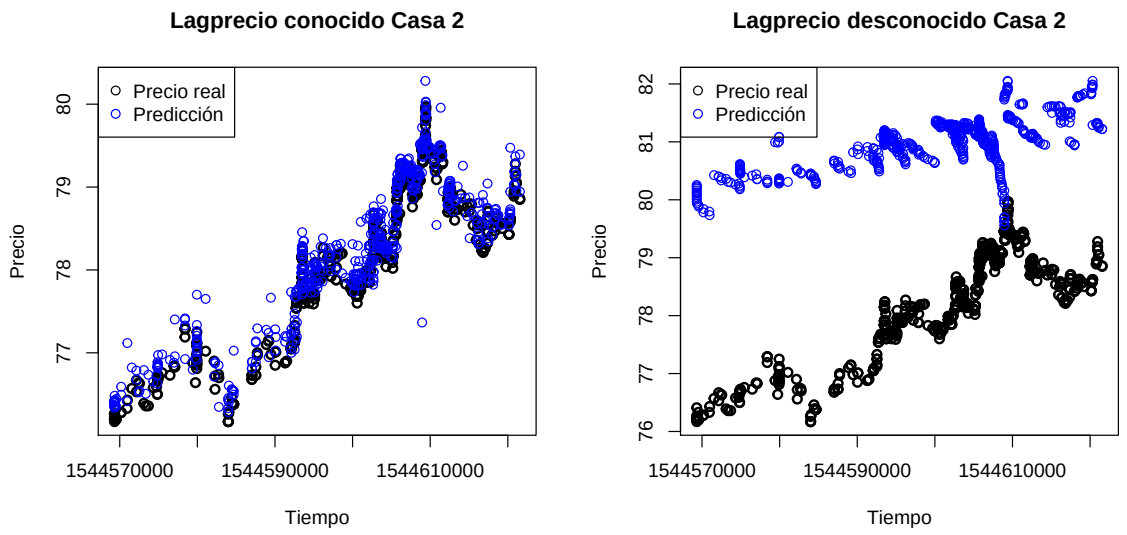

Figura 4.31: Precio $\sim Lagprecio + Cantidad * Tiempo * Casa.Tensores.Casa\,2$

```
par(mfrow =c(1,2))

d <- a %>% filter(casa==3)
plot(d$time, d$precio, ylim=c(min(d$predic1, d$precio),
                              max(d$predic1, d$precio)),
     pch=1, lwd=2, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio conocido Casa 3"))
points(d$time,d$predic1, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))

c <- b %>% filter(casa==3)
plot(c$time, c$precio, ylim=c(min(c$predic2, c$precio),
                              max(c$predic2, c$precio)),
     lwd=2, pch=1, xlab = "Tiempo", ylab="Precio")
title(paste0("Lagprecio desconocido Casa 3"))
points(c$time,c$predic2, col =" blue", pch=1, lwd=1)
legend("topleft", legend = c("Precio real","Predicción"),
       col=c("black","blue"),  pch=c(1,1))
```

```
par(mfrow =c(1,1))
```

Observamos como, diferenciando entre las casas de cambio, el ajuste realizado refleja las constantes variaciones sufridas por el precio de la criptomoneda a lo largo del día.

A continuación, realizamos un gráfico evolutivo (4.33) del precio de la criptomoenda incluyendo ambas predicciones.

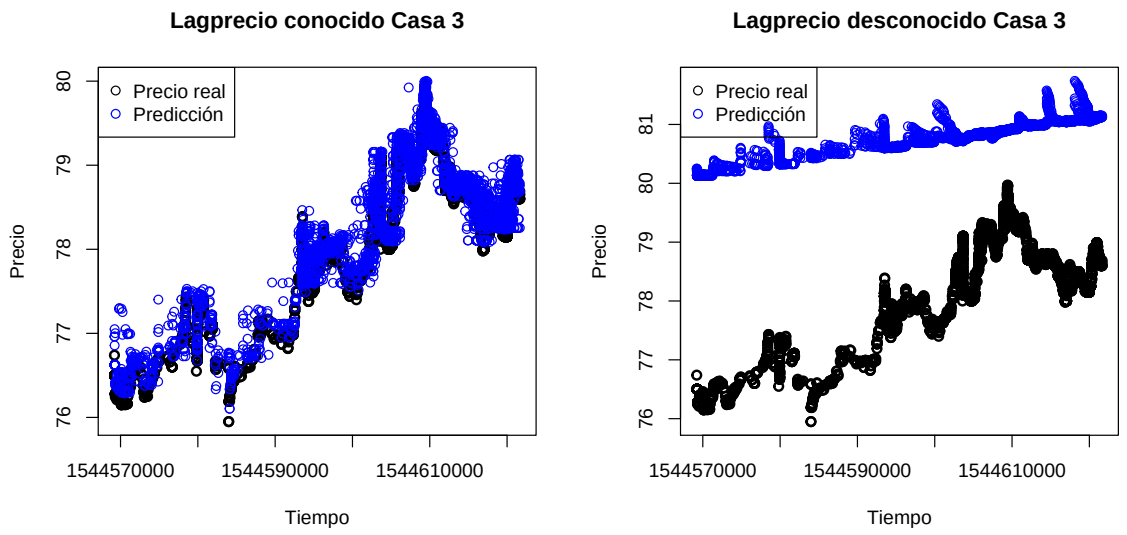

Figura 4.32: Precio $\sim$ *Lagprecio* + *Cantidad* $*$ *Tiempo* $*$ *Casa.Tensores.Casa* 3

```
plot(datos_train$tiempo, datos_train$precio,
     xlim=c(min(datos_train$tiempo), max(datos_test$tiempo)),
     type="l", main="Evolución del precio",
     xlab="Tiempo", ylab="Precio")
lines(datos_test$tiempo, predic1, col ="green", lwd=2)
lines(datos_test$tiempo, predic2, col ="red", lwd=2)
legend("topleft", legend = c("Lagprecio conocido",
                             "Lagprecio desconocido"),
       col=c("green","red"), lty=1)
```

Observamos que, supuesto Lagprecio conocido, el precio sufre una leve bajada al comienzo del día, pero va subiendo hasta superar el nivel del precios del día anterior y refleja las constantes subidas y bajadas del precio. Por otro lado, la predicción supuesto Lapgrecio desconocido sufre una temprana subida de alrededor de 3 dólares, pero refleja las constantes variaciones sufridas por el precio de la criptomoneda a lo largo del día.

Simulación

Tras la construcción de una gran variedad de modelos, tanto en este documento como en el anexo, llega el momento de escoger aquel a partir del cual elegiremos la casa de cambio para realizar cualquier transacción de una determinada cantidad de criptomonedas. Para ello, consideremos los cuatro modelos expuestos en este documento y las predicciones realizadas supuesta la variable Lagprecio desconocida.

Para comenzar, podemos afirmar que no haremos uso del tercer modelo (4.3.3), *Precio* $\sim$ *Lagprecio* + *Cantidad* + *Tiempo* + *Casa*, pues cuenta con un grado de bondad de ajuste del orden de 16 millones y a partir de los gráficos 4.22, 4.23, 4.24 y 4.25 observamos que las predicciones realizadas no se asemejan a la evolución del

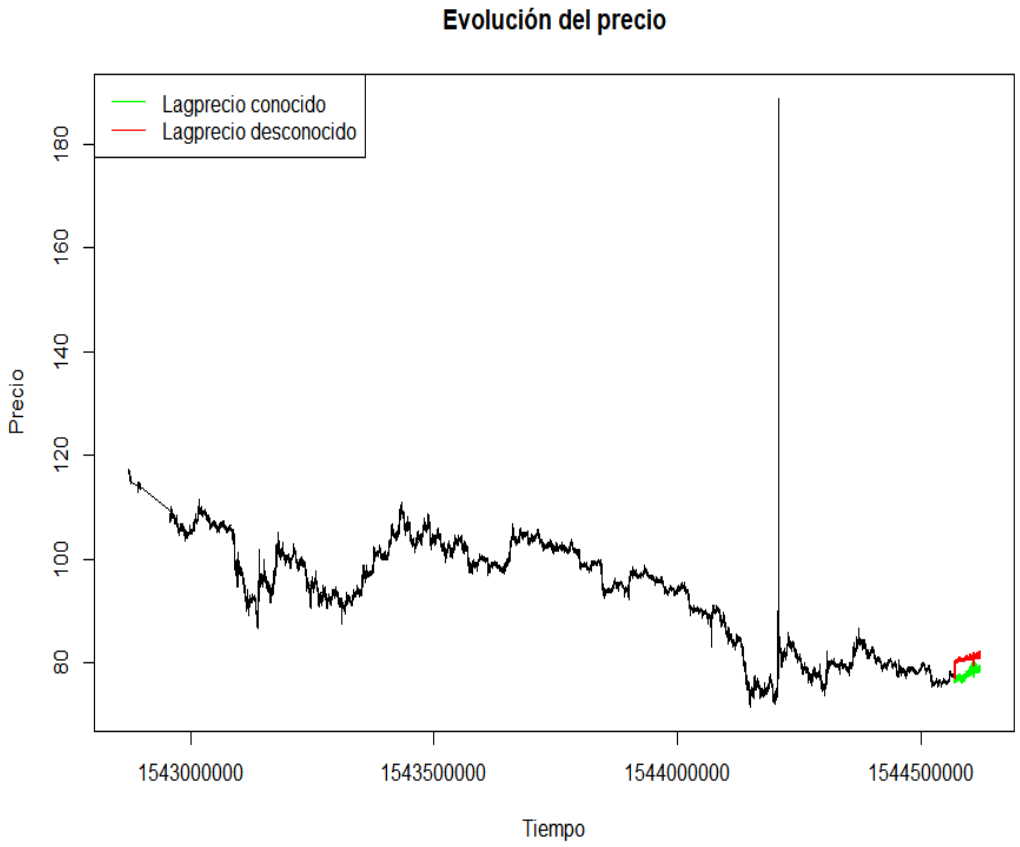

Figura 4.33: *Precio* $\sim$ *Lagprecio + Cantidad * Tiempo * Casa. Tensores. Evolución del precio*

precio real.

A pesar de que la bondad de ajuste de los dos primeros modelos (4.3.1, 4.3.2) sean 4692 y 4156, respectivamente, es necesario hacer hincapié en los gráficos 4.7, 4.8, 4.9 y 4.10, para el modelo *Precio* $\sim$ *Lagprecio + Tiempo*, y los gráficos 4.15, 4.16, 4.17 y 4.18, para el modelo *Precio* $\sim$ *Lagprecio*. En éstos podemos observar que las predicciones realizadas no se asemejan a la evolución del precio real. Ambas predicciones comienzan el día con un precio superior al real y, en el caso del primer modelo, los precios van disminuyendo durante el resto del día y, en el caso del segundo modelo, se mantienen constantes durante el resto del día.

Por último, el cuarto modelo (4.3.4), Precio $\sim$ *Lagprecio + Cantidad* $*$ *Tiempo* $*$ *Casa*, cuenta con un grado de bondad de ajuste del orden de 33 mil, pero como podemos observar en los gráficos 4.29, 4.30, 4.31 y 4.32, las predicciones reflejan las constantes variaciones sufridas por el precio de la criptomoneda a lo largo del día. Este será el modelo a partir del cual realizaremos la simulación para mostrar como podríamos aplicarlo en la práctica.

La simulación consiste en una comparación de los precios predichos, supuesto Lag-

precio desconocido, para las tres casas de cambio y para las mismas cantidades con el objetivo de dirimir a partir de qué casa de cambio conviene más realizar una transacción para una determinada cantidad de criptomonedas.

Para tomar esta decisión veremos cómo se comportan los precios para cada casa de cambio durante los 10 minutos posteriores a la última transacción recogida en la base de datos.

Primero, obtenemos el modelo haciendo uso de todo el conjunto de datos original.

```
fit=gam(precio ~ lagprecio +
          te(tiempo, cantidad, bs=c("cr", "tp"), by=casa),
        data=datos2[-1,])
summary(fit)

##
## Family: gaussian
## Link function: identity
##
## Formula:
## precio ~ lagprecio + te(tiempo, cantidad, bs = c("cr", "tp"),
##     by = casa)
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 3.0509756  0.0526259   57.98   <2e-16 ***
## lagprecio   0.9667791  0.0005717 1691.09   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                                edf Ref.df     F p-value
## te(tiempo,cantidad):casa1  7.730  8.327  89.5  <2e-16 ***
## te(tiempo,cantidad):casa2 14.225 14.786 111.5  <2e-16 ***
## te(tiempo,cantidad):casa3  9.038  9.121 272.8  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.986   Deviance explained = 98.6%
## GCV = 1.7761  Scale est. = 1.7758    n = 200187
```

Calcularemos, para cada casa y las mismas cantidades, las predicciones a lo largo de 10 minutos. Obtendremos 60 predicciones separadas cada 10 segundos.

```
tiempo_pred = rep(NA, 60)
tiempo_pred[1] = datos2$tiempo[length(datos2$tiempo)] + 1
for (i in 2:60){
```

```
  tiempo_pred[i] = tiempo_pred[i-1] + 10
}
```

```
summary(datos2$cantidad)

##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
##    0.0000    0.4782    1.8800    7.0415    5.0000 3134.2355

quantile(datos2$cantidad, .05)

##   5%
## 0.01

quantile(datos2$cantidad, .95)

##      95%
## 30.19999
```

Observamos que el rango de la variable Cantidad va de 0 a 3134,23. Decidimos generar números aleatoriamente entre el percentil 5 y el percentil 95 de la variable Cantidad para evitar valores extremos. Estos valores servirán para obtener las predicciones necesarias para seleccionar la mejor casa de cambio.

```
set.seed(77819)
cantidad_test = rep(NA, 60)
for (i in 1:60){
  cantidad_test[i] = runif(1,
                           min=quantile(datos2$cantidad, .05),
                           max=quantile(datos2$cantidad, .95))
}
head(cantidad_test)

## [1]  4.991826 19.354782 11.195411  5.715054 16.122378 22.158347
```

Una vez obtenidas, pasamos a calcular las predicciones supuesto Lagprecio desconocido para las tres casas.

```
casa1 = rep(1, 60)
lagprecio_test = rep(NA, 60+1)
predic2_casa1 = rep(NA, 60)

lagprecio_test[1] = datos2$precio[length(datos2$precio)]
lagprecio_test[2] = predict(fit,
                            newdata=list(lagprecio=lagprecio_test[1],
                                         cantidad=cantidad_test[1],
                                         tiempo=tiempo_pred[1],
```

```
                                          casa=casa1[1]))
predic2_casa1[1] = lagprecio_test[2]


for (i in 2:60){
  predic2_casa1[i] = predict(fit,
                             newdata=list(lagprecio=lagprecio_test[i],
                                          cantidad=cantidad_test[i],
                                          tiempo=tiempo_pred[i],
                                          casa=casa1[i]))
  lagprecio_test[i+1]= predic2_casa1[i]
}
```

```
casa2 = rep(2, 60)
lagprecio_test = rep(NA, 60+1)
predic2_casa2 = rep(NA, 60)

lagprecio_test[1] = datos2$precio[length(datos2$precio)]
lagprecio_test[2] = predict(fit,
                            newdata=list(lagprecio=lagprecio_test[1],
                                         cantidad=cantidad_test[1],
                                         tiempo=tiempo_pred[1],
                                         casa=casa2[1]))
predic2_casa2[1] = lagprecio_test[2]


for (i in 2:60){
  predic2_casa2[i] = predict(fit,
                             newdata=list(lagprecio=lagprecio_test[i],
                                          cantidad=cantidad_test[i],
                                          tiempo=tiempo_pred[i],
                                          casa=casa2[i]))
  lagprecio_test[i+1]= predic2_casa2[i]
}
```

```
casa3 = rep(3, 60)
lagprecio_test = rep(NA, 60+1)
predic2_casa3 = rep(NA, 60)

lagprecio_test[1] = datos2$precio[length(datos2$precio)]
lagprecio_test[2] = predict(fit,
                            newdata=list(lagprecio=lagprecio_test[1],
                                         cantidad=cantidad_test[1],
                                         tiempo=tiempo_pred[1],
                                         casa=casa3[1]))
```

```
predic2_casa3[1] = lagprecio_test[2]


for (i in 2:60){
  predic2_casa3[i] = predict(fit,
                             newdata=list(lagprecio=lagprecio_test[i],
                                          cantidad=cantidad_test[i],
                                          tiempo=tiempo_pred[i],
                                          casa=casa3[i]))
  lagprecio_test[i+1]= predic2_casa3[i]
}
```

Realizamos el gráfico (4.34) que muestra la evolución de las predicciones calculadas anteriormente para las tres casas.

```
b = data.frame(casa = factor(c(casa1, casa2, casa3)),
               pred = c(predic2_casa1, predic2_casa2, predic2_casa3),
               time = c(tiempo_pred, tiempo_pred, tiempo_pred)
)

b %>% ggplot(aes(aes(y=pred, x=time)))+
  geom_line(aes(y=pred, x=time, colour=casa), size=2)+
  xlab('Tiempo')+ylab('Precio')+
  ggtitle('Simulación')
```

A la hora de aconsejar al usuario una de estas tres casas de cambio a partir de la cual realizar la transacción, podemos optar por la primera ya que, a nivel de precios, al término de los 10 minutos se encuentra por debajo de las otras dos casas. Los precios de la Casa 2 resultan próximos a los de la Casa 1 y especialmente durante los primeros minutos, pero terminan siendo mayores que los de la Casa 1. Observamos también que los precios de la Casa 3 superan con creces a los de las otras dos casas y supera la barrera de los 80 dólares a los poco minutos, mientras que las otras dos casas la superan al término de los 10 minutos.

Conclusión

La modelización estadística es un proceso de simplificación de la realidad que debe valorarse en la práctica en función de su utilidad para los objetivos del decisor. La modelización depende de muchos factores tales como decidir si optar por un enfoque paramétrico o no paramétrico, estudiar qué tipo de relación existe entre la variable objetivo y las explicativas, etc. Es aquí donde entra en juego el Modelo Aditivo Generalizado ya que permite que las relaciones entre la variable objetivo y las variables explicativas no deban ser lineales mientras se mantiene la aditividad.

Algunas de las técnicas destacadas para llevar a cabo el estudio del modelo son la regresión polinómica, los splines cúbicos de regresión o los de suavizado. Si bien desde el punto de vista práctico dichas técnicas se caracterizan por tener una gran

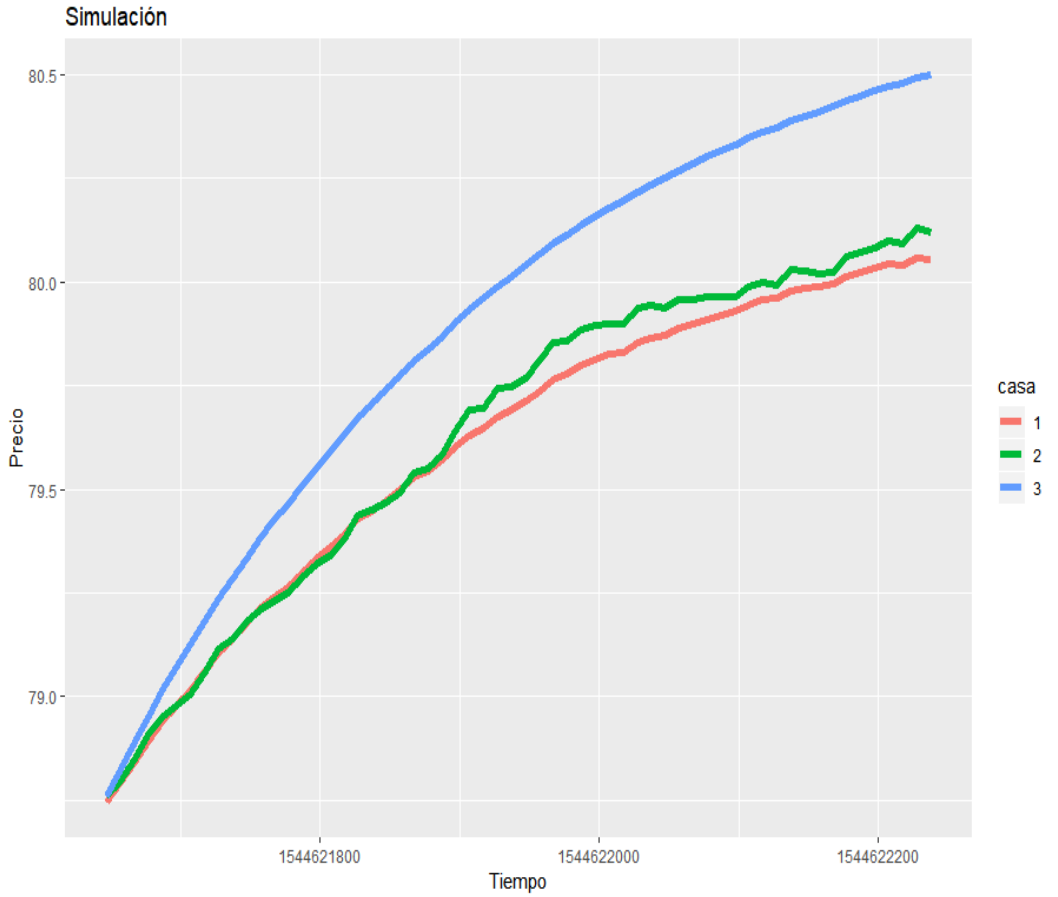

Figura 4.34: *Precio Lagprecio + Cantidad * Tiempo * Casa. Simulación.*

interpretabilidad, el hecho de elegir una base de funciones y la localización de los nodos introduce un grado de subjetividad al proceso de ajuste del modelo. En los tres primeros modelos presentados en el último capítulo hemos observado cómo la regresión polinómica, los splines cúbicos de regresión, los splines de suavizado y los P-splines no son capaces de reflejar las constantes variaciones sufridas por el precio a lo largo del día ni de proporcionar información fiable sobre qué casa de cambio elegir para realizar una transacción de una determinada cantidad de criptomonedas.

Dado que las técnicas anteriores, cuando se trabaja con más de una variable explicativa, no tienen en cuenta la posible interacción entre ellas, hemos presentado los splines de regresión Thin Plate y los Tensores, dos procedimientos que sí permiten estudiar la interacción entre las variables explicativas. En el último modelo expuesto hemos observado cómo el uso de los Tensores sí consigue reflejar las constantes variaciones sufridas por el precio a lo largo del día. Esta técnica también consigue proporcionarnos información útil sobre qué casa de cambio elegir para realizar cualquier transacción de una determinada cantidad de criptomonedas.

Con esta aplicación para datos de criptomonedas hemos mostrado las posibilidades que ofrecen las distintas técnicas usadas para realizar el estudio del Modelo Aditivo Generalizado. También hemos evidenciado la importancia de implementar técnicas de

captura de la información en tiempo real y de estimar los modelos en el menor tiempo posible para mejorar las predicciones.

Bibliografía

[1] BRUSCH, Khai T. *Generalized Additive Models for very large datasets with Apache Spark*, Universidad de Hamburgo, 2016.

[2] Coinmarketcap. *Disponible en* `https://coinmarketcap.com/es/intro-to-crypto/what-are-cryptocurrencies/`. Consultado el 1 de mayo de 2019.

[3] Criptomoneda.Ninja. *Disponible en* `https://criptomoneda.ninja/blockchain-cadena-bloques/`. Consultado el 1 de mayo de 2019.

[4] DOUBOVA, Anna & ECHEVARRÍA, Rosa. *LaTeX Composición de textos científicos con el ordenador*, Universidad de Sevilla, 2018.

[5] DURBÁN, María *Modelos Aditivos Generalizados con P-splines*, Universidad Carlos III de Madrid, 2018.

[6] Economipedia. *Disponible en* `https://economipedia.com/definiciones/criptomoneda.html`. Consultado el 3 de mayo de 2019.

[7] GARCÍA, Joaquín. *Apuntes de la asignatura de Modelos Lineales. Tema 3*, Universidad de Sevilla, 2017.

[8] HASTIE, T. ; TIBSHIRANI, R. *Generalized Additive Models (with discussion)*, Statistical Science 1, 1986.

[9] JAMES, Gareth ; WITTEN, Daniela ; HASTIE, Trevor ; TIBSHIRANI, Robert. *An Introduction to Statistical Learning: With Applications in R*, Springer Publishing Company, 2014.

[10] Luque, P.L. *Escribir un Trabajo Fin de Estudios con R Markdown*, Disponible en `http://destio.us.es/calvo` ,2015.

[11] McCullagh, P. ; Nelder, J.A. *Generalized Linear Models*, London:Chapman and Hall, 1989

[12] Nelder, J.A. ; Wedderburn, R.W.M. *Generalized Linear Models* , Journal of the Royal Statistical Society, 1972

[13] Oroyfinanzas. *Disponible en* `https://www.oroyfinanzas.com/2014/10/que-es-criptomoneda/`. Consultado el 3 de mayo de 2019.

[14] PINO, J.L. *Apuntes Master Big Data y Data Science*, Universidad de Sevilla, 2016.

[15] WOOD, Simon N. *Generalized Additive Models: An Introduction with R*, Chapman and Hall/CRC, 2006.