CAPÍTULO III

Árboles de decisión en economía: una aplicación a la determinación del precio de la vivienda

Máximo Camacho*
Salvador Ramallo
Manuel Ruiz Marín

En este trabajo describimos los árboles de decisión y repasamos su utilización en el análisis de datos económicos en el contexto de big data. De manera simplificada, realizamos un análisis comparativo de los algoritmos más utilizados en la construcción de árboles de decisión: CART, *bagging, random forests* y *boosting.* A modo de ilustración de estos procedimientos, examinamos cuáles son los determinantes del precio de la vivienda en Madrid en una muestra de 20.386 viviendas del portal Idealista en 2018. El fuerte componente no lineal de la relación del precio con sus determinantes es clave para destacar las ventajas de los árboles de decisión frente a propuestas más convencionales.

Palabras clave: CART, big data, precio de la vivienda.

* Los autores agradecen los comentarios de Fernando López y la financiación de los proyectos PID2019-107192 GB-I00 (AEI/10.13039/501100011033) y del Grupo de Excelencia de la Región de Murcia (Fundación Séneca) 19884/GERM/15. Los datos y programas de R que replican los principales resultados se encuentran en la web de los autores. Los errores cometidos son nuestra responsabilidad.

1. INTRODUCCIÓN

No se sabe con seguridad quién acuñó el termino *big data* por primera vez, ni siquiera si fue dentro o fuera del ámbito académico, aunque parece, en cualquier caso, que el mérito debería ser compartido (Diebold, 2012). Actualmente, el término recoge la esencia de un fenómeno de alto impacto en la ciencia y, en particular, en la economía y la empresa. Aunque no resulta fácil definir big data, las características de este tipo de análisis se pueden resumir en tres magnitudes: volumen, variedad y velocidad. Big data hace referencia a un gran volumen de datos, en numerosas ocasiones de naturaleza variada, organizados de manera estructurada o no estructurada, a veces disponibles (casi) en tiempo real y con una alta frecuencia, facilitado por los recientes avances en la tecnología de la información. Debido a estas características, el análisis estadístico y la visualización de los resultados obtenidos a partir de big data supone, en numerosas ocasiones, un reto.

Con el fin de superar este reto, se han propuesto en la literatura reciente numerosas técnicas de partición recursiva de datos encaminadas al tratamiento de big data. Estas técnicas se clasifican en dos grandes grupos: algoritmos supervisados y algoritmos no supervisados. Los algoritmos de aprendizaje supervisados son aquellos en los que tanto las variables de entrada como de salida están etiquetadas y son conocidas. Ejemplos de este tipo de algoritmos son la regresión lineal, regresión logística, maquinas de vectores de soporte, redes neuronales artificiales, k-vecinos mas cercanos y árboles de regresión y clasificación.

Por otro lado, la principal característica de los algoritmos de aprendizaje no supervisados es que pueden aprender a identificar procesos y patrones complejos sin que se les proporcione un etiquetado ni de las variables de entrada ni a las de salida. Están, por tanto, más orientados al aprendizaje de máquinas en *clustering* y reducción de la dimensionalidad. Ejemplos de algoritmos no supervisados son el análisis clustering por k-medias, inicialmente propuesto por MacQueen (1967), y el análisis de componentes principales.

En este capítulo nos centraremos en algoritmos supervisados basados en árboles de regresión y clasificación (*CART,* por sus siglas en inglés), popularizados tras la aparición de los algoritmos desarrollados por Breiman *et al.* (1984). Aunque no existe una herramienta de análisis de big data que sea superior a las demás en todos los escenarios, el comportamiento de los CART es bastante competitivo, y en algunos casos superior a otras alternativas como la regresión lineal, regresión logística, maquinas de vector soporte o redes neuronales artificiales (Kurt, Ture, y Kurum, 2008; Choubin *et al.,* 2018; Yang *et al.*, 2017).

Los CART se han utilizado en diversas ramas de la ciencia como en ecología (De'ath y Fabricius, 2000) para describir los tipos de hábitat de coral a partir de variables ambien-

tales; en medicina (Austin *et al.,* 2012) para predecir la mortalidad en pacientes con enfermedades cardiovasculares; en epidemiología (Austin, 2013) para clasificar y predecir enfermedades; en transporte (Ghasri, Rashidi, y Waller, 2017) para seleccionar modelos de transporte alternativos; o en psicología (Kitsantas, Moore y Sly, 2007) para estudiar los determinantes del consumo de tabaco.

Debido a la ingente cantidad de aportaciones en diversas ramas de la literatura (Loh, 2011; James *et al.,* 2013), en este trabajo introduciremos los CART poniendo especial atención a sus aplicaciones para la economía y la empresa. Aunque dedicaremos una sección completa a describir las aplicaciones más significativas de modelos CART en estas disciplinas, algunos ejemplos son la predicción de recesiones (Ng, 2014, Döpke, Fritsche y Pierdzioch, 2017); el abandono de clientes (Xie *et al.,* 2009); la predicción del fracaso empresarial (Gepp, Kumar y Bhattacharya, 2010); la detección de fraude financiero (Liu *et al.,* 2015) y la predicción de crisis financieras (Ward, 2017).

Los motivos por los que hemos seleccionado los CART para ilustrar el análisis de big data en economía y empresa son los mismos que están detrás del creciente éxito de esta metodología:

- Son una herramienta no paramétrica de regresión y clasificación de observaciones muy simple y flexible, permitiendo el uso de datos continuos y/o categóricos y bastante robusto a la presencia de datos atípicos. Por tanto, resultan una herramienta muy útil para capturar relaciones complejas entre las variables.

- Permiten operar con bases de datos muy grandes ya que funcionan con algoritmos muy eficientes desde el punto de vista computacional.

- Los resultados obtenidos con estas técnicas son muy fáciles de interpretar porque suelen ir acompañados de representaciones gráficas muy intuitivas que ayudan a discriminar cuáles son las variables explicativas más relevantes y cómo se relacionan con la variable dependiente.

En concreto, en este trabajo realizamos una introducción a los CART, centrándonos en datos de corte transversal aunque resultaría relativamente sencillo adaptarlos a datos de series temporales. Para ello, describiremos sus componentes y los métodos más habituales para su construcción y para la realización de predicciones económicas y empresariales. Acompañaremos la presentación de los CART con un ejemplo cuyo principal objetivo es esclarecer los determinantes del precio de la vivienda en Madrid en una muestra de 20.386 viviendas obtenidas a partir del portal Idealista en 2018. Como resultados más interesantes, obtenemos que el principal determinante del precio es la superficie de la vivienda, seguida del número de baños y habitaciones. La posición geográfica se encuentra también entre las variables explicativas más importantes. Como

ilustración de la capacidad de los CART para capturar relaciones complejas, mostraremos que la relación entre la posición geográfica y el precio de la vivienda es fuertemente no lineal. Por tanto, los métodos econométricos tradicionales serían incapaces de capturar la complejidad de esta relación.

El resto del artículo está estructurado como sigue. En la sección segunda, establecemos la notación y describimos los métodos de formación de un árbol de decisión. La sección tercera está destinada a desarrollar los métodos de *bagging, random forest* y *boosting* aplicados a controlar la inestabilidad de los árboles de decisión. En la cuarta sección repasamos algunos ejemplos de árboles de decisión aplicados al tratamiento de datos de economía y empresa. En la sección 5 concluimos.

2. ÁRBOLES DE REGRESIÓN Y CLASIFICACIÓN

Los árboles de decisión aparecen por primera vez en la literatura en el trabajo de Morgan y Sonquist (1963), aunque se popularizaron tras los algoritmos CART. Bajo estas siglas se encuentra un conjunto de algoritmos supervisados que tratan de modelizan una variable de respuesta a partir de un conjunto de p variables explicativas. Los árboles de decisión aplicados a variables de respuesta que toman valores numéricos con un orden intrínseco se denominan árboles de regresión. Cuando la variable de respuesta es cualitativa se aplican árboles de clasificación.

Un CART se entrena mediante un conjunto de datos que se usa para crear particiones recursivas, normalmente ortogonales, del espacio de definición del conjunto de variables explicativas. De esta manera las particiones son un conjunto de prismas dimensionales. Esto se consigue optimizando localmente en cada paso una función de pérdida cuya especificación depende de si el objetivo es la regresión o la clasificación.

En esta sección detallaremos la construcción del árbol de regresión y clasificación, definiremos la notación básica y los diferentes modelos de predicción. Aunque existen varios métodos para construir árboles de decisión, principalmente *CHAID (Chi-Square Automatic Interaction Detector), QUEST (Quick Unbiased Efficient Statistical Tree)* y CART, nosotros nos centraremos en los CART. Una comparativa de los algoritmos de construcción de CART más populares, como ID3, CART, C4.5 y See5/C5.0, se puede encontrar, por ejemplo, en Venkata y Kiruthika (2015)[1].

2.1. Construcción del árbol de regresión y clasificación

Con el fin de establecer la notación necesaria, consideremos una variable de respuesta Y que tratamos de modelizar a partir de un conjunto de p variables explicativas, $X = (X_1,...,X_p) \in \mathbb{R}^p$ El algoritmo CART se basa en encontrar las particiones del espacio

[1] Básicamente, los algoritmos se diferencian en las estrategias utilizadas para realizar las particiones.

de definición de X en un conjunto de M regiones terminales disjuntas $\{R_1,R_2,...,R_M\}$, que minimizan una determinada función de pérdida.

Para facilitar la descripción del algoritmo, consideremos un CART de particiones binarias con M nodos terminales, cada uno de los cuales $m \in \{1,...,M\}$ representa una rama del árbol que se caracteriza por las m^* escisiones que se producen en los nodos internos[2]. El espacio de parámetros que define cada rama lo denotaremos por $\theta_m^{m^*} = \left(v_{m,1}^{d_1}, s_{m,1}, ..., v_{m,m^*}^{d_{m^*}}, s_{m,m^*}\right)$, donde $X_{v_{m,i}}$ es la variable que determina la escisión i-ésima, con $i \in \{1,...,m^*\}$, del nodo terminal m, con $v_{m,i} \in \{1,...,p\}$. En el contexto de particiones ortogonales, para cada punto de escisión $s_{m,i}$, diremos que $d_i = 0$ si $X_{v_{m,i}} \leq s_{m,i}$ mientras que $d_i = 1$ si $X_{v_{m,i}} > s_{m,i}$. Por tanto, cada escisión i de una rama m dará lugar a dos subregiones, R_{mi}^0 cuando $d_i = 0$ y R_{mi}^1, cuando $d_i = 1$.

Como no es posible evaluar todas las particiones posibles, el objetivo de CART es proponer un mecanismo de partición recursiva en la que en cada paso se divide en dos subregiones una de las regiones creadas en el paso anterior. El criterio para realizar las particiones es encontrar aquella que minimiza una función de pérdida en el espacio de definición de las variables explicativas $\mathcal{X}$ y que dará lugar a M regiones finales disjuntas. La recursividad del algoritmo implica que el camino seguido por las particiones determinadas por la m-ésima rama es una sucesión de particiones θ_m^i tales que,

$$\theta_m^i = \theta_m^{i-1} \bigcup \left(v_{m,i}^{d_i}, s_{m,i}\right). \qquad [1]$$

con $i = 1,...,m^*$.

Supongamos que disponemos de una muestra de T observaciones, $\{(Y_1,\mathcal{X}_1),...,(Y_T,\mathcal{X}_T)\}_{t=1}^T$, donde Y_t es el valor de la variable dependiente o de respuesta y $\mathcal{X}_t$ es el vector de valores de las p variables explicativas para la observación t. Para cada escisión i que se produce en cada rama m definimos una función de predicción $f\left(\mathcal{X},\theta_m^i\right)$. Generalmente se trata de una medida relacionada con el valor que toma la variable dependiente condicionada a las regiones que se producen en la escisión. En árboles de regresión suele ser una medida de posición central y en árboles de clasificación la región se suele asociar al evento más probable. A la predicción para la subregión $R_{mi}^{d_i}$ de la rama m que se produce en la escisión i la llamaremos $\hat{Y}\left(R_{mi}^{d_i}\right)$.

Asociado a la función de predicción, en cada escisión de cada rama se define la función de pérdida,

$$L\left(Y, f\left(\mathcal{X},\theta_m^i\right)\right). \qquad [2]$$

[2] En vez de particiones binarias se podrían realizar particiones múltiples. Sin embargo, en la práctica no suelen funcionar bien porque particionan el espacio de definición demasiado rápido, dejando pocas observaciones para las nuevas particiones.

En árboles de regresión, la función de pérdida asociada a una escisión suele ser la suma del error cuadrático. Denotemos por $T_{mi}^{d_i}$ el número de observaciones que hay en la región $R_{mi}^{d_i}$. Si usamos la media como medida de posición central, podemos determinar la función de pérdida de cada escisión como:

$$L\left(Y, f\left(\mathcal{X}, \theta_m^i\right)\right) = \Sigma_{\mathcal{X}_t \in R_{mi}^0}\left(Y_t - \overline{Y}_{R_{mi}^0}\right)^2 + \Sigma_{\mathcal{X}_t \in R_{mi}^1}\left(Y_t - \overline{Y}_{R_{mi}^1}\right)^2 \quad [3]$$

donde, $\overline{Y}_{R_{mi}}^{d_i} = \frac{1}{T_{mi}^{d_i}} \Sigma_{\mathcal{X}_t \in R_{mi}^{d_i}} Y_t$ es la media de Y en la región $R_{mi}^{d_i}$ con $d_i \in .\ \{0,1\}$

En árboles de clasificación, la función de pérdida asociada a una escisión suele ser una medida del grado de impureza de las regiones resultantes. Supongamos que hay K clases distintas. Sea $I(\bullet)$ la función indicador que vale 1 si se cumple la condición dentro del paréntesis y 0 en caso contrario. Entre las diversas alternativas, vamos a elegir el índice de Gini para medir la impuridad o grado de impureza de la región m-ésima como:

$$IG\left(R_{mi}^{d_i}\right) = \Sigma_{k=1}^{K} P_k\left(R_{mi}^{d_i}\right)\left(1 - P_k\left(R_{mi}^{d_i}\right)\right), \quad [4]$$

donde,

$$P_k\left(R_{mi}^{d_i}\right) = \frac{1}{T_{mi}^{d_i}} \Sigma_{\mathcal{X}_t \in R_{mi}^{d_i}} I\left(Y_t = k\right),$$

es la proporción de observaciones de la región $R_{mi}^{d_i}$ que se clasifican como clase k de las K posibles. La función de pérdida de cada escisión se determina como la media ponderada de la impuridad de las regiones resultantes. Específicamente si $\mathcal{X} \in R_{mi}$

$$L\left(Y, f\left(\mathcal{X}, \theta_m^i\right)\right) = \frac{T_{mi}^0}{T_{mi}^0 + T_{mi}^1} IG\left(R_{mi}^0\right) + \frac{T_{mi}^1}{T_{mi}^0 + T_{mi}^1} IG\left(R_{mi}^1\right). \quad [5]$$

Si las regiones resultantes son muy puras, en el sentido de que contienen observaciones de una sola clase, el grado de impureza de esas regiones, medido por la expresión [4], será próximo a cero y, en ese caso, la función de pérdida también será próxima a cero. Cuanto mayor sea la mezcla de clases en las regiones, mayor será su grado de impureza y, por ende, mayor será el valor de la función de pérdida.

La descripción del algoritmo es muy sencilla. Al comienzo, todas las observaciones están en una única región que coincide con todo el espacio de definición de $\mathcal{X}$. Para realizar la primera escisión, seleccionaremos el indicador X_i y el punto de escisión s_i que den lugar a las dos regiones que minimicen la función de pérdida. Para cada una de esas dos regiones resultantes, volvemos a buscar el indicador y el punto de escisión que minimizan la función de pérdida, pero solo se realiza la escisión de la región que de lugar a una función de pérdida menor. El algoritmo se repite sucesivamente hasta que

se alcanza algún criterio que detiene al algoritmo. Algunas opciones habituales son que el número de observaciones que pertenecen a las regiones finales no sea inferior a un mínimo establecido por el usuario, que el árbol tenga un número máximo de regiones finales o que la reducción en la función de pérdida tenga que superar un mínimo para que tenga lugar una nueva escisión. En el apartado 2 de esta sección veremos una técnica más sofisticada.

Una vez que el algoritmo termina, la última etapa consiste en dar una predicción a las observaciones que se encuentran en cada una de las regiones terminales. En el caso de árboles de regresión, la predicción en la región m, será la media de las observaciones que pertenezcan a esa región, $\hat{Y}(R_m) = \overline{Y}_{R_m}$. Para los árboles de clasificación, la predicción en la región m será la clase k^* para la que se alcance el mayor valor de la proporción de observaciones de cada clase en esa región entre las $k=1,...,K$ clases posibles, $\hat{Y}(R_m) = P_{k^*}(R_m)$.

Para conocer la verdadera capacidad de predicción del árbol de decisión debemos examinar hasta qué punto es capaz de formar inferencia adecuada de las observaciones que no se han usado para generar el árbol. Como los árboles se encuentran dentro de lo que conocemos como métodos de aprendizaje supervisado basado en la experiencia pasada, resulta útil dividir la muestra en dos submuestras. La primera submuestra es la de entrenamiento o training, en la que se estima el CART. La segunda es la de evaluación o test, en la que se examina la habilidad del CART para formar inferencia de las observaciones que no se han usado para estimarlo[3].

Una vez que se ha estimado el árbol de decisión, una herramienta muy útil para interpretar el resultado obtenido es la importancia relativa propuesta por Breiman *et al.* (1983). La importancia relativa del indicador X_i entre los $(X_1,...,X_p)$ indicadores existentes viene determinada por el número de veces que ese indicador se ha utilizado para realizar las particiones ponderada por la reducción en la función de pérdida (error cuadrático o índice de Gini) que proporcionan las particiones en las que el indicador participa.

Supongamos que en un árbol se hacen un total de M^* subparticiones (una por cada nodo del árbol) y que el indicador que se usa para hacer la partición m-ésima es X_{v_m}, con $v_m \in \{1,...,p\}$ y $m \in \{1,...,M^*\}$. Si llamamos ∇L_m a la reducción de la función de pérdida que se produce en esa partición, la importancia relativa del indicador X_i en la construcción del árbol de decisión:

$$IR_i = \Sigma_{m=1}^{M^*} \nabla L_m I\left(X_{v_m} = X_i\right). \qquad [6]$$

[3] Sin pérdida de generalidad, describiremos las técnicas suponiendo que la submuestra de entrenamiento es la muestra total. La distinción entre submuestras de entrenamiento y evaluación se hará cuando se examine la habilidad predictiva de las distintas técnicas empleadas.

donde $i=1,...p$. Para facilitar la interpretación, el indicador se suele normalizar para que las importancias relativas de los p indicadores sumen 100[4].

Una segunda herramienta habitual en el contexto de árboles de decisión es el Gráfico de Dependencia Parcial, que mide el efecto de una variable explicativa sobre la variable dependiente. Sin pérdida de generalidad, supongamos que queremos medir el efecto *céteris páribus* de la primera variable explicativa X_1 sobre la predicción del árbol de decisión. El primer paso para calcular la dependencia parcial consiste en construir el árbol de decisión usando las técnicas que hemos descrito con anterioridad para la base de datos original. En el segundo paso, generamos un conjunto de posibles valores para X_1, que llamaremos $X_{1i} \in \{X_{11},...,X_{1N}\}$. Llamemos al conjunto formado por cada valor generado de X_1 y el resto de valores de las explicativas $\mathcal{X}(1i) = (X_{1i},X_2,...,X_p)$. Para cada uno de los elementos de este conjunto haremos una predicción con el árbol estimado y calcularemos la media:

$$\hat{Y}^*\left(X_{1i}\right)=\frac{1}{N}\Sigma_{i=1;\mathcal{X}(1i)\in R_k}^{N}\hat{Y}\left(R_k\right). \qquad [7]$$

El gráfico de $\left\{X_{1i},\hat{Y}^*\left(X_{1i}\right)\right\}$, para $i=1,...,N$, es el Gráfico de Dependencia Parcial de la variable explicativa X_1.

Se puede generalizar el Gráfico de Dependencia Parcial para cualquier subconjunto de variables explicativas incluidas en X. Aunque el efecto sobre la variable dependiente de combinaciones complejas de las variables explicativas va a ser difícil de interpretar, la opción que más se utiliza en la literatura es la que analiza el efecto de la combinación de dos variables explicativas, dando lugar a los gráficos que miden el efecto interacción. Sin pérdida de generalidad, supongamos que estamos interesados en medir el efecto conjunto de las dos primeras variables explicativas, X_1 y X_2, sobre la variable dependiente Y. Siguiendo el razonamiento de los Gráficos de Dependencia Parcial, el efecto parcial se aproxima generando pares de valores de las dos primeras variables explicativas, (X_{1i},X_{2i}), con $i=1,...,N$; para cada valor de i, predecimos el valor de la variable dependiente usando los valores reales del resto de variables explicativas y calculamos la media, $\hat{Y}_{\tau}^*\left(X_{1i},X_{2i}\right)$, siguiendo la lógica propuesta en la expresión [7]; por último, representamos gráficamente $\left\{X_{1i},X_{2i},\hat{Y}^*\left(X_{1i},X_{2i}\right)\right\}$, para $i=1,...,N$.

2.1.1. *Prunning*

La estimación del árbol de decisión con el algoritmo anterior sufre el riesgo de sobreajuste. Supongamos que el proceso generador de datos es $Y = f(\mathcal{X},\Theta_M) + u$. Como la estimación que realizamos con el árbol depende de sus regiones terminales y éstas de

[4] Otras veces se hace 100 la importancia relativa del indicador más importante y se relativizan las importancias relativas del resto de indicadores respecto al más importante.

los parámetros estimados $\hat{\Theta}_M$, podemos llamar al estimador resultante del árbol de decisión $\hat{f} = f\left(\mathcal{X}, \hat{\Theta}_M\right)$. La descomposición que viene determinada por el Error Cuadrático Medio (ECM) para ese estimador implica:

$$ECM = \left(E\hat{f} - f\right)^2 + var\left(\hat{f}\right) + var\left(u\right), \qquad [8]$$

donde $f = f\left(\mathcal{X}, \Theta_M\right)$. Si obviamos el último componente, referido a la varianza irreducible del error, la estimación $\hat{f}$ puede modificar el sesgo y la varianza del árbol de decisión. Cuanto más complejo sea el árbol de decisión, medido como el número de regiones finales M, menor será el sesgo pero mayor será la varianza y esto provoca falta de robustez en los resultados provocando fallos en las predicciones. La poda de un árbol de decisión consiste en encontrar un árbol con un número de regiones óptimo que equilibre la balanza entre el sesgo y la varianza.

Sea $f = f\left(\mathcal{X}, \Theta_{|M}\right)$ la función de pérdida calculada a partir de las M regiones del árbol. Para un árbol de regresión, la pérdida estimada será:

$$\Sigma_{m=1}^{M} \Sigma_{\mathcal{X}_t \in R_m} \left(Y_t - \hat{Y}\left(R_m\right)\right)^2. \qquad [9]$$

Si definimos $Q(R_m)$ como la proporción de observaciones de la región m sobre el total de observaciones, para un árbol de clasificación la pérdida será:

$$\Sigma_{m=1}^{M} Q\left(R_m\right)\left(1 - \Sigma_{k=1}^{K} P_k\left(R_m\right)^2\right), \qquad [10]$$

que se interpreta como la suma ponderada de la impureza de las regiones terminales.

Dado un valor de α, la técnica consiste en comenzar a generar el árbol con M_1 regiones. Seguidamente, aumentamos las escisiones sucesivamente incrementando el valor de las regiones terminales hasta que lleguemos a un árbol donde no se puedan encontrar más escisiones sin que las M_N regiones resultantes tengan menos observaciones que un número fijado. Para el valor dado de α, calculamos el valor de la expresión:

$$PR_{M_i}\left(\alpha\right) = L\left(Y, f\left(\mathcal{X}, \Theta_{M_i}\right)\right) + \alpha M_i \qquad [11]$$

para cada valor de M_i. De la colección de valores $\{PR_{M_1}(\alpha), \ldots, \{PR_{M_N}(\alpha)\}$, elegiremos el árbol con un número de regiones M para el que [11] alcance el valor más pequeño. A medida que el árbol es más complejo, la función de pérdida que aparece en el primer sumando tiende a caer. Sin embargo, el incremento en el número de regiones hace que el segundo sumando crezca en proporción al valor de α. Si α es cero el tamaño del árbol será muy grande y crecerá la varianza, mientras que si α es demasiado alto el árbol

resultante podría ser demasiado simple como para garantizar una mínima bondad del ajuste a los datos.

Para determinar el tamaño óptimo de que controla el *trade-off* entre el ajuste a los datos y la complejidad del árbol resultante, se suelen utilizar técnicas de validación cruzada. El primer paso de estas técnicas consiste en dividir la muestra en J submuestras seleccionadas aleatoriamente. Usaremos todas las muestras menos una, que llamaremos submuestra j, para estimar los árboles y dejaremos la submuestra j para evaluarlos. En segundo lugar fijamos un valor de α pequeño, que llamaremos α_1, y para ese valor se estiman árboles para distintos valores de M_i que se evalúan en la submuestra j. Para cada submuestra se elige el número de regiones que minimiza [11] evaluado en la muestra j. Seguidamente para cada una de las j - 1 submuestras se calcula la media $\overline{PR}(\alpha_1)$. Finalmente, se repite esta operación incrementando en cada paso el valor de α hasta un valor máximo α_L. El valor de α elegido será el valor,

$$\alpha^* = \arg\min_{\alpha_i}\left\{\overline{PR}(\alpha_1),\ldots,\overline{PR}(\alpha_L)\right\} \qquad [12]$$

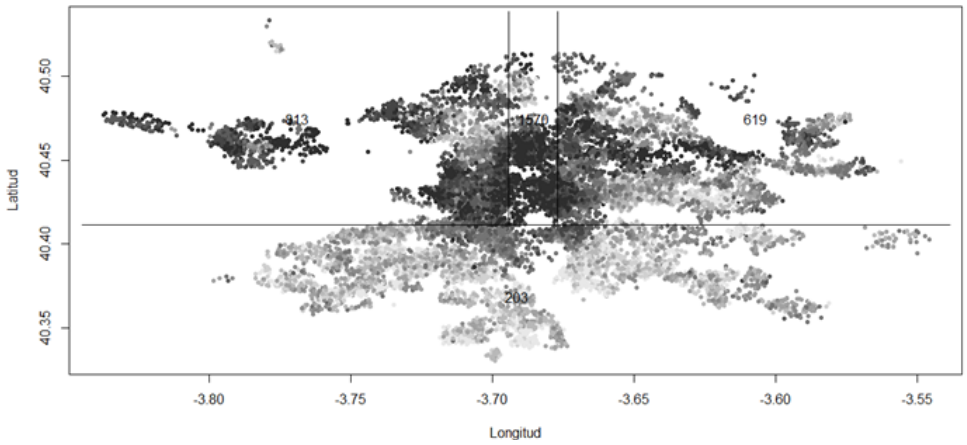
que minimiza la media.

2.2. Ejemplo de árbol de decisión

Para ilustrar el mecanismo de construcción de los árboles de decisión, vamos a usar una base de datos de precios de viviendas (en miles de euros) en el municipio de Madrid

FIGURA 1

PRECIO DE LA VIVIENDA Y PARTICIÓN



Fuente: Elaboración propia.

extraídos a partir de la base de datos del portal Idealista referido a las 20.385 viviendas que se anunciaron en el portal en enero de 2018[5]. La figura 1 muestra la localización en un mapa de los precios de la vivienda teniendo en cuenta su posición en términos de latitud y longitud de sus coordenadas geográficas. Para facilitar la inspección visual de los datos, cada vivienda aparece en el plano con un color más oscuro cuanto mayor es su precio. En la zona oeste no se venden viviendas en la zona central del distrito de Moncloa-Aravaca, donde se encuentra la Casa de Campo y en la zona norte donde se sitúan los montes del distrito Fuencarral-El Pardo.

Parece evidente que la localización geográfica tiene importancia en la determinación de los precios. En concreto, el mapa muestra que los precios más baratos se localizan en la latitud sur de Madrid donde están los distritos de la Latina, Carabanchel, Usera, Villaverde, Puente de Vallecas, Villa de Vallecas y Vicálvaro. Sin embargo, el precio de la vivienda se encarece para las viviendas que están en el centro, y en menor medida en el norte. Respecto a la zona centro-norte, los precios más altos están en la longitud central, donde se encuentran los distritos Centro, Retiro, Salamanca y Chamartín. En segundo lugar, aparecen las viviendas situadas en el oeste, en los distritos de Chamberí y algunas urbanizaciones de Fuencarral-El Pardo y de Moncloa-Aravaca. En tercer lugar, la zona situada más al este de la zona norte donde se encuentran Ciudad Lineal, Hortaleza y Barajas. Estas diferencias por longitud no están tan acentuadas en la zona sur.

El objetivo de un árbol de decisión donde las explicativas sean la latitud y la longitud consiste en particionar recursivamente este mapa geográfico dando lugar unas regiones para las que se minimice el error cuadrático entre el precio de cada vivienda y el precio medio de la región resultante de la partición a la que cada vivienda pertenezca. Con un fin únicamente ilustrativo, vamos a estimar un árbol de decisión con un máximo de tres particiones. Para visualizar el resultado, las cuatro regiones resultantes se superponen en el mapa que se muestra en la figura 1, junto con el precio medio de las viviendas localizadas en esas regiones. El precio estimado para una vivienda situada al sur es de unos 200.000 euros. Entre las viviendas del centro-norte, las que que se localizan en el este son las más baratas, con un precio esperado de unos 620.000 euros. Para las que se sitúan en el oeste estimamos un precio de unos 810.000 euros. Para las que se sitúan en el centro, estimamos un precio de un 1.600.000 euros aproximadamente.

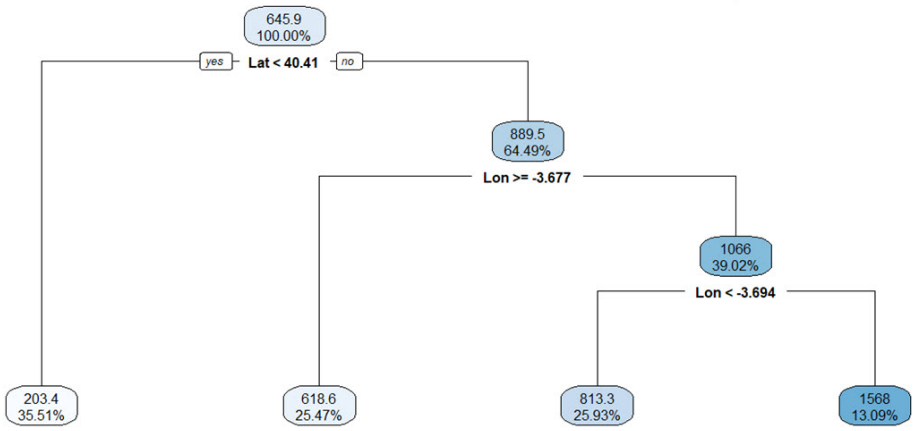
La figura 2 muestra el algoritmo recursivo que se ha utilizado para realizar las particiones. Al comienzo del algoritmo, todas las viviendas están situadas en una única región, para las que el precio medio es de 646.000 euros. El árbol comienza con la latitud y para un conjunto de puntos de escisión calcula la suma al cuadrado de la diferencia entre los precios de la vivienda y la media de las dos regiones resultantes a las que pertenecen las viviendas, tal y como aparece en la ecuación [10]. Acto seguido, realiza la

[5] Fan, Ong y Koh (2006) estiman un árbol de decisión mediante el algoritmo CART para analizar los determinantes del precio de la vivienda en Singapore. Aunque se fijan en características de la localización, como cercanía de colegios, no tienen en cuenta las coordenadas geográficas.

misma operación para longitud y un conjunto de puntos de escisión posibles para esa variable. Finalmente, elige la pareja de variable (latitud o longitud) y punto de escisión que minimizan [10]. En este caso, el mínimo lo ha encontrado para la variable latitud y con punto de escisión 40,41. Como se observa en figura 2, la región que se encuentra más al sur de esa latitud (pare izquierda de la figura) no se vuelve a particionar y se estima un precio medio para esas viviendas, que suponen un 35 % del total, de unos 200.000 euros.

FIGURA 2

ÁRBOL DE DECISIÓN



Fuente: Elaboración propia.

La región que queda al norte de la latitud 40,41 se vuelve a particionar de la misma manera que hemos descrito anteriormente, siendo ahora la longitud la variable elegida para realizar la partición y el punto de escisión que minimiza [10] de -3,677. La región que se encuentra al este este de esa longitud ya no se vuelve a particionar y el precio que se estima para las viviendas de esa región es de unos 620.000 euros. Sin embargo, la región que se encuentra al oeste de la longitud -3,677 se particiona de nuevo usando a la longitud como variable y el punto de escisión de -3,694. Las dos regiones son la que se encuentran al oeste de esa longitud para la que se estima un precio de la vivienda de unos 810.000 euros, y la región central (al este de -3,694 pero al oeste de -3,677) para la que se estima un precio de la vivienda de casi 1.600.000 euros.

Aunque didáctico, el ejemplo que hemos descrito hasta ahora es demasiado poco ambicioso. Como veremos más adelante, ni siquiera podemos asegurar que el efecto *ceteris paribus* de la latitud y la longitud sea el que hemos detectado en el ejercicio anterior porque el modelo es tan sencillo que puede esconder relaciones más complejas entre el precio de la vivienda y su posición geográfica cuando se añadan otras variables explicativas. El análisis de big data con árboles de decisión puede ser mucho más útil

con modelos más completos, como vamos a ilustrar en el análisis más completo del precio de la vivienda que proponemos.

Supongamos que queremos vender o comprar una vivienda en Madrid pero no sabemos cuál sería el precio adecuado al que deberíamos realizar la transacción. Las coordenadas geográficas, aunque potencialmente importantes, no parecen ser la única característica que tienen en cuenta compradores y vendedores a la hora de determinar los precios de sus transacciones. Generalmente, además de la posición geográfica, el mercado de la vivienda toma en cuenta una serie de características de la vivienda tales como superficie, planta, tipo (casa, piso, ático, estudio, chalet o dúplex), número de habitaciones y baños, si se encuentra en buen estado, si es exterior, o si la vivienda dispone de terraza, ascensor, aire acondicionado, garaje, piscina, trastero, armarios empotrados, o jardín.

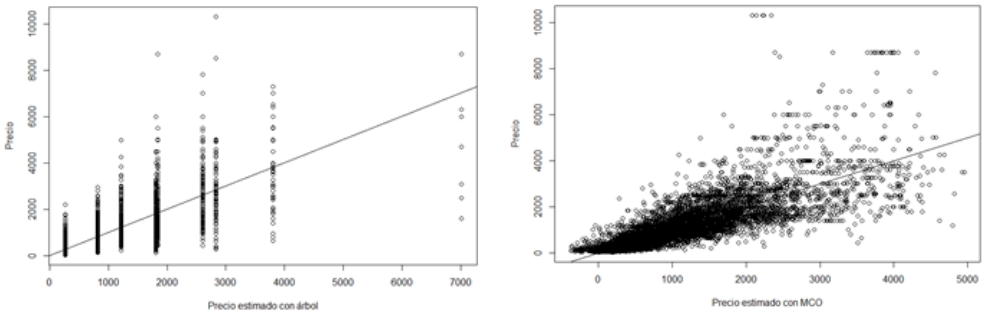
Otra ventaja de los árboles de decisión en el uso de grandes bases de datos es la sencillez con la que un usuario poco familiarizado con técnicas estadísticas sofisticadas puede utilizar el resultado del modelo para tomar decisiones económicas. Usando las técnicas de pruning que se han descrito anteriormente, el árbol que minimiza la expresión [11] es el que tiene nueve regiones. Un usuario sólo tiene que conocer el valor de las variables explicativas de la vivienda que quiere vender o comprar para que el árbol localice en qué región se encuentra la combinación de las explicativas y proponer como precio de la vivienda la media de las viviendas que se quieren comprar y vender en esa región. Por ejemplo, la vivienda de la última observación de la muestra se trata de un piso exterior de una habitación en segunda planta con ascensor y un baño, con unas coordenadas de longitud -3,702 y latitud 40,452, de 68 metros cuadrados, que no necesita reforma y no tiene jardín, trastero, piscina, aire acondicionado ni terraza. El precio propuesto por el árbol de decisión para esa vivienda es de 263.000 euros, algo más barato del precio en el que aparece en Idealista de 280.000 euros.

La predicción del árbol de decisión en cada una de las nueve regiones para todas las viviendas de la muestra, junto con el precio que se ofrecen en la plataforma aparece en el gráfico izquierdo de la figura 3. Para facilitar la interpretación, se ha incluido una línea que corta al eje y tiene pendiente unitaria. Esta figura permite apreciar cómo aumenta la heterogeneidad en el precio de las viviendas en las distintas regiones que se forman en el árbol de decisión a medida que aumenta el precio medio de la vivienda predicho para esas regiones. Como comparativa, en el gráfico derecho de la figura 3 aparece la estimación realizada por el modelo lineal estimado por mínimos cuadrados ordinarios. Como resumen de la bondad del ajuste de los modelos, el pseudo R^2, medido como uno menos la proporción del ECM sobre la varianza del precio alcanza un valor de 0,73 para el árbol de decisión y de 0,72 para MCO.

Un problema importante de la construcción de los árboles de decisión que hemos descrito es el de la falta de estabilidad ante pequeños cambios en el proceso de construcción del árbol. Aunque vamos a analizar este problema con detalle en la próxima

FIGURA 3

PREDICCIÓN DEL PRECIO DE LA VIVIENDA

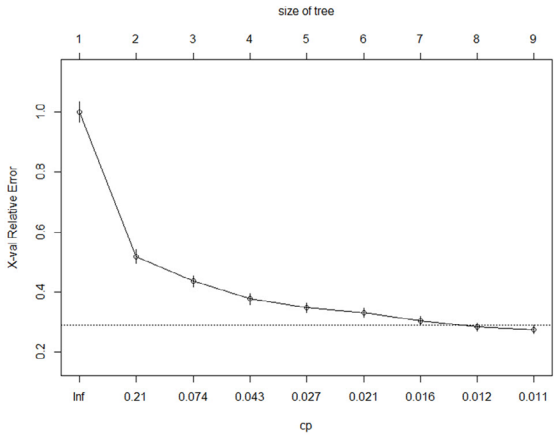


Fuente: Elaboración propia.

sección, podemos ilustrar uno de los efectos de la inestabilidad analizando el resultado de la validación cruzada que se ha utilizado para seleccionar un árbol con nueve regiones. En la figura 4 se muestra el valor de la expresión [11] para cada valor de α (escala inferior del gráfico) y del número de regiones que se eligen para cada α[6] Si elegimos el número de regiones como el valor que minimiza la expresión [11], el árbol debería tener nueve regiones.

FIGURA 4

VALIDACIÓN CRUZADA



Fuente: Elaboración propia.

[6] Para facilitar la interpretación, el valor de [11] se ha relativizado respecto del que tomaría el árbol sin particiones.

Sin embargo, Breiman *et al.* (1984) sugieren que se debería elegir no el árbol que minimice [11] sino el que tenga el menor número de regiones necesarias para las que se alcance el umbral de una desviación típica por encima del mínimo. Siguiendo este método alternativo, el árbol que cruza el umbral que aparece como una línea horizontal discontinua en el gráfico es el que tiene ocho regiones. Si flexibilizamos un poco el umbral, también serían defendibles árboles con un número de regiones comprendido entre tres y siete. Sin embargo, es evidente que modificar el número de regiones cambiaría drásticamente las predicciones del árbol, como se ilustra en la figura 3.

Además, el precio que se paga por la simpleza del proceso de construcción de árboles de decisión que hemos descrito hasta ahora es la inestabilidad asociada a que un error cometido en las primeras escisiones de una rama provoca un efecto dominó en las ramas inferiores que amplifica el efecto del error en el árbol resultante. La inestabilidad da lugar a árboles poco robustos puesto que pequeños cambios en los datos producen grandes cambios en la estimación del árbol de decisión, lo que reduce la capacidad de interpretación de los resultados y dificulta su uso para realizar predicciones de la variable dependiente.

3. INESTABILIDAD Y TRATAMIENTO

El control de la inestabilidad debe ir encaminado a proporcionar métodos de estimación de árboles de decisión menos condicionados a pequeños cambios en los datos manteniendo controlado el sesgo de los árboles resultantes. La racionalidad que hay detrás de estos procedimientos es la de construir el árbol de decisión a partir de la combinación de otros árboles más sencillos estimados con los datos disponibles. En este contexto, los métodos más utilizados son los que se denominan bagging y random forest.

La principal diferencia entre estos dos métodos se encuentra en la forma en la que se diseñan los árboles sencillos a partir de los que se construyen los árboles complejos. En bagging (Breiman, 1996), y su versión mejorada random forest (Breiman, 2001), la construcción de los árboles sencillos se realiza en paralelo a partir de boostraps de la muestra de datos original. Sin embargo, en boosting (Freund y Schapire, 1997), la construcción de los árboles resultantes es secuencial, acumulando la aportación de árboles sencillos de tal manera que cada nuevo árbol pone más peso en las observaciones que han sido mal estimadas por el acumulado anterior.

Usando la descomposición del ECM propuesta en la expresión [8], en bagging, se emplean árboles más complicados, con muy poco sesgo pero mucha varianza. Sin embargo, agregando muchos de estos árboles se consigue reducir la varianza sin apenas introducir sesgo. Por el contrario, en boosting, se emplean agregaciones de árboles muy sencillos (en ocasiones, con una sola escisión) y muy poca varianza pero mucho sesgo. Sin embargo, el sesgo cometido por la agregación anterior se va ajustando secuencialmente con la aportación de las nuevas particiones.

3.1. Bagging y random forest

La primera contribución en este tipo de algoritmos se debe a Breiman (1996), quien propuso la metodología de árboles de decisión bootstrap agregados, conocida como bagging. Este procedimiento funciona entrenando en paralelo un gran número, B, de árboles de decisión formando un "bosque", en lugar de ajustar un único árbol. Cada uno de estos árboles tiene poco sesgo y mucha varianza. En cada nueva predicción, todos los árboles que forman el "bosque" participan aportando su predicción. Como valor final, se toma la media de todas las predicciones (variables continuas) o la clase más frecuente (variables cualitativas).

En el caso particular de los árboles, la técnica de bagging ha demostrado incrementar en gran medida la precisión de las predicciones. El algoritmo se puede resumir en los siguientes pasos:

- Generar B conjuntos de entrenamiento mediante bootstrapping con reemplazamiento a partir de la muestra original.

- Entrenar un árbol con cada una de las B muestras del paso 1. Cada árbol se crea sin apenas restricciones y no se somete a pruning, por lo que tiene varianza alta pero poco sesgo.

- Para cada nueva observación, obtener la predicción de cada uno de los B árboles. El valor final de la predicción se obtiene como la media de las B predicciones en el caso de árboles de regresión y como la clase predicha más frecuente (moda) para árboles de clasificación.

En el proceso de bagging, el número de árboles creados no es un hiperparámetro crítico en cuanto a que, por mucho que se incremente el número, no se aumenta el riesgo de sobreajuste. A pesar de ello, desde un punto de vista computacional no conviene almacenar el resultado de un gran número de árboles si, a partir de un número determinado, añadir nuevos árboles no aporta reducciones significativas del ECM. Una opción es obtener el número de árboles que optimiza el ECM usando técnicas de validación cruzada, aunque esta opción suele ser computacionalmente muy intensiva.

Una manera directa de estimar el error de un modelo al que se aplica bagging sin necesidad de aplicar la validación cruzada es mediante el método *out-of-bag* (OOB), propuesto originariamente por Breiman (1996). En concreto, el método consiste en entrenar cada árbol generado en el bagging en aproximadamente de las observaciones, mientras que el restante, conocidas como OOB, se usa para evaluar el árbol ajustado. Esta técnica es similar al *leave-one-out,* dando lugar a aproximadamente $B/3$ predicciones. La predicción OOB consiste en promediar los valores de los $B/3$ que no se

han considerado en su entrenamiento en el caso de árboles de regresión, o en considerar la moda en el caso de árboles de clasificación.

Con un número suficientemente elevado de árboles (B) el error OOB puede llegar a ser equivalente al error de validación leave-one-out. Además, el método basado en OOB para estimar el error resulta conveniente cuando se aplica bagging en grandes conjuntos de datos, para los que aplicar la validación cruzada sería computacionalmente muy costoso.

En el caso particular de los árboles de decisión, la técnica de bagging ha demostrado incrementar la precisión de las predicciones realizadas con los árboles resultantes de los algoritmos CART. Sin embargo, con esta técnica se pierde en interpretabilidad (representación gráfica) y se hace más difícil identificar la importancia de cada variable en la estimación del modelo. Una manera de determinar las variables que tienen mayor importancia es identificar aquel predictor con menor varianza residual total, en caso de árboles de regresión o índice de Gini para árboles de clasificación, como resultado de las divisiones sobre cada predictor, promediado sobre todos los B árboles. Un predictor importante será el que consiga una reducción promedio mayor.

Tal y como hemos expuesto con anterioridad el proceso de bagging se basa en el hecho de que, promediando un conjunto de modelos con bajo sesgo, se consigue reducir la varianza y aumentar la precisión de la estimación, manteniendo el sesgo. Pero esta reducción de la varianza depende del grado de correlación de los árboles que constituyen el bagging. Si la correlación es alta, la reducción de varianza que se puede lograr es pequeña. Por ejemplo, cuando un predictor es muy importante o influyente casi todos los árboles generados por bagging usarán este predictor en la primera ramificación, de manera que los árboles generados acabarán siendo similares unos a otros, sus predicciones estarán altamente correlacionadas, y su promedio no producirá una reducción sustancial de la varianza con respecto a la de un solo árbol.

Para solucionar este problema, se han propuesto variaciones al procedimiento bagging que tienen como objetivo reducir la correlacion, sin necesidad de aumentar substancialmente el sesgo. La variación del bagging más conocida es el random forest, originariamente desarrollado en Breiman (2001). Como en el caso de bagging, en random forest se entrenan los árboles en una muestra bootstrap. Sin embargo, en vez de seleccionar el conjunto completo de p predictores para realizar las escisiones en los nodos del árbol, se selecciona un subconjunto aleatorio de $q < p$ de estos predictores. Este proceso se repite para cada uno de los B árboles individuales de manera que el predictor random forest, al igual que en el caso de Bagging, se obtiene promediando las B predicciones individuales para árboles de regresión o seleccionando la clase más frecuente para árboles de clasificación.

3.2. Boosting

Aplicados a árboles de decisión, boosting es una idea relativamente simple. Consiste en la agregación secuencial de árboles sencillos, conocidos en la literatura relacionada como *weak learners,* asignando en cada iteración un mayor peso a los datos con mayor error en la estimación realizada mediante la agregación de weak learners hasta la iteración anterior. El resultado final, conocido como *strong learner,* es una agregación de árboles como una suma ponderada para el caso de árboles de regresión o como una mayoría cualificada en el caso de árboles de clasificación. El control de la varianza está asegurado usando como weak learners a árboles muy sencillos en cada iteración, mientras que el sesgo se controla con el mecanismo de ajuste a los datos mal estimados hasta la iteración anterior.

La idea que hay detrás de las técnicas de boosting fue originariamente diseñada para métodos de clasificación binarios en el influyente trabajo de Schapire (1990). El autor demuestra que el strong learner puede clasificar mejor que un weak learner si añadimos secuencialmente dos weak learners adicionales aplicados a muestras filtradas para que se enfoquen en las observaciones más difíciles de clasificar por los weak learners anteriores[7]. Freund (1995) extiende este algoritmo a clasificaciones no necesariamente binarias.

Freund y Schapire (1996) contribuyeron a popularizar el boosting como algoritmo de clasificación al proponer unas reglas muy simples para clasificar a partir del algoritmo conocido como *Adaptive Boosting (Adaboost).* Cuando el weak learner es un árbol de clasificación, el input del algoritmo son las T observaciones, $\{(Y_1,\boldsymbol{\mathcal{X}}_T),...,(Y_1,\boldsymbol{\mathcal{X}}_T)\}$, donde $Y_t \in \{1,...,K\}$ hace referencia a la clase que toma la variable dependiente en t. Al comienzo del algoritmo, todas las observaciones tienen el mismo peso $D_1(t) = 1/T$ para todo t. Para simplificar la exposición, presentaremos la versión del algoritmo Discrete Adaboost que propone Ng (2014) para una clasificación binaria donde $Y_t \in \{-1,1\}$.

El algoritmo *Discrete Adaboost* consiste en iterar J veces los siguientes tres pasos:

1. Aplicamos el árbol de decisión a las observaciones iniciales ponderadas por los pesos, de tal forma que se minimice el error $\varepsilon_j = \Sigma_{t=1,\mathcal{X}_t \in R^t}^{T} w_j(t) Y_t \hat{Y}^j(R^t)$, donde $\hat{Y}^j(R^t)$ es la estimación en el paso j-ésimo en la región R_t con $\boldsymbol{\mathcal{X}}_t \in R_m^t$. El algoritmo solo continua si $\varepsilon_j > 0{,}5$.

2. Calculamos la ponderación de la clasificación de la iteración j,$\alpha_j = 0{,}5\log((1-\varepsilon_j)/\varepsilon_j)$.

3. Recalculamos los pesos para la siguiente iteración $w_{j+1}(t) = w_j(t) exp\left(-\alpha_j Y_t \hat{Y}^j(R^t)\right)$. Los pesos son normalizados para que sumen 1.

[7] En árboles de clasificación, es frecuente usar como *wear learner* los stumps, que suponen una única partición a partir de una única variable explicativa.

La clasificación final es $\hat{Y}^J\left(R^t\right)=sign\left(\widehat{SY}^J\left(R^t\right)\right)$, donde $\widehat{SY}^J\left(R^t\right)=\Sigma_{j=1}^J\alpha_j\hat{Y}^j\left(R^t\right)$, con *sign(v)* = si $v>0$ y *sign(v)* = -1 si $v<0$[8]. Por tanto, podemos interpretar que la clasificación de una observación $\boldsymbol{x}_t$ se determinará según lo que decida la mayoría cualificada de clasificaciones en las iteraciones, con más peso aquellas iteraciones con menores errores.

La propuesta de Friedman *et al.* (2000) supone un avance significativo en los árboles de decisión estimados con boosting porque se establecen las bases estadísticas que permiten entender el buen ajuste de los algoritmos *Adaboost* a los datos y generalizar los algoritmos para incluir diversos escenarios estadísticos. Estos autores demuestran que los algoritmos se pueden interpretar como un problema de optimización de una función de pérdida y, usando funciones aditivas, se pueden resolver mediante algoritmos de Newton.

Por ejemplo, el algoritmo *Discrete Adaboost* se puede interpretar como la búsqueda del árbol de clasificación que minimiza la función de pérdida exponencial $E\left(exp\left(-Y_t\widehat{SY}^J\left(R^t\right)\right)\right)$. El clasificador que minimiza esta expresión es:

$$\widehat{SY}\left(R\right)\quad 0.5\log\frac{p\left(Y\quad\right)}{p\left(Y=-\right)}, \qquad [13]$$

que coincide con la mitad del *odds ratio* de las probabilidades. Por similitud con modelos logísticos, el algoritmo Adaboost consiste en estimar el clasificador como una aproximación no paramétrica basada en modelos de regresión aditiva $\widehat{SY}^J\left(R^t\right)=\Sigma_{j=1}^J\alpha_j\hat{Y}^j\left(R^t\right)$. Friedman *et al.* (2000) muestran que el clasificador óptimo es el que minimiza el error ponderado del algoritmo Adaboost. Además, las ponderaciones que minimizan la función de pérdida exponencial y la ponderación óptima, coinciden con las de dicho algoritmo.

La versión del algoritmo para el caso de los árboles de regresión implica la minimización de la función de pérdida cuadrática $E(\left(Y-\Sigma_{j=1}^J\alpha_j\hat{Y}^j\left(R,\Theta_j\right)\right)^2)$, donde hemos hecho explícito que los árboles que se usan como weak learners en la iteración *j* dependen de los parámetros de escisión Θ_j. El problema de optimización consiste en encontrar el clasificador final con la secuencia $\{\alpha_j,\Theta_j\}_{j=1}^J$ que minimiza la función de pérdida.

Resulta útil ver el problema anterior como un algoritmo de optimización secuencial en el que, dado el acumulado de los weak learners hasta la iteración *j*-1, $S\hat{Y}^{j-1}$, en la iteración *j* tenemos que encontrar $\{\alpha_j,\Theta_j\}$ que minimizan:

[8] Se puede demostrar que si el error cometido por los weak learners es mayor que 0.5, el error de la clasificación final disminuye exponencialmente con .

$$E\left[\left(Y-\Sigma_{j=1}^{J-1}\alpha_i\hat{Y}^i\left(R,\Theta_j\right)-\alpha_j\hat{Y}^j\left(R,\Theta_j\right)\right)^2\right]. \qquad [14]$$

En muestras finitas, el objetivo en cada iteración es encontrar el árbol de decisión que minimiza la suma de los cuadrados de los errores que se han cometido hasta la iteración anterior[9]. Friedman *et al.* (2000) proponen un algoritmo de Newton secuencial para encontrar el óptimo de dicha función y sientan las bases del algoritmo *Gradient Boosting*.

El algoritmo Gradient Bossting, propuesto por Friedman (2001), usa el algoritmo de optimización numérica *Functional Gradient Descent* para encontrar iterativamente el mínimo de la función de pérdida. En cada iteración, $j \in \{1,...,J\}$ se busca la dirección y el tamaño de actualización encaminada a optimizar la reducción del valor de la función de pérdida. En concreto, en cada iteración, la dirección que más reduce la función de pérdida es la opuesta al gradiente de la función de pérdida. Por esa razón, buscaremos el weak learner que más se aproxima al negativo del gradiente. El tamaño de la actualización, que llamaremos c_j, será el que optimiza la caída de la función de pérdida en la dirección seleccionada.

Si definimos a la función de pérdida como $L\left(Y,\widehat{SY}(R)\right)$, el algoritmo Gradient Bossting es el resultado de la iteración de los siguientes pasos:

1. Inicializamos el algoritmo $\widehat{SY}^0 = \arg\min_{SY} \Sigma_{t=1}^{T} L\left(Y_t, SY\right)2$. Dado un número máximo de iteraciones Z, para cada $z = 1,2,...,Z$, buscamos la dirección óptima para minimizar la función de pérdida, que coincide con el negativo del gradiente de la función de pérdida.

   - Para cada $t = 1,2,...,T$ calcular

$$r_z^t = -\left[\frac{\partial L\left(Y_t,\widehat{SY}\left(R^t\right)\right)}{\partial \widehat{SY}\left(R^t\right)}\right]_{\widehat{SY}\left(R^t\right)=\widehat{SY}^{z-1}\left(R^t\right)} \qquad [15]$$

   A r_z^t se le conoce como pseudoresiduo.

   - Determinar el árbol de decisión que determina $\mathcal{X}$ que mejor se ajusta a r_z^t proporcionando A_i^t regiones terminales con $i = 1,2,...,K_z$.

[9] Para determinados casos, resulta útil establecer otras funciones de pérdida. Por ejemplo, la función de Huber es recomendable cuando para el tratamiento de bases de datos con atípicos.

- Para cada $i = 1,2,...,K_t$ calcular, $c_i^t = \arg\min_{SY}\left\{\Sigma_{X_t \in A_i^t} L\left(Y_t, \widehat{SY}^{z-1}\left(A_i^t\right)\right) + SY\right\}$.

- Actualizar $SY^z(R^t) = SY^{z-1}(R^t) + \Sigma_{i=1, x_t \in R^t \cap A_i^t} SY(A_i^t)$

3. Por último, el estimador es $\widehat{SY} = SY^{K_z}$.

Friedman (2001), incorpora un parámetro de ajuste que controla el tamaño de los saltos que se dan en el algoritmo en cada iteración, por lo que la actualización del strong learner se produce escalada por un parámetro μ*:*

$$\widehat{SY}^j\left(R^t\right) = \Sigma_{i=1}^{j-1}\widehat{SY}^i\left(R^t\right) + \mu\widehat{SY}^j\left(R^t\right), \qquad [16]$$

donde $0 < \mu \leq 1$. Cuanto más pequeño sea el valor del parámetro de ajuste mejor será la clasificación en el período de entrenamiento, pero mayor será el número de iteraciones necesarias para alcanzar el óptimo. Por otro lado, valores muy grandes pueden subestimar el número de árboles necesarios. Los resultados empíricos de Friedman (2001) sugieren usar valores $\mu \leq 0{,}1$.

Finalmente, Friedman (2002) incorpora un elemento adicional en el algoritmo *Stochastic Gradient Boosting:* el muestreo aleatorio de un porcentaje λ de observaciones que forman parte del periodo de entrenamiento. En concreto, en cada iteración del algoritmo, el nuevo árbol de decisión se ajusta empleando únicamente una fracción λ de datos del periodo de entrenamiento, extraída de forma aleatoria y sin reemplazo. Este procedimiento mejora el ajuste a los datos y agiliza la computación. En la práctica, para reducir tiempo de computación, se recomienda que λ sea menor cuanto mayor sea el número de variables explicativas disponibles.

Con el objetivo de que los árboles de clasificación se puedan usar para seleccionar de manera automática las variables explicativas más influyentes a la hora de formar los árboles de decisión, Friedman (2001) propone una medida de la importancia relativa en el algoritmo boosting. Siendo IR_p^j la importancia relativa de la variable p en el árbol que se estima en la iteración j, la importancia relativa de esa variable en el boosting será la media de su importancia relativa en los J árboles generados, $IR_p^{Bo} = \frac{1}{J}\Sigma_{i=1}^{j} IR_p^j$

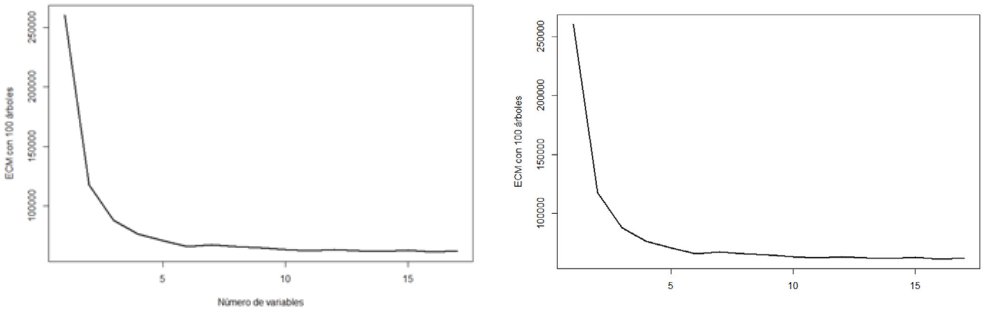
3.3. Ejemplo de bagging y boosting

En esta sección vamos a examinar cómo se ajustan las tres propuestas anteriores, bagging, random forest y boosting a los datos de los precios de la vivienda en Madrid. Si comenzamos con la estimación bagging, tenemos que elegir el número de árboles que se van a muestrear con bootstrap. El gráfico izquierdo de la figura 5 muestra el

ECM que se alcanza en función del número de árboles, con un máximo de 500 árboles. Como se puede observar, a partir de unos 100 árboles apenas se alcanzan reducciones significativas en el error. Por tanto, usaremos bagging con un número de boostraps de 100 árboles.

FIGURA 5

HIPERPARÁMETROS DE BAGGING Y RANDOM FOREST

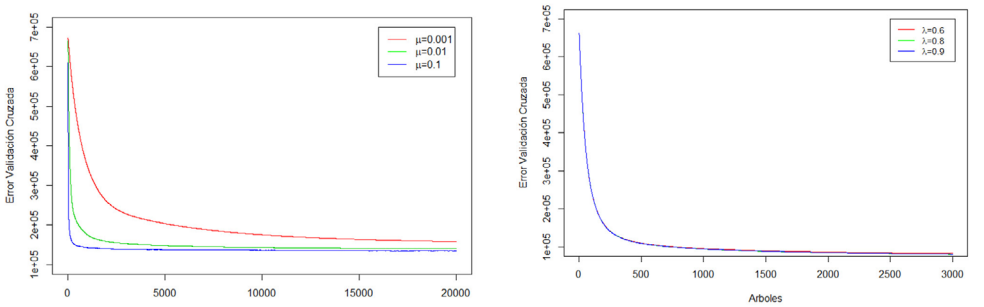


Fuente: Elaboración propia.

Para ilustrar las ventajas de random forest frente a bagging, el gráfico derecho de la figura 5 muestra cómo evoluciona el ECM con 100 árboles que se obtiene con un número de variables seleccionadas en cada muestreo que varía desde 1 a 17, que es el número total de variables explicativas de la muestra. Como podemos observar en la figura, a partir de 5 variables para cada muestreo, el error el árbol de regresión no es capaz de reducir el ECM de manera significativa.

FIGURA 6

HIPERPARÁMETROS DE BOOSTING



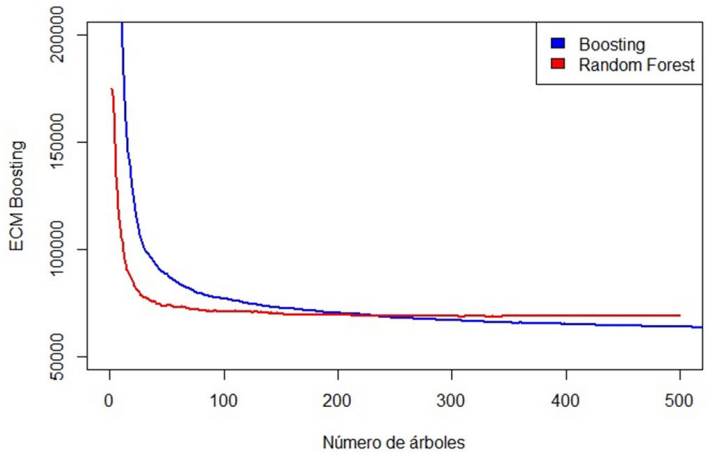
Fuente: Elaboración propia.

La figura 6 nos va a ayudar a optimizar los hiperparámetros que se usan para aplicar el algoritmo boosting. El gráfico de la izquierda muestra la evolución del ECM en función del número de árboles Z para distintos valores del parámetro de ajuste $\mu = 0{,}001, 0{,}01, 0{,}1$. En este gráfico podemos observar que el ECM se estabiliza para un número de árboles alrededor de 200 y que valores del parámetro de ajuste entre y no afectan mucho a la capacidad de ajuste a los datos del modelo. El gráfico de la derecha muestra que el ajuste del método Stochastic Gradient Boosting no depende mucho de la fracción de datos que usan para el entrenamiento, para valores de λ entre 0,6 y 0,9.

En la literatura de los árboles de decisión, la elección entre random forest o boosting ha resultado ser una cuestión empírica porque no hay un ganador absoluto entre ellos. Hamza y Larocque (2005) encuentran que random forest funcionaba mejor que boosting, aunque Gashler, Giraud-Carrier Martínez (2008) muestran que random forest no funciona peor cuantas más variables explicativas irrelevantes contenga la muestra de datos. En nuestro ejemplo, la figura 7 muestra que ninguna de estas dos opciones aparezca como una ganadora clara. En términos generales, El ECM de ambos algoritmos es muy parecido. El error que comete el bagging es un poco más reducido que el de boosting para un número de árboles inferior a 200. Sin embargo, a partir de 200 árboles, el algoritmo boosting se ajusta un poco más a los datos que el algoritmo random forest.

FIGURA 7

ECM DE RANDOM FOREST Y BOOSTING



Fuente: Elaboración propia.

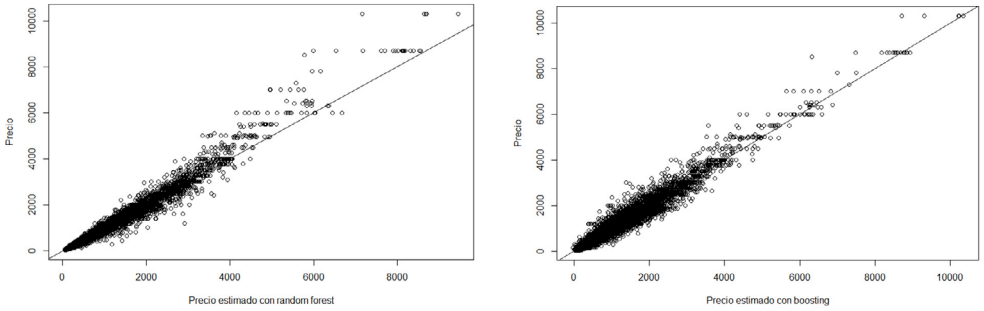
Una diferencia importante de los árboles de decisión estimados con random forest o con boosting respecto de los árboles estimados con CART es la formación de predicciones de la variable dependiente. La figura 8 muestra que la predicción obtenida con estos dos algoritmos no es tan particionada como la que se realiza con CART, como

mostraba el gráfico izquierdo de la figura 3. La forma de construcción del árbol con random forest y bagging, donde la inferencia de cada observación es una media de la inferencia realizada con cada uno de los árboles que forman parte de esos algoritmos hace que la inferencia final sea más parecida a la que estamos acostumbrados a obtener a partir de un modelo de estimación tradicional, como la predicción con MCO que aparece en el gráfico derecho de la figura 3.

Para comparar la bondad del ajuste de todos los modelos propuestos, hemos calculado el pseudo R^2 para estos dos algoritmos. En el caso de random forest el pseudo R^2 es de 0,97, mientras que en el caso de boosting el pseudo R^2 es de 0,96. Estos valores son claramente superiores a los que se alcanzaban con árboles estimados mediante CART (0,73) y con MCO (0,72). Sin embargo, debemos señalar que la habilidad de un modelo para ajustarse a los datos dentro de la muestra no necesariamente se tiene que corresponder con la capacidad predictiva del modelo.

FIGURA 8

PREDICCIÓN DEL PRECIO DE LA VIVIENDA CON RANDOM FOREST Y BOOSTING



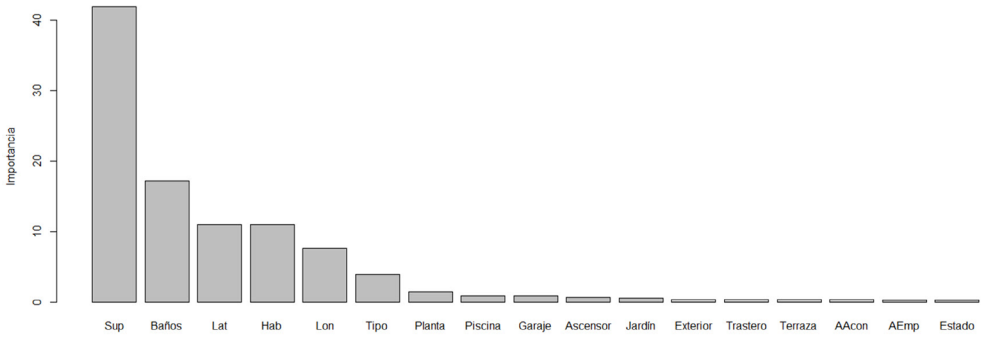
Fuente: Elaboración propia.

Para examinar la capacidad predictiva de los modelos, hemos dividido aleatoriamente la muestra en un dos grupos de datos. El grupo de entrenamiento supone el 75% de los datos y se ha usado para estimar los modelos. El grupo de evaluación lo forman el resto de datos y se usa para predecir con los modelos estimados en el grupo de entrenamiento. El *(out-of-sample)* pseudo R^2 que alcanzan los modelos MCO, CART, random forest y boosting es de 0,71, 0,72, 0,89 y 0,90. Como se esperaba, todos los modelos muestran un deterioro en la capacidad predictiva. Sin embargo, los modelos random forest y boosting siguen apareciendo como los modelos con mayor capacidad predictiva. Como los resultados que obtenemos con los algoritmos random forest y booting son muy parecidos, para simplificar el análisis nos centraremos en los resultados obtenidos con random forest.

La figura 9 muestra la importancia relativa de las variables que determinan el precio de la vivienda calculada a partir del método random forest[10]. La figura revela que la principal variable para determinar el precio de la vivienda es la superficie, seguida del número de baños. Posiblemente, estas dos variables estén correlacionadas y reflejen, junto con el número de habitaciones, la relación entre el precio y la dimensión de la vivienda. Las variables que determinan la posición de la vivienda también ocupan un papel relevante en la determinación del precio de la vivienda. La última variable importante es el tipo de vivienda, mientras que el resto de variables parecen tener una importancia relativa mucho menor.

FIGURA 8

IMPORTANCIA RELATIVA



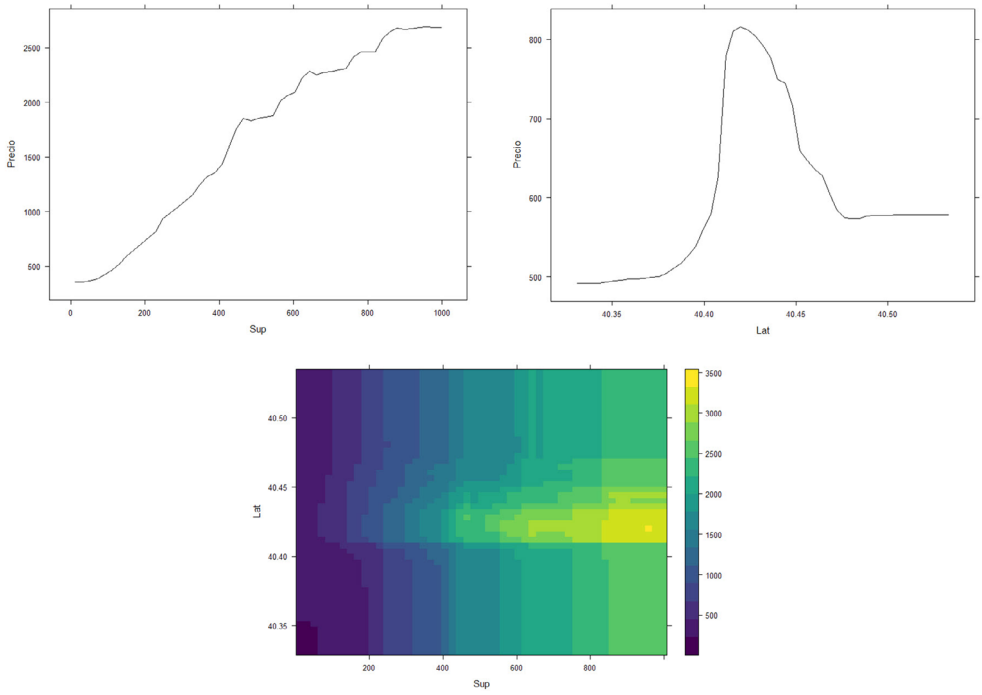
Fuente: Elaboración propia.

Una de las ventajas más significativas de los árboles de decisión estimados con random forest y boosting frente a modelos tradicionales como CART es su mayor capacidad de capturar relaciones no lineales entre las variables explicativas y la variable dependiente. La figura 10 muestra claramente que el efecto de la latitud en el precio de la vivienda es no lineal. Hasta una latitud de 40.41 el efecto sobre el precio es positivo, es decir, las viviendas del sur aumentan de precio conforme las viviendas se sitúan más al norte. Hasta una latitud de 40.43, donde se encuentran las viviendas del centro la latitud deja de tener importancia. Finalmente, para las viviendas situadas al norte de la latitud 40.45, la relación con el precio se invierte, siendo las viviendas más baratas conforme las viviendas se sitúan del centro hacia el norte. Para las viviendas muy en el sur o muy en el norte, la latitud no tiene efecto en el precio. La superficie, sin embargo, tiene una relación positiva y lineal con el precio de la vivienda. El gráfico inferior de la figura 10 muestra un efecto interacción no lineal entre esas dos variables.

[10] Cualitativamente, la importancia relativa que se obtiene con boosting es similar.

FIGURA 10

GRÁFICOS DE DEPENDENCIA PARCIAL: BOOSTING



Fuente: Elaboración propia.

En último lugar vamos a ilustrar la ventaja de los árboles de decisión sobre los métodos de estimación clásicos como MCO usando la figura 11. Si recordamos la figura 1, parece evidente que la relación entre la posición geográfica, medida a través de la latitud y la longitud de la vivienda, y el precio de la misma dista mucho de ser lineal. Sin embargo, si tratamos de estimar un modelo lineal entre el precio de la vivienda y sus características mediante MCO, tanto la latitud como la longitud aparecen con un coeficiente positivo. Eso explica que el efecto interacción que se muestra en el gráfico izquierdo de la figura 11, las viviendas con más valor aparezcan erróneamente en el noreste geográfico.

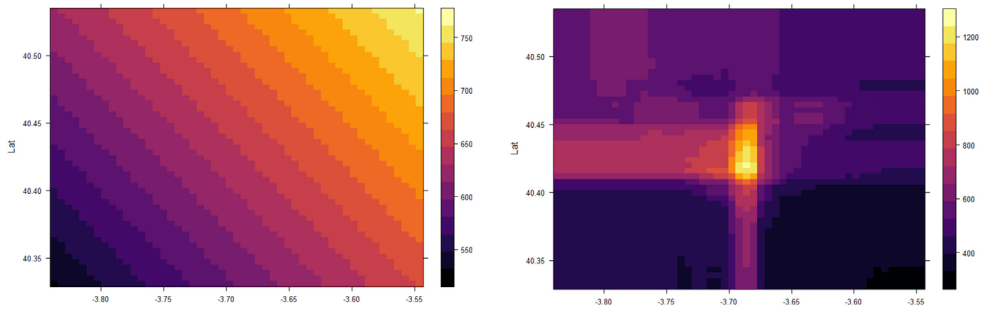
Sin embargo, el efecto interacción del gráfico derecho, que se ha obtenido a partir de un árbol de decisión estimado con el algoritmo random forest, muestra una relación claramente no lineal entre el precio de la vivienda y su posición geográfica. Las viviendas más caras se sitúan en el centro de Madrid. Hacia el sur, no aparece una relación evidente entre el precio de la vivienda y su posición geográfica excepto entre las que se encuentran en una longitud central, referida a las viviendas más caras. Para esas

viviendas, el precio cae cuando se sitúan más al sur. Las casas más caras se sitúan al norte de la latitud 40,41, especialmente las de latitud más central (entre -3,70 y -3,80). Al norte de la latitud 40,41, también ocurre que las viviendas son más baratas cuando nos alejamos del centro.

FIGURA 11

POSICIÓN GEOGRÁFICA Y PRECIO DE LA VIVIENDA

[MCO] [Arbol de decisión]



Fuente: Elaboración propia.

3.4. Extensiones

Una característica de los árboles de regresión, tal y como son normalmente implementados, es que las particiones del espacio de definición de las variables explicativas, se hacen de manera ortogonal a la variable de escisión, generando una partición formada por prismas rectangulares p-dimensionales. Esta forma de particionar está justificada por la reducción del espacio de búsqueda del algoritmo, lo que se traduce en ganancias computacionales. Sin embargo, las particiones ortogonales pueden fallar a la hora de encontrar particiones apropiadas en algunos casos, haciendo que el comportamiento del algoritmo sea mediocre.

Con el fin de paliar este problema, han aparecido en la literatura un conjunto de propuestas de partición alternativas. Una de las más populares está basada en la realización de particiones con fronteras oblicuas, como las que se proponen en los trabajos de Murthy, Kasif, y Salzberg (1994), Wickramarachchi *et al.* (2016) y Cantu-Paz y Kamath (2003). Además de la complejidad computacional, unos de los problemas de estas propuestas es que las realización de las particiones suele depender de diversos parámetros que deben ser elegidos por los usuarios, limitando la robustez de los resultados.

Otra opción es la que han propuesto Paez *et al.* (2019), quienes introdujeron una novedosa forma de generar particiones no ortogonales en el espacio de definición de las variables, produciendo un incremento en la complejidad en la construcción del árbol bastante modesto. En concreto, esta aproximación se basa en la aplicación de funcio-

nes base interactivas, lo que permite particionar con fronteras de cualquier forma funcional, manteniendo como caso particular las particiones oblicuas. La principal ventaja de esta técnica es su fácil implementación ya que permite seguir usando los mismos códigos que se usan tradicionalmente para particiones ortogonales.

La segunda extensión que vamos a considerar en esta sección es el tratamiento de datos perdidos *(missing data),* que resulta muy habitual en el análisis de datos económicos. Como se refleja en Tawala (2009), la forma en la que se ha resuelto este problema en el contexto de árboles de decisión es muy diversa. Una opción es eliminar de la base de datos las variables con datos perdidos o recortar la muestra para que todos los datos sean observados en todas las variables. Otra opción es imputar los datos que faltan con la media, la mediana o la moda de la variable explicativa correspondiente. También se han propuesto formas de imputación más complejas como las que se basan en algoritmos *Expectation Maximization (EM).*

Una de las alternativas a la imputación más habituales en algoritmos CART consiste en usar puntos de escisión *(splits)* subrogados. Supongamos que, en un determinado nodo, el árbol tiene que elegir entre la región izquierda y la derecha usando la variable de escisión X_p que no se observa en un elemento de la muestra ($X_{pt} = NA$). Para implementar esta medida, el primer paso consiste en estimar el árbol de decisión con el subconjunto completo de observaciones que excluye a t. En el segundo paso, se elige una variable (X_j) y un punto de escisión (s) que proporcionan la partición más parecida a la que se obtiene con X_p para el nodo determinado. Por último, para la observación t se realiza la partición usando la variable subrogada X_j.

4. OTRAS APLICACIONES ECONÓMICAS Y EMPRESARIALES

Aunque los primeros métodos de construcción de árboles de decisión aparecen en los años sesenta, la aplicación de estas técnicas a problemas económicos y empresariales es mucho más reciente. Sin ánimo de ser exhaustivos, en esta sección vamos a describir algunas de las aplicaciones más interesantes de estas técnicas para resolver problemas relacionados con economía y empresa.

Entre las aplicaciones de árboles de clasificación a la empresa, destacamos la que realizaron Qabbaah, Sammou y Vanhoof (2019) para mejorar la eficiencia de campañas de marketing realizadas a través de correo electrónico. Tirenni, Kaiser y Herrmann (2007) usan árboles de decisión para analizar la segmentación de los consumidores en empresas de aerolíneas. Tras una comparativa con otros modelos de clasificación, Gepp, Kumar, y Bhattacharya (2010) concluyen que los árboles de decisión son la mejor herramienta para predecir el fracaso empresarial. Xie *et al.* (2009) desarrollan una extensión de los árboles de decisión para anticipar el abandono de clientes de entidades financieras.

Relacionados con economía de la educación, Zeng *et al.* (2014) han aplicado técnicas de árboles de regresión para examinar los condicionantes de la elección de universidades en China. Mythili y Shanavas (2014) realizan una comparativa entre diversos algoritmos de clasificación para anticipar la mejora en los resultados de los estudiantes a partir de un conjunto de variables socio-económicas. Entre los métodos evaluados, destacan árboles estimados con random forest .

La aplicación de árboles de clasificación al análisis de ciclos económicos se inicia con el trabajo de Ng (2014), quien usa técnicas de boosting con stumps como weak learners para evaluar la capacidad de un elevado número de indicadores económicos para predecir las recesiones de EE. UU. con 3, 6, y 12 meses de adelanto. Döpke, Fritsche, y Pierdzioch, (2017) introducen como weak learners árboles más complejos en el algoritmo de boosting para examinar el efecto interacción de los indicadores económicos en la predicción de las recesiones en Alemania. Piger (2020) compara la habilidad de los árboles de decisión construidos con random forest y boosting junto con otras técnicas de machine learning y modelos *Markov-switching* para anticipar las recesiones en EE. UU. En sus resultados, Piger (2020) destaca la habilidad del algoritmo boosting sobre el resto de métodos evaluados. En relación con estos trabajos, Ward (2017) utiliza el algoritmo random forest para identificar episodios de crisis financieras en un conjunto amplio de países.

Rossi y Timmermann (2015) proponen un nuevo procedimiento para medir el riesgo en un modelo intertemporal de fijación de precios de los activos de capital (*ICAPM,* por sus siglas en inglés). Los árboles de regresión construidos con boosting se usan para realizar predicciones no lineales de las matrices de covarianza. En una comparativa entre cuatro metodologías diferentes, Liu *et al.* (2015) encuentran que los árboles de decisión construidos mediante random forest son la mejor opción para anticipar el fraude financiero, siendo la relación deuda-patrimonio (*debt-to-equity ratio*) la variable más significativa.

5. CONCLUSIONES Y FUTURAS LÍNEAS DE INVESTIGACIÓN

El principal propósito de este trabajo es el de contribuir en la difusión de la metodología de árboles de decisión como una técnica de análisis de problemas relacionados con economía y empresa. Consideramos que esta herramienta es especialmente útil para resolver dos problemas a los que se enfrentan actualmente las técnicas de análisis tradicionales. El primer problema tiene que ver con el manejo de datos de gran volumen y variedad, conteniendo variables cuantitativas y cualitativas. El segundo problema es la detección de relaciones complejas entre las variables del modelo, difíciles de capturar con modelos lineales y/o paramétricos. Estamos convencidos que la versatilidad de los árboles de decisión puede ser especialmente útil en estos dos casos.

Para conseguir este objetivo, en el manuscrito hacemos una descripción autocontenida de la metodología de árboles de decisión, empezando por la construcción de

un árbol de decisión sencillo mediante el algoritmo CART. Seguidamente, mostramos las extensiones más habituales, entre las que destacan las que se basan en algoritmos random forest y boosting. Además, dedicamos una sección al tratamiento de extensiones menos habituales como las que producen particiones no ortogonales. Finalmente, dedicamos otra sección a mostrar numerosos ejemplos para ilustrar cómo los árboles de decisión han permitido esclarecer algunas relaciones entre variables relacionadas con la economía y la empresa.

Para facilitar el acercamiento de los economistas a estas técnicas, proponemos un ejemplo enfocado a examinar la determinación del precio de la vivienda en el municipio de Madrid. La base de datos, extraídos del portal de Idealista en 2018, contiene el precio de 20.385 viviendas, junto con un conjunto de potenciales variables explicativas. Entre ellas, las que destacan por su importancia final en la determinación del precio de la vivienda son la superficie, el número de baños y de habitaciones, la posición geográfica, y el tipo de vivienda. Usando el Pseudo-R^2, encontramos que la capacidad de los árboles de decisión para predecir el precio de la vivienda basados en los algoritmos random forest y boosting es similar, aunque muy superior a la árboles basados en CART y a la de mínimos cuadrados ordinarios. Además, mostramos cómo los árboles son capaces de capturar relaciones no lineales complejas entre, por ejemplo, la posición geográfica y el precio de la vivienda que las técnicas tradicionales serían incapaces de determinar.

Con el fin último de simplificar la adopción de estas técnicas en economía y empresa, hemos puesto a disposición de los lectores en la página web de los autores tanto la base de datos como un código en R que replica los principales resultados que se presentan en este trabajo. Confiamos en que este material sirva como primer paso de muchas aplicaciones en el contexto de la economía y la empresa en el futuro.

Referencias

Austin, P. C., Lee, D. S., Steyerberg, E. W. y Tu, J. V. (2012). Regression trees for predicting mortality in patients with cardiovascular disease: What improvement is achieved by using ensemble-based methods? *Biometrical journal,* 54(5), pp. 657-673.

Austin, P. C., Tu, J. V., Ho, J. E., Levy, D. y Lee, D. S. (2013). Using methods from the data-mining and machine-learning literature for disease classification and prediction: a case study examining classification of heart failure subtypes. *Journal of clinical epidemiology,* 66(4), pp. 398-407.

Breiman, L., Friedman, J., Stone, C. J. y Olshen, R. A. (1984). *Classification and regression trees.* New York: Chapman and Hall.

Cantu-Paz, E. y Kamath, C. (2003). Inducing oblique decision trees with evolutionary agorithms. *Transactions on Evolutionary Computation,* 7(1), pp. 54-68.

Choubin, B., Darabi, H., Rahmati, O., Sajedi-Hosseini, F. y Klove, B. (2018). River suspended sediment modelling using the Cart model: A comparative study of machine learning techniques. *Science of the Total Environment*, 615, pp. 272-281.

De'ath, G. y Fabricius, K. E. (2000). Classification and regression trees: a powerful yet simple technique for ecological data analysis. *Ecology,* 81(11), pp. 3178-3192.

Diebold, F. X. (2012). On the origin(s) and development of the term Big Data. *Documento de Trabajo PIER,* 12-037

Döpke, J., Fritsche, U. y Pierdzioch, C. (2017). Predicting recessions with boosted regression trees. *International Journal of Forecasting,* 33(4), pp. 745-759.

Fan, G., Ong, S. y *Koh, H.* (2006). Determinants of house price: A decision tree approach. *Urban Studies,* 43(12), pp. 2301-2316.

Gashler, M., Giraud-Carrier, C. y Martínez, T. (2008). Decision tree ensemble: small heterogeneous is better than large homogeneous. *Proceesing of the 7th International Conference on Machine Learning and Applications,* pp. 900-905.

Ghasri, M., Rashidi, T. y Waller, S. (2017). Developing a disaggregate travel demand system of models using data mining techniques. *Transportation Research Part a-Policy and Practice,* 105, pp. 138–153.

Gepp, A., Kumar, K. y Bhattacharya, S. (2010). Business failure prediction using decision trees. *Journal of forecasting,* 29(6), pp. 536-555.

Hamza, M. y Larocque, D. (2005). An empirical comparison of ensemble methods based on classification trees. *Journal of Statistical Computation and Simulation,* 75, pp. 629–643.

James, G., Witten, D., Hastie, T. J. y Tibshirani, R. J. (2013). *An introduction to statistical learning with applications in R.* New York: Springer-Verlag.

Kitsantas P., Moore T. y Sly D. (2007). Using classification trees to profile adolescent smoking behaviors. *Adictive Behaviors,* 32(1), pp. 9–23.

Kurt, I., Ture, M. y Kurum, A. T. (2008). Comparing performances of logistic regression, classification and regression tree, and neural networks for predicting coronary artery disease. *Expert Systems with Applications,* 34 (1), pp. 366-74.

Liu, C., Chan, Y., Kazmi, S. y Fu, H. (2015). Financial fraud detection model: Based on random forest. *International Journal of Economics and Finance,* 7(7), pp. 178-188.

Loh, W. Y. (2011). Classification and regression trees. *Wiley Interdisciplinary Reviews – Data Mining and Knowledge Discovery,* 1(1), pp. 14-23.

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability,* 1, pp. 281-297.

Morgan, J. y Sonquist, J. A. (1963). Problems in the analysis of survey data, and a proposal. *Journal of the American Statistical Assocciation,* 58, pp. 415–434.

Murthy, S., Kasif, S. y Salzberg, S. (1994). A System for Induction of Oblique Decision Trees. *Journal of Artificial Intelligence Research,* 2, pp. 1-32.

Mythili, M. y Shanavas, A. (2014). An Analysis of students' performance using classification algorithms. *IOSR Journal of Computer Engineering,* 16(1), pp. 63-69.

Ng, S. (2014). Boosting recessions. *Canadian Journal of Economics,* 47, pp. 1-34.

Páez, A., López, F., Ruiz, M. y Camacho, M. (2019). Inducing non-orthogonal and non-linear decision boundaries in decision trees via interactive basis functions. *Expert Systems with Applications,* 122, pp. 183-206.

Piger, J. (2020). Turning points and classification. En: P. Fuleky (ed.), *Macroeconomic forecasting in the era of big data: Theory and application.* Springer International Publishing.

Qabbaah, K., Sammour, G. y Vanhoof K. (2019). Decision tree analysis to improve e-mail marketing campaigns. International Journal of Information *Theories and Applications,* 26(1), pp. 3-36.

Rossi, A. y Timmermann, A. (2015). Modeling covariance risk in Merton's ICAPM. *Review of Financial Studies,* 28(5), pp. 1428–1461.

Schapire, R. (1990). The strength of weak learnability. *Machine Learning,* 5, pp. 197–227.

Twala, B. (2009). An empirical comparison of techniques for handling incomplete data when using decision trees. *Applied Artificial Intelligence,* 23, pp. 373–405.

Tirenni, G., Kaiser, Ch. y Herrmann, A. (2007). Applying decision trees for value-based customer relations management: Predicting airline customers' future values. *Journal of Database Marketing and Customer Strategy Management,* 14, pp. 130–142.

Venkata, S. y Kiruthika, P. An overview of classification algorithm in data mining. International *Journal of Advanced Research in Computer and Communication Engineering,* 4(12), pp. 255-257.

Ward, F. (2017). Spotting the danger zone: Forecasting financial crises with classification tree ensembles and many predictors. *Journal of Applied Econometrics,* 32(2), pp. 359-378.

Wickramarachchi, D., Robertson, B., Reale, M., Price, C. y Brown, J. (2016). HHCART: An oblique decision tree. *Computational Statistics and Data Analysis,* 96, pp. 12-23.

Xie, Y., Li, X., Ngai, E. W. T. y Ying, W. (2009). Customer churn prediction using improved balanced random forests. *Expert Systems with Applications,* 36(3), pp. 5445-5449.

Yang, L. J., Liu, S., Tsoka, S. y Papageorgiou. L. G. (2017). A regression tree approach using mathematical programming. *Expert Systems with Applications,* 78, pp. 347-57.

Zeng, X., Yuan, S., Li, Y. y Zou, Q. (2014). Decision tree classification model for ppularity forecast of Chinese colleges. *Journal of Applied Mathematics,* Article ID 675806.