Programa de Doctorado en Estadística, Optimización
y Matemática Aplicada

Una nueva metodología basada en Gradient Boosting para la estimación de fronteras de mejores prácticas

Tesis Doctoral

María Dolores Guillén García

Director

Dr. Juan Aparicio Baeza

Universidad Miguel Hernández de Elche

2024

La presente Tesis Doctoral, titulada **"Una nueva metodología basada en Gradient Boosting para la estimación de fronteras de mejores prácticas"**, se presenta bajo la modalidad de **tesis por compendio** de las siguientes **publicaciones**:

A **Gradient tree boosting and the estimation of production frontiers**. Guillen, M. D., Aparicio, J., & Esteve, M. (2023). *Expert Systems with Applications*, 214, 119134. https://doi.org/10.1016/j.eswa.2022.119134

B **Performance Evaluation of Decision-Making Units Through Boosting Methods in the Context of Free Disposal Hull: Some Exact and Heuristic Algorithms**. Guillen, M. D., Aparicio, J., & Esteve, M. (2023). *International Journal of Information Technology & Decision Making*, 1-30. https://doi.org/10.1142/S0219622023500050

C **boostingDEA: A boosting approach to Data Envelopment Analysis in R**. Guillen, M. D., Aparicio, J., & España, V. J. (2023). *SoftwareX*, 24, 101549. https://doi.org/10.1016/j.softx.2023.101549

UNIVERSITAS
Miguel Hernández

El Dr. D. Juan Aparicio Baeza director, director de la tesis doctoral titulada "**Una nueva metodología basada en Gradient Boosting para la estimación de fronteras de mejores prácticas"**

INFORMA:

Que Dña. María Dolores Guillén García ha realizado bajo nuestra supervisión el trabajo titulado **"Una nueva metodología basada en Gradient Boosting para la estimación de fronteras de mejores prácticas"** conforme a los términos y condiciones definidos en su Plan de Investigación y de acuerdo al Código de Buenas Prácticas de la Universidad Miguel Hernández de Elche, cumpliendo los objetivos previstos de forma satisfactoria para su defensa pública como tesis doctoral.

Lo que firmo para los efectos oportunos, en Elche en julio de 2024.

Director de la tesis

Dr. D. Juan Aparicio Baeza

El Dr. D. Domingo Morales González, Coordinador del Programa de Doctorado en **Estadística, Optimización y Matemática Aplicada**

INFORMA:

Que Dña. María Dolores Guillén García ha realizado bajo la supervisión de nuestro Programa de Doctorado el trabajo titulado "**Una nueva metodología basada en Gradient Boosting para la estimación de fronteras de mejores prácticas"** conforme a los términos y condiciones definidos en su Plan de Investigación y de acuerdo al Código de Buenas Prácticas de la Universidad Miguel Hernández de Elche, cumpliendo los objetivos previstos de forma satisfactoria para su defensa pública como tesis doctoral.

Lo que firmo para los efectos oportunos, en Elche en julio de 2024.

Prof. Dr. D. Domingo Morales González

Coordinador/a del Programa de Doctorado en
Estadística, Optimización y Matemática Aplicada

La presente **Tesis Doctoral** ha sido **financiada** por:

- Subvención para la contratación de personal investigador predoctoral CIACIF/2021/345, financiada por la Conselleria de Innovación, Universidades, Ciencia y Sociedad Digital (Generalitat Valenciana).
- Proyecto PROMETEO 2021/063 financiado por la Conselleria de Educación, Universidades y Empleo (Generalitat Valenciana).
- Proyecto de I+D+i PID2019-105952GB-I00, financiado por el Ministerio de Ciencia e Innovación/Agencia Estatal de Investigación/ 10.13039/501100011033.
- Proyecto de I+D+i PID2022-136383NB-I00, financiado por MICIU/AEI /10.13039/501100011033 y por FEDER, UE

Agradecimientos

A Juan, por animarme a hacer el doctorado, por confiar en todo momento y por siempre buscar lo mejor para mí. Esta Tesis ha sido posible gracias a ti.

A mis compañeras y compañeros de la sala de becarios, por nuestros almuerzos en el comedor, las excursiones y las fiestas de pueblos alicantinos. Sin vosotros, habría perdido la cordura.

A las investigadoras, los investigadores y al personal mantenimiento y servicios del CIO, por hacer de él un sitio especial. En particular, gracias a mi equipo de madrugadores y a Manu, por los cafés (y el té) y por enseñarme cómo ser una buena docente.

A mis amigas y amigos (los de toda la vida, los que conocí en la universidad y las que me acogieron como una más), por interesarse en el desarrollo de esta Tesis, aunque no entendieran sobre qué iba.

A mi madre, mi padre y mi hermano, por quererme incondicionalmente y por dejarme elegir qué hacer en cada momento. Al resto de mi familia, por apoyarme y animarme en cada paso.

A MJ, por recorrer este camino junto a mí y por tener la certeza de que estarás a mi lado para recorrer todo lo que venga. Y a Cleo, *miau miau*.

Resumen

Dentro de los campos de la econometría y la ingeniería de producción, un tema de interés es la evaluación de la eficiencia técnica de entidades a partir de la estimación de la frontera de mejores prácticas, la cual delimita el conjunto de posibilidades de producción o tecnología. Por definición, una tecnología debe satisfacer un conjunto de postulados microeconómicos. Del mismo modo, un estimador válido de una tecnología debe cumplir el mismo conjunto de axiomas. Dentro de los enfoques no paramétricos, destacan el Data Envelopment Analysis (DEA) y el Free Disposal Hull (FDH). Ambas metodologías son deterministas y cumplen el principio de mínima extrapolación. Esto implica que son susceptibles a errores de medición debido al ruido, y al sobreajuste de la muestra de datos usada para generar el estimador, limitando su capacidad de inferencia fuera de la muestra de datos.

La literatura reciente ha explorado el uso de técnicas de aprendizaje automático para mejorar la estimación de fronteras de producción. Sin embargo, no se ha explorado el uso de técnicas de *boosting*, una metodología de aprendizaje automático basada en la combinación secuencial de múltiples modelos débiles para mejorar la predicción. En esta Tesis se desarrolla una nueva metodología basada en el algoritmo de aprendizaje automático Gradient Tree Boosting para la estimación de fronteras de producción. Como se señala nada más comenzar, la Tesis es un compendio de tres artículos publicados, recogidos en los Apéndices A, B y C. En el primero de ellos, se adapta el algoritmo original de modo que el estimador resultante cumpla con los axiomas de monotonicidad y libre disponibilidad (necesarios para los estimadores de fronteras de producción), dando lugar al algoritmo EATBoosting. En el segundo, se muestra cómo calcular diferentes medidas de eficiencia técnica utilizando como base la tecnología generada por el nuevo estimador. Sin embargo, desde un punto de vista computacional, los problemas de optimización asociados al nuevo enfoque presentan miles de variables de decisión, lo que dificulta su resolución. Para hacer frente a este problema, también se propone una aproximación heurística a las medidas de eficiencia exactas. Finalmente, para facilitar el uso de esta nueva metodología por

parte de otros investigadores y profesionales, se ha desarrollado una librería en R denominada `BoostingDEA`, que incorpora las funcionalidades principales de DEA, FDH y EATBoosting.

La principal ventaja del nuevo enfoque radica en su capacidad para abordar el problema del sobreajuste. A diferencia de las técnicas tradicionales, nuestra metodología no subestima sistemáticamente la ineficiencia real de las Decision Making Units (DMUs), funcionando más como una herramienta inferencial que meramente descriptiva. Esto permite un mayor poder discriminatorio, conduciendo a una identificación más precisa de las ineficiencias, mejorando a FDH en los escenarios simulados tanto en error cuadrático medio como en sesgo. Además, nuestro enfoque proporciona una posible solución al problema de la maldición de la dimensionalidad, presente cuando la relación entre el número de DMUs y el número de variables es baja. La aplicación de EATBoosting en estos casos permite realizar un análisis de eficiencia más sólido y preciso.

Abstract

In econometrics and production engineering, a topic of interest is the evaluation of technical efficiency of firms from the estimation of the best practice frontier, which delineates the production possibility set or technology. By definition, a technology must satisfy a set of microeconomic postulates. Likewise, a valid estimator of a technology should meet the same set of axioms. Among non-parametric approaches, Data Envelopment Analysis (DEA) and Free Disposal Hull (FDH) stand out. Both methodologies are deterministic and fulfill the minimal extrapolation principle. This implies that they are susceptible to random and systematic measurement errors due to noise, and to overfitting of the sample data used to generate the estimator, limiting their ability for inference outside the data sample.

Recent literature has explored the use of machine learning techniques to improve the estimation of production frontiers. However, the use of boosting techniques, a machine learning methodology based on the sequential combination of multiple weak models to improve the final prediction, has not been explored. In this Thesis, a new methodology based on the Gradient Tree Boosting algorithm for the estimation of production frontiers is developed. As pointed out in the very beginning, the Thesis is a compendium of three published articles, gathered in Appendices A, B and C. In the first of these, the original algorithm is adapted so that the resulting estimator meets the axioms of monotonicity and free disposability (compulsory for production frontier estimators), leading to the EATBoosting algorithm. In the second one, it is shown how to calculate different measures of technical efficiency using the technology generated by the new estimator as a basis. Nevertheless, from a computational point of view, the new approach involves thousands of decision variables, making it difficult to solve. To address this issue, a heuristic approximation to exact efficiency measures is also proposed. Finally, to facilitate the use of this new methodology by other researchers and professionals, an R library called `BoostingDEA` has been developed, which includes the main functionalities of DEA, FDH, and EATBoosting.

The main advantage of the new approach lies in its ability to tackle the

problem of overfitting. Unlike traditional techniques, our methodology does not systematically underestimate the real inefficiency of the Decision Making Units (DMUs), functioning more as an inferential tool rather than merely descriptive. This allows for greater discriminatory power, leading to a more precise identification of inefficiencies, outperforming FDH in the simulated scenarios in both mean squared error and bias. Additionally, our approach provides a potential solution to the curse of dimensionality problem, which occurs when the ratio between the number of DMUs and the number of variables is low. The application of EATBoosting in these cases allows for a more robust and precise efficiency analysis.

Índice general

Capítulo 1

Introducción

Dentro del campo de la econometría y de la ingeniería de la producción, uno de los temas de interés es la evaluación de la eficiencia técnica. La evaluación de la eficiencia técnica tiene como objetivo medir el rendimiento de entidades, en este ámbito denominadas Unidades Tomadoras de Decisiones (*Decision Making Units* o DMUs, por sus siglas en inglés), las cuales convierten una serie de recursos (*inputs*) en productos o resultados (*outputs*). Las DMUs abarcan desde empresas orientadas al mercado, como por ejemplo bancos, hasta administraciones públicas sin ánimo de lucro, como colegios. La medición de su eficiencia técnica se lleva a cabo gracias al concepto de frontera de producción. Esta frontera indica la cantidad máxima de *outputs* obtenible a partir de una cierta cantidad dada de *inputs* y delimita el conjunto de posibilidades de producción, también llamado tecnología. Este conjunto está formado por todas las combinaciones de *inputs* y *outputs* que son técnicamente factibles, es decir, que se pueden llegar a producir. De este modo, la ineficiencia técnica de una DMU se determina como la desviación de dicha unidad respecto a la frontera de producción. No obstante, dicha frontera es desconocida y debe ser estimada.

Actualmente, existen múltiples enfoques en la literatura para la estimación de fronteras de producción. Tradicionalmente, estos enfoques se dividen en métodos paramétricos y no paramétricos. No obstante, las técnicas no paramétricas han tomado mayor relevancia en los últimos años debido a su mayor flexibilidad, a que requieren condiciones más relajadas (no requieren de la especificación de la función de producción) y a su tratamiento natural de múltiples *inputs* y *outputs* (Orea y Zofío, 2019). Cronológicamente hablando, fue Farrell (1957) el primer autor en mostrar cómo estimar una frontera de producción de forma no paramétrica. Posteriormente, esta línea de investigación fue continuada por Charnes et al.

(1978) y Banker et al. (1984), dando lugar al Análisis Envolvente de Datos (Data Envelopment Analysis, DEA, en inglés). El DEA genera una frontera de producción que envuelve superiormente a la nube de puntos observada (las DMUs) y que geométricamente se corresponde con una función de producción lineal a trozos, definiendo un conjunto de posibilidades de producción poliédrico. En particular, este conjunto determinado mediante DEA satisface una serie de axiomas básicos de la microeconomía, como son la convexidad y el principio de libre disponibilidad de *inputs* y *outputs*, es decir, siempre se puede hacer peor. La relajación de la propiedad de convexidad conduce a una metodología alternativa que genera una frontera de producción escalonada, denominada Free Disposal Hull (FDH) (Deprins et al., 1984).

Sin embargo, tanto DEA como FDH comparten un mismo inconveniente y es que ambos métodos son deterministas. Esto significa que los *inputs* y los *outputs* no son asumidos como variables aleatorias. No obstante, desde un punto de vista estadístico (y, por tanto, opuesto al determinista), estos datos representan una única realización de todos los resultados diferentes que podrían haberse obtenido del proceso de generación de datos subyacente (*Data Generation Process*, DGP). En concreto, en este contexto se asume la existencia de un modelo de probabilidad de los vectores de *inputs*-*outputs* sobre el conjunto de posibilidades de producción, que está representado por una función de densidad de probabilidad conjunta (Simar y Wilson, 1998; Simar y Wilson, 2000). Conceptualmente, si no se considera que los datos provienen de la DGP, no es posible dar cabida al hecho de que en un escenario real existe ruido en cada muestra realizada. Por lo tanto, métodos como DEA y FDH no pueden ir más allá de esa única realización, viéndose afectados por errores de medición aleatorios y sistemáticos que resultan en mediciones de eficiencia sesgadas. Además, esto se ve agravado por el postulado de mínima extrapolación que cumplen ambas técnicas. Este principio implica que el conjunto de posibilidades de producción estimado tanto por DEA como FDH es aquel que, cumpliendo los axiomas microeconómicos requeridos respectivamente, se sitúa más próximo a los datos, es decir, es el conjunto más pequeño. Como consecuencia, el estimador de la frontera de producción sufre de sobreajuste (Esteve et al., 2020; Valero-Carreras et al., 2021; Tsionas, 2022). En el campo de la estadística, el sobreajuste es un problema que se produce cuando el modelo estimado se ajusta perfectamente a la muestra de datos. Cuando esto ocurre, el modelo suele proporcionar una descripción de la muestra, en lugar de información relevante sobre la DGP que está detrás de la generación de las observaciones concretas. Por lo tanto, aunque ambos métodos puedan medir correctamente la eficiencia de un conjunto concreto de observaciones, esta estimación no es representativa desde la perspectiva de la DGP subyacente; es decir, no es capaz de realizar inferencia fuera de la muestra de datos.

Recientemente, se ha producido un notable avance en la literatura sobre eficiencia técnica con la integración de métodos de aprendizaje automático (*machine learning*) para estimar la frontera eficiente subyacente, con el objetivo de predecir la verdadera función de producción y mejorar la naturaleza determinista de DEA y FDH. Parmeter y Racine (2013) introdujeron estimadores de frontera basados en *kernels* no paramétricos, estableciendo una frontera de producción más suave. Daouia et al. (2016) desarrollaron una técnica para estimar funciones de producción utilizando *splines* cuadráticos y cúbicos. Esteve et al. (2020) introdujeron los Árboles de Análisis de Eficiencia (EAT), adaptando los árboles de regresión para mejorar FDH. Aparicio et al. (2021) extendieron EAT con suposiciones de convexidad, dando lugar a los Árboles de Análisis de Eficiencia Convexificados (CEAT). Valero-Carreras et al. (2021, 2022) aplicaron máquinas de soporte vectorial para estimar funciones de producción, denominadas fronteras de vectores de soporte. Olesen y Ruggiero (2022) propusieron hiperplanos articulados como una representación flexible y no paramétrica de funciones de producción. Tsionas (2022) introdujo árboles de regresión probabilísticos para la estimación de funciones de producción cóncavas, monótonas no decrecientes y suaves, para datos de panel. Guerrero et al. (2022) desarrollaron un método para estimar tecnologías poliédricas siguiendo el principio de minimización del riesgo estructural. Esteve et al. (2022) adaptaron Random Forest para evaluar la eficiencia dentro del marco FDH. Aparicio et al. (2023) sugirieron el uso de árboles de regresión para la estimación de eficiencia dinámica. Moragues et al. (2023) emplearon una máquina de soporte vectorial de una clase adaptada para la estimación de tecnologías poliédricas; y España et al. (2024) propusieron modelos aditivos basados en *splines* para estimar funciones de producción.

No obstante, ninguno de estos enfoques se basa en *boosting*, una de las técnicas más interesantes dentro del aprendizaje automático. Esta técnica se basa en la combinación secuencial de múltiples modelos (*ensembles*), y ha sido aplicada en un amplio espectro de aplicaciones prácticas (véase, por ejemplo, la revisión de Ferreira y Figueiredo, 2012). De hecho, el algoritmo de Gradient Boosting (la implementación de la técnica de *boosting* para problemas de regresión) (Friedman, 1999) se considera una de las ideas sobre aprendizaje automático más potentes introducidas en la literatura (Hastie et al., 2017) para la estimación de una cierta variable respuesta a través de una serie de variables predictoras. A diferencia del enfoque de los modelos estadísticos clásicos o del enfoque tradicional del aprendizaje automático, donde directamente se construye un único modelo predictivo “fuerte”, es decir, un modelo que está altamente correlacionado con la variable de respuesta, el enfoque de *ensembles* se basa en la combinación de un gran número de modelos relativamente "débiles" (modelos que están levemente correlacionados con la variable de respuesta), denominados modelos base, para obtener una predicción mejor de la que es capaz de alcanzar de manera individual un único modelo "fuerte". Sin embargo, la peculiaridad de la técnica del *boosting* es que, a diferencia de otras técnicas de *ensembles*, como Random Forest (Breiman, 2001), donde el resultado final es la media de la predicción de los diferentes modelos que conforman el *ensemble*, los algoritmos de *boosting* se basan en una estrategia iterativa hacia adelante, donde un nuevo modelo es creado secuencialmente. En concreto, en cada iteración, un nuevo modelo "débil" es entrenado con respecto al error cometido por los modelos del *ensemble* creado hasta el momento.

Los mejores resultados en Gradient Boosting se obtienen utilizando árboles CART (*Classification and Regression Trees*) (Breiman et al., 1984) de un tamaño fijo (para limitar su poder predictivo) como modelos base (Hastie et al., 2017). Estos árboles permiten una partición recursiva del espacio de los predictores. La idea que hay detrás de CART es relativamente simple: un cierto criterio es utilizado para generar biparticiones de los datos de forma recursiva hasta que no es posible realizar más subdivisiones o se cumple cierta regla de parada. En los árboles CART para regresión se suele utilizar como medida de bondad de la predicción la suma de la diferencia al cuadrado de los datos respecto a la media de la variable respuesta en cada uno de los nodos. Cuanto más "profundo" es el árbol, menor es el error cometido, pero menor también es el poder predictivo (generalización) de la técnica. De hecho, hay una relación directa entre la profundidad del árbol y la intensidad de sobreajuste del modelo. Para este caso especial, Friedman (2001) propone una modificación del algoritmo tradicional, denominado Gradient Tree Boosting, que mejora la calidad de ajuste de cada modelo base, evitando el problema del sobreajuste.

El primer objetivo de esta Tesis es crear una nueva metodología basada en el algoritmo de Gradient Tree Boosting para la estimación de fronteras de producción. Dicha adaptación no es trivial, dado que el estimador generado por DEA y FDH requiere del cumplimiento de ciertas propiedades no exigidas en los casos generales abordados a través de Gradient Tree Boosting. En particular, el estimador debe cumplir como mínimo los axiomas de monotonicidad y envoltura de los datos, al igual que FDH. Estos dos axiomas se traducen en que la función estimada por el algoritmo debe ser monótona no decreciente y debe envolver a los datos por encima. De este modo, el problema se puede entender como la creación de un estimador con restricciones de forma. Además, el algoritmo Gradient Tree Boosting estándar estima a nivel de promedio, al igual que la regresión clásica, y debe abandonar esta filosofía para determinar una frontera de mejores prácticas que evite, a su vez, el sobreajuste. Para ello, se deben modificar ciertos pasos del algoritmo original y se sustituyen los árboles CART por los árboles EAT (*Effiency Analysis Trees*), una adaptación de estos al contexto de la estimación de fronteras de producción (Esteve et al., 2020). El desarrollo de esta metodología se lleva a cabo en el Apéndice A, dando lugar al algoritmo bautizado como EATBoosting.

Como se ha comentado previamente, la ineficiencia técnica se define como la distancia al estimador de la frontera de producción desde cualquier punto interior del conjunto de posibilidades de producción. No obstante, existen numerosos caminos potenciales hacia la frontera desde cualquier punto situado en el interior de dicho conjunto. Cada camino está asociado a una medida de eficiencia técnica determinada. Estas medidas utilizan las tecnologías estimadas por FDH y DEA como base para definir los problemas de optimización que permiten calcular los valores de eficiencia (*scores*) correspondientes, difiriendo en las trayectorias seleccionadas para proyectar cada DMU en la frontera. El segundo objetivo de esta Tesis es determinar cómo calcular las diferentes medidas de eficiencia técnica usando como base la tecnología generada por el estimador desarrollado en el Apéndice A. Esto se lleva a cabo en el Apéndice B. No obstante, desde un punto de vista computacional, los problemas de optimización relacionados con el nuevo enfoque presentan miles de variables de decisión, lo que dificulta su resolución. Para hacer frente a este problema, también se propone una aproximación heurística a las medidas exactas.

Finalmente, con el objetivo de fomentar el uso real de la nueva técnica por parte de otros investigadores, gestores y responsables políticos, se ha desarrollado una librería en R, bautizada como `BoostingDEA`, que incorpora las principales funcionalidades de DEA, FDH y EATBoosting. Los detalles de dicha librería se presentan en el Apéndice C.

Capítulo 2

Materiales y métodos

En este capítulo se detallan los conceptos fundamentales relacionados con *Free Disposal Hull* (FDH) y sus medidas de eficiencia técnica asociadas, los Árboles de Eficiencia Técnica (*Effiency Analysis Trees*, EAT) y el algoritmo Gradient Tree Boosting.

2.1. Free Disposal Hull (FDH)

Consideremos un conjunto observado $\aleph$ formado por n DMUs a evaluar, donde cada DMU$_i$ consume un perfil de *inputs* $\boldsymbol{x}_i = (x_i^{(1)}, ..., x_i^{(m)}) \in \mathbb{R}_+^m$ para producir un perfil de *outputs* $\boldsymbol{y}_i = (y_i^{(1)}, ..., y_i^{(s)}) \in \mathbb{R}_+^s$ [1]. En el caso mono-*output*, asumimos que existe una función desconocida $f(\boldsymbol{x}) : \mathbb{R}_+^m \to \mathbb{R}_+^1$ generadora de $\aleph$ que dada un perfil de *inputs* $\boldsymbol{x}$ indica la cantidad máxima de *ouputs* producible. Sin embargo, también asumimos la presencia de ineficiencia técnica en el proceso de producción. De este modo, el perfil de *ouputs* observado para dicho perfil $\boldsymbol{x}$ es $y = f(\boldsymbol{x}) - u$, donde $u \geq 0$ es una variable aleatoria que representa la ineficiencia técnica.

El conjunto de posibilidades de producción o tecnología Ψ es el conjunto de combinaciones técnicamente factibles de $(\boldsymbol{x}, \boldsymbol{y})$, definido formalmente de la siguiente manera.

$$\Psi := \{(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}_+^{m+s} : \boldsymbol{x} \text{ puede producir } \boldsymbol{y}\} \tag{2.1}$$

De este modo, la frontera eficiente $\partial(\Psi)$ es un subconjunto de Ψ definido

[1]Respecto a lo notación, se usa negrita para hacer referencia a vectores, y sin negrita, a escalares.

formalmente como aparece a continuación [2].

$$\partial(\Psi) := \{(\boldsymbol{x},\boldsymbol{y}) \in \Psi : \hat{\boldsymbol{x}} < \boldsymbol{x}, \hat{\boldsymbol{y}} > \boldsymbol{y} \Rightarrow (\hat{\boldsymbol{x}},\hat{\boldsymbol{y}}) \notin \Psi\} \tag{2.2}$$

Free Disposal Hull (Deprins et al., 1984) proporciona un estimador de la tecnología cumpliendo los axiomas de envoltura de los datos y libre disponibilidad de *inputs* y *outputs*. Estos se traducen en que $(\boldsymbol{x}_i,\boldsymbol{y}_i) \in \Psi$ para todo $i = 1,...,n$, y en que si $(\boldsymbol{x},\boldsymbol{y}) \in \Psi$, $\boldsymbol{x}' \geq \boldsymbol{x}$ y $0 \leq \boldsymbol{y}' \leq \boldsymbol{y}$, entonces $(\boldsymbol{x}',\boldsymbol{y}') \in \Psi$, respectivamente. Además, también cumple el principio de mínima extrapolación, es decir, de todos los posibles conjuntos que cumplen los dos axiomas anteriores, es el situado lo más cerca posible a la nube de datos (conjunto más pequeño). Formalmente, el estimador de la tecnología FDH se define de la siguiente manera.

$$\begin{aligned}\hat{\Psi}_{FDH} =& \big\{(\boldsymbol{x},\boldsymbol{y}) \in \mathbb{R}^{m+s}_{+} : y^{(r)} \leq \sum_{i=1}^{n} \lambda_i y_i^{(r)}, r = 1,...,s, x^{(p)} \geq \sum_{i=1}^{n} \lambda_i x_i^{(p)}, p = 1,...,m,\\ & \sum_{i=1}^{n} \lambda_i = 1, \lambda_i \in \{0,1\}, i = 1,...,n\big\}\end{aligned} \tag{2.3}$$

Si además consideramos el axioma de convexidad, es decir, que dados los pares $(\boldsymbol{x},\boldsymbol{y}) \in \Psi$ y $(\boldsymbol{x}',\boldsymbol{y}') \in \Psi$, entonces $\lambda(\boldsymbol{x},\boldsymbol{y}) + (1-\lambda)(\boldsymbol{x}',\boldsymbol{y}') \in \Psi$ para todo $\lambda \in [0,1]$, Banker et al. (1984) define el estimador de la tecnología DEA de la siguiente forma.

$$\begin{aligned}\hat{\Psi}_{DEA} =& \big\{(\boldsymbol{x},\boldsymbol{y}) \in \mathbb{R}^{m+s}_{+} : y^{(r)} \leq \sum_{i=1}^{n} \lambda_i y_i^{(r)}, r = 1,...,s, x^{(p)} \geq \sum_{i=1}^{n} \lambda_i x_i^{(p)}, p = 1,...,m,\\ & \sum_{i=1}^{n} \lambda_i = 1, \lambda_i \geq 0, i = 1,...,n\big\}\end{aligned} \tag{2.4}$$

La única diferencia entre los estimadores de la tecnologías en 2.3 y 2.4 es la naturaleza de la variables de decisión λ. Mientras que en 2.3 estas son binarias, en 2.4 son continuas. Esto implica que mientras FDH proporciona estimadores de la frontera de producción mediante funciones escalonadas, DEA lo hace mediante

[2]Dados dos vectores $\boldsymbol{z} = (z^{(1)},...,z^{(g)})$ y $\hat{\boldsymbol{z}} = (\hat{z}^{(1)},...,\hat{z}^{(g)})$, $\boldsymbol{z} \leq \hat{\boldsymbol{z}}$ indica que $z^{(l)} \leq \hat{z}^{(l)}$ para todo $l = 1,...,g$. Del mismo modo, $\boldsymbol{z} < \hat{\boldsymbol{z}}$ indica que $z^{(l)} < \hat{z}^{(l)}$ para todo $l = 1,...,g$.

funciones lineales a trozos. No obstante, durante esta Tesis nuestro punto de referencia va a ser la tecnología FDH. Esto se debe a dos razones principales: la primera es la mayor flexibilidad de la tecnología FDH al no tener que asumir el axioma de convexidad (véase, por ejemplo, Kerstens y Van de Woestyne, 2021; Ang et al., 2023; Kerstens et al., 2022). La segunda está relacionada con la naturaleza de los árboles de regresión CART, y por consiguiente con los árboles EAT, los cuales forman parte de la base de nuestro enfoque, y que producen estimadores escalonados de las funciones objetivo. Por consiguiente, la relación gráfica entre FDH y la nueva técnica es evidente.

La ineficiencia de las DMUs en $\aleph$ se calcula como la distancia de estas a la frontera de Ψ. En la literatura, se han desarrollado diferentes medidas que permiten cuantificar la eficiencia dependiente de la dirección en la que se proyectan las unidades hacia la frontera. Entre ellas, destaca la medida radial orientación *output* (Farrell, 1957), en la cual se dejan fijos los *inputs* y se determina la proporción en la que deben incrementarse los *outputs* de forma equiproporcional para situar la unidad evaluada sobre la frontera. Formalmente, dada la DMU$_k$ a evaluar, $\phi(\boldsymbol{x}_k, \boldsymbol{y}_k) = \text{máx}\{\phi_k \in \mathbb{R} : (\boldsymbol{x}_k, \phi_k \boldsymbol{y}_k) \in \Psi\}$. Bajo el enfoque FDH, la medida radial orientación *output* se calcula a través del siguiente problema de optimización lineal.

$$\begin{aligned}
\phi^{FDH}(\boldsymbol{x}_k, \boldsymbol{y}_k) = \text{máx}\, & \phi_k \\
s.t.& \\
& \sum_{i=1}^{n} \lambda_i x_i^{(p)} \leq x_k^{(p)}, p = 1, ..., m, \\
& \sum_{i=1}^{n} \lambda_i y_i^{(r)} \geq \phi_k y_k^{(r)}, r = 1, ..., s, \\
& \sum_{i=1}^{n} \lambda_i = 1, \lambda_i \in \{0,1\}, i = 1, ..., n
\end{aligned} \tag{2.5}$$

En un contexto de producción con un único *output*, la función de producción generada por la tecnología estimada por FDH puede aproximarse mediante la expresión $f_{FDH}(\boldsymbol{x}_k) = y_k \cdot \phi_k$ para los vectores de *inputs* observados. En la figura 2.1 se muestra un ejemplo gráfico de la función de producción estimada por FDH, resaltando su típica forma escalonada no decreciente.

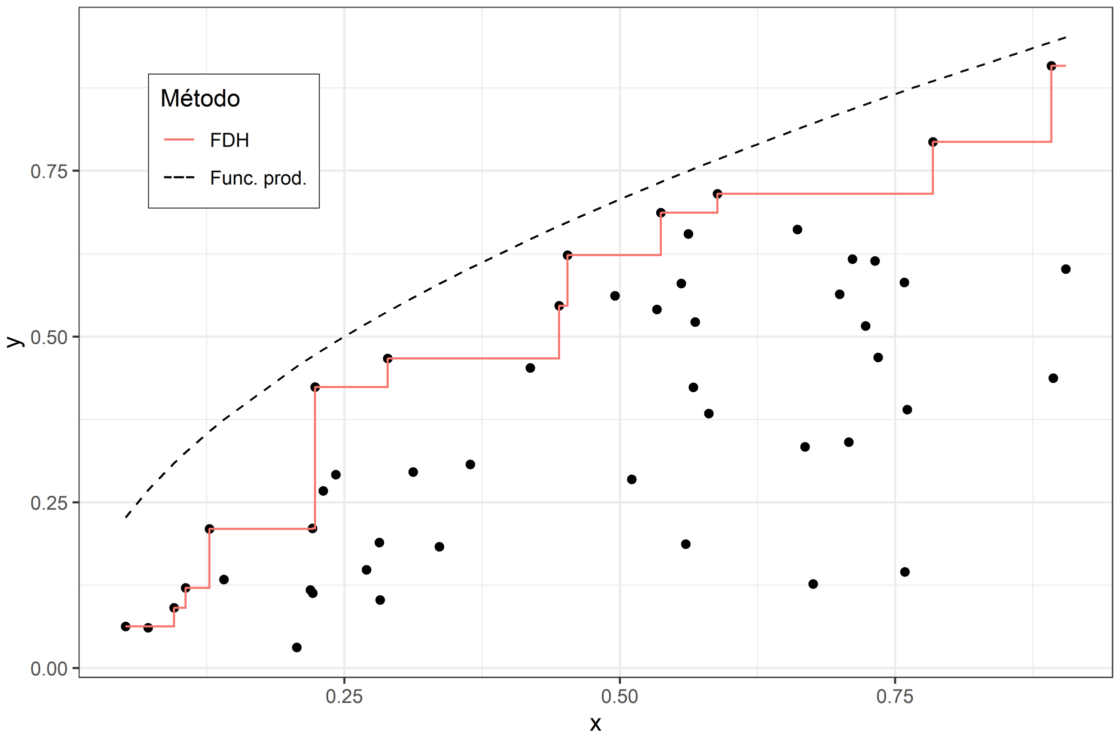

Figura 2.1: Ejemplo de la función de producción estimada por FDH en un escenario mono-*input* mono-*output* con $y = x^{0{,}5}e^{-u}$, donde $x \sim Uni[0, 1]$ y $u \sim |N(0, 0{,}4)|$.

2.2. Eficiency Analysis Trees

Los Árboles de Eficiencia Técnica (Efficiency Analysis Trees, EAT), introducidos en Esteve et al. (2020), son una técnica basada en el aprendizaje automático que proporciona estimaciones de la frontera de producción satisfaciendo los mismos axiomas asumidos por FDH, excepto el principio de mínima extrapolación. Desde el punto de vista computacional, los árboles EAT son más complejos que FDH, puesto que son una adaptación de los árboles CART (Breiman et al., 1984) al contexto de la eficiencia técnica y, por tanto, implementan un algoritmo recursivo de bipartición. Sin embargo, desde el punto de vista del estimador final, EAT proporciona una estimación menos sobreajustada de la tecnología subyacente. En la Figura 2.2 se muestra un ejemplo gráfico comparando el estimador de la función de producción proporcionado por FDH y por EAT.

El algoritmo de los árboles EAT funciona de la siguiente manera. Dada una muestra de datos $\aleph = \{(\boldsymbol{x}_i, \boldsymbol{y}_i)\}_{i=1,\dots,n}$, se parte de un nodo t_0, el cual contiene todas las observaciones. Este nodo debe dividirse en dos hijos, el nodo hijo izquierdo, t_L, y el nodo hijo derecho, t_R, que a su vez deberán dividirse en pasos posteriores.

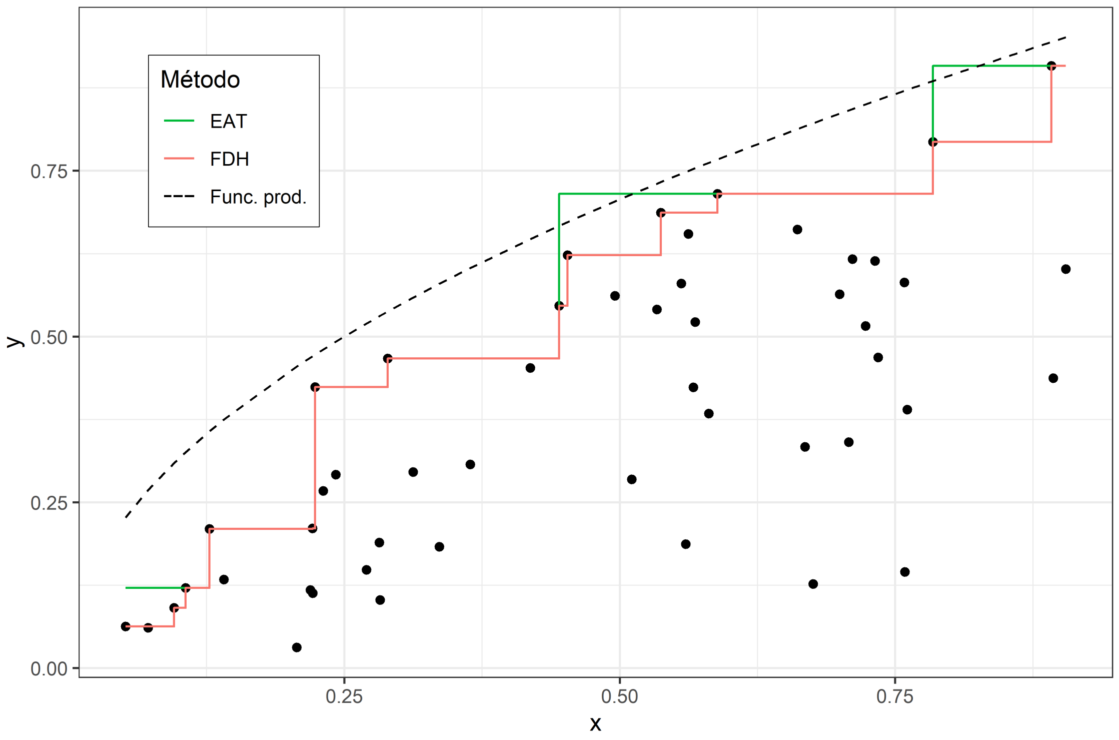

Figura 2.2: Ejemplo de la función de producción estimada por FDH y por EAT en un escenario mono-*input* mono-*output* con $y = x^{0{,}5}e^{-u}$, donde $x \sim Uni[0,1]$ y $u \sim |N(0, 0{,}4)|$.

Para explicar cómo se hace este proceso de partición, supongamos que tenemos un nodo, t, en la estructura de árbol que hay que dividir. Este nodo contendrá un subconjunto de la muestra de datos original $\aleph$. Entonces, el algoritmo selecciona un *input* p,$p = 1, ..., m$, y un valor para este *input* $s^{(p)} \in S^{(p)}$, donde $S^{(p)}$ es el conjunto de todos los valores observados para la variable p en $\aleph$, de modo que se minimice la suma del error cuadrático medio (*Mean Square Error*, MSE) asociado a las observaciones pertenecientes al nodo hijo izquierdo y al nodo hijo derecho. Las observaciones pertenecientes al nodo hijo izquierdo serán aquellas que cumplan la condición $x^{(p)} < s^{(p)}$, mientras que las pertenecientes al nodo hijo derecho serán aquellas que cumplen $x^{(p)} \geq s^{(p)}$. Formalmente, esto consiste en encontrar la mejor combinación $(x^{(p)*}, s^{(p)*})$ que minimice la expresión

$$\begin{aligned} MSE\,(t_L) + MSE\,(t_R) = & \frac{1}{n} \sum_{(\boldsymbol{x}_i, \boldsymbol{y}_i) \in t_L} \sum_{r=1}^{s} \left(y_i^{(r)} - y^{(r)}\,(t_L) \right)^2 + \\ & \frac{1}{n} \sum_{(\boldsymbol{x}_i, \boldsymbol{y}_i) \in t_R} \sum_{r=1}^{s} \left(y_i^{(r)} - y^{(r)}\,(t_R) \right)^2 \end{aligned}$$

donde $y^{(r)}(t)$ representa la predicción del r-ésimo *output* en el nodo t. En el algoritmo se considera que un nodo es un nodo hoja cuando se satisface cierta regla de parada y, como consecuencia, no debe volver a dividirse. La regla de parada usada en EAT es $n(t) \leq n_{min} = 5$, donde $n(t)$ representa el número de observaciones en el nodo t. Esta regla es la misma que la propuesta por Breiman et al. (1984) para los árboles CART.

Una parte fundamental del algoritmo es la definición del predictor $y^{(r)}(t)$ de modo que este envuelva a los datos por encima y satisfaga el principio de libre disponibilidad. Para esto último, se necesita recurrir al concepto de soporte de un nodo, denominado R. Al dividir un nodo, lo que realmente se está haciendo es una partición disjunta del espacio de los *inputs*. Se define el soporte de un nodo como el subconjunto del espacio de los *inputs* asociado a dicho nodo. Formalmente, dichas regiones se definen como $R_t = \{\boldsymbol{x} \in \mathbb{R}^m_+ : a_t^{(p)} \leq x^{(p)} < b_t^{(p)}, p = 1, ..., m\}$, donde los parámetros $a_t^{(p)}$ y $b_t^{(p)}$ surgen a partir de los valores escogidos durante el proceso de división. A partir del concepto de soporte, también se puede definir el concepto de Pareto-dominancia entre nodos. Se define el conjunto de nodos hoja Pareto-dominados por t como $I(t) = \{t' \in \tilde{T}(\aleph) - t : \exists \boldsymbol{x} \in R_t \text{ y } \exists \boldsymbol{x}' \in R_{t'} \text{ tal que } \boldsymbol{x}' \leq \boldsymbol{x}\}$, donde $\tilde{T}(\aleph)$ es el conjunto de todos los nodos hoja del árbol.

A partir de estos conceptos, al dividir un nodo t, se define el estimador de su nodo hijo izquierdo t_L para la el *output* r-ésimo, $y^{(r)}(t_L)$, como el máximo entre el máximo de los valores para el *output* r de las observaciones perteneciente a t_L, y el máximo de los predictores del *output* r-ésimo de los nodos Pareto-dominados por t_L. Formalmente,

$$y^{(r)}(t_L) = \text{máx}\{\text{máx}\{y_i^{(r)} : (\boldsymbol{x}_i, \boldsymbol{y}_i) \in t_L\}, \text{máx}\{y^{(r)}(t') : t' \in I(t_L)\}\}$$

donde $r = 1, ..., s$. El estimador del nodo hijo derecho t_R se puede definir de forma análoga, pero siempre coincide con la estimación del nodo padre, es decir, $y^{(r)}(t_R) = y^{(r)}(t)$, con $r = 1, ..., s$.

2.3. Gradient Tree Boosting

El algoritmo Gradient Boosting (Friedman, 1999) es una implementación al contexto de la regresión de la técnica de *boosting*, la cual se basa en crear modelos secuencialmente, de modo que estos minimicen el error cometido hasta el momento.

Por tanto, el resultado del algoritmo de Gradient Boosting es una función de predicción $f(x)$ de la forma de una suma ponderada de funciones $h(x)$, denominados modelos "débiles" o base, pertenecientes a una clase de modelos H, los cuales intentan minimizar una determinada función de pérdidas $L(y, f(x))$ (por ejemplo, el MSE).

Para ello, dada una muestra de datos $\aleph = \{(x_i, y_i)\}_{i=1,...,n}$, el algoritmo empieza con una primera predicción que consiste en la función constante que minimice la función de pérdidas para todas las observaciones, $f_0(x) = \arg\min\limits_{\gamma} \sum\limits_{i=1}^{n} L(y_i, \gamma)$, donde γ representa el valor que minimiza dicha función (en el caso del MSE, la media). A continuación, se calculan los componentes del gradiente negativo de la función de pérdidas, conocidos como pseudo-residuos. Los pseudo-residuos, e_q, se definen como $e_{iq} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{q-1}}$. Posteriormente, debe entrenarse el modelo $h_q(x)$ usando como muestra de datos aquella formada por los *inputs* originales y los pseudo-residuos. Es decir, el modelo $h_q(x)$ es entrenado con la muestra $\tilde{\aleph}_q = \{x_i, e_{iq}\}_{i=1,...,n}$. Una vez entrenado, la predicción de la iteración actual se actualiza como $f_q(x) = f_{q-1}(x) + \gamma_q h_q(x_i)$, donde γ_q se calcula resolviendo el modelo de optimización $\gamma_q = \arg\min\limits_{\gamma} \sum\limits_{i=1}^{n} L(y_i, f_{q-1}(x_i) + \gamma h_q(x_i))$.

Como adaptación del algoritmo estándar, Friedman (2001) propuso el algoritmo Gradient Tree Boosting, en el cual se utilizan árboles CART de un tamaño fijo como modelos base en el algoritmo. De este modo, este algoritmo ajusta un árbol de decisión $h_q(x)$ a los pseudo-residuos en la q-ésima iteración, limitando su tamaño a un máximo de J_q nodos hoja. Esta condición de parada se establece para reducir el poder predictivo de los árboles CART, que por sí solos son considerados modelos "fuertes". Como se ha comentado en la sección anterior, los árboles dividen el espacio de los *inputs* en regiones disjuntas (soportes), $R_{1q}, ..., R_{J_q q}$. Sea γ_{jq} la predicción del árbol $h_q(x)$ para el soporte R_{jq}, con $j = 1, ..., J_q$. De este modo, $h_q(x) = \sum\limits_{j=1}^{J_q} \gamma_{jq} I\left(x \in R_{jq}\right)$, donde $I\left(x \in R_{jq}\right) = \begin{cases} 1 & \text{si } x \in R_{jq} \\ 0 & \text{si } x \notin R_{jq} \end{cases}$. Friedman (2001) propuso modificar el algoritmo original para que en vez de calcular el γ_q que minimice el error de todo el espacio de los *inputs* (el árbol completo), utilizar las predicciones que minimicen los soportes de los nodos hoja, es decir, γ_{jq}. Así, la actualización de la predicción se calcula como $f_q(x) = f_{q-1}(x) + \sum\limits_{j=1}^{J_q} \gamma_{jq} I\left(x \in R_{jq}\right)$. Además, para evitar que los árboles sean demasiado profundos (demasiados nodos hoja) en las primeras iteraciones, lo cual disminuye el rendimiento y aumenta la complejidad del algoritmo, Friedman propuso restringir todos los árboles al mismo tamaño ($J_q = J, \forall q$). Así mismo, también propuso introducir una tasa de aprendizaje v, $0 < v \leq 1$, que sirve como factor de regularización para limitar la contribución de cada árbol con el objetivo de mejorar la capacidad de generalización del modelo final. El Algoritmo 1 muestra los pasos de Gradient Tree Boosting.

Algoritmo 1: Gradient Tree Boosting

```
  **Entrada:** $\aleph$, $Q$, $J$, $v$
  **Salida**  : $\hat{f}_Q(\mathbf{x})$
  Sea $f_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma)$
  **para** $q = 1$ *hasta* $Q$ **hacer**
      Calcular los pseudo-residuos: $e_{iq} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=f_{q-1}}, i = 1, ..., n$
      Ajustar un árbol CART a la muestra $\tilde{\aleph}_q = \{x_i, e_{iq}\}_{i=1,...,n}$ dadas las
        regiones terminales $R_{jq}$, $j = 1, ..., J$
      Obtener la predicción del árbol CART para cada región terminal $R_{jq}$: $\gamma_{jq}$
      Actualizar $f_q(x) = f_{q-1}(x) + v \cdot \sum_{j=1}^{J} \gamma_{jq} I(x \in R_{jq})$
  **fin**
  **devolver** $f_Q(x)$
```

Capítulo 3

Discusión de los resultados principales

Este capítulo pretende recoger las principales aportaciones originales de la tesis, de forma que se tenga una visión panorámica de la misma. Para más detalles, véanse los Apéndices A, B y C, los cuales corresponden a las secciones de este capítulo respectivamente.

3.1. Algoritmo EATBoosting

El algoritmo EATBoosting es una adaptación del algoritmo Gradient Tree Boosting para la estimación de fronteras de producción, generando estimadores que satisfacen los axiomas de libre disponibilidad y de envoltura de los datos, pero no el principio de mínima extrapolación. Para ello, se propone la modificación de ciertos pasos del algoritmo estándar, así como la sustitución de los árboles CART por los árboles EAT.

Dentro del proceso de adaptación, lo primero que debe hacerse es seleccionar una función de pérdidas $L(\boldsymbol{y}, \boldsymbol{f}(\boldsymbol{x}))$. Puesto que los árboles EAT recurren al error cuadrático medio (MSE) durante el proceso de división de los nodos, se define nuestra función de pérdidas como $L(\boldsymbol{y}, \boldsymbol{f}(\boldsymbol{x})) = \frac{1}{2}(\boldsymbol{y}_i - \boldsymbol{f}(\boldsymbol{x}_i))^2, \forall i = 1, ..., n$ (Hastie et al., 2017). Ahora, siguiendo los pasos del Algoritmo 1, se debe inicializar $\boldsymbol{f}_0(\boldsymbol{x})$ con el valor constante que minimice dicha función de pérdidas. No obstante, en este contexto debemos imponer una condición adicional para asegurar que la predicción envuelva por encima a los datos, es decir, que $\boldsymbol{f}(\boldsymbol{x}_i) \geq \boldsymbol{y}_i, \forall i = 1, ..., n$. Por tanto, en este caso, se puede demostrar que dicha función es $\boldsymbol{f}_0(\boldsymbol{x}) = \left(\max_{1\leq i\leq n}\{y_i^{(1)}\},...,\max_{1\leq i\leq n}\{y_i^{(s)}\}\right)$ (Esteve et al., 2020). Ahora, durante una serie finita de iteraciones, $q = 1,..,Q$, un nuevo modelo $\boldsymbol{h}_q(\boldsymbol{x})$ se debe ajustar a los componentes del gradiente negativo del error. En este contexto, los pseudo-residuos toman directamente la forma $\boldsymbol{e}_{iq} = -\left[\frac{\partial L(\boldsymbol{y}_i, f(\boldsymbol{x}_i))}{\partial \boldsymbol{f}(\boldsymbol{x}_i)}\right]_{\boldsymbol{f}=\boldsymbol{f}_{q-1}} = \boldsymbol{y}_i^{(s)} - \boldsymbol{f}^{(s)}(\boldsymbol{x}_i)$. Además, el modelo $\boldsymbol{h}_q(\boldsymbol{x})$ se define como el árbol resultante de ejecutar el algoritmo de los árboles EAT, modificando la regla de parada original de modo que el árbol resultante tenga un máximo de J nodos hoja, de forma análoga a lo se hace en Friedman (2001) para limitar el poder predictivo de los árboles CART. Este modelo debe, por tanto, entrenarse con la muestra $\tilde{\aleph}_q = \{\boldsymbol{x}_i, \boldsymbol{e}_{iq}\}_{i=1,...,n}$. En este caso, el valor γ_{jq} que minimiza los soportes R_{jq} del espacio de los *inputs* y que cumple los axiomas requeridos es la predicción del árbol EAT para dichas regiones. Finalmente, la forma en la que se actualiza la predicción es análoga a cómo se hace en el algoritmo estándar. Como resultado, se obtiene el Algoritmo 2.

Algoritmo 2: EATBoosting

```
Entrada: $\aleph$, $Q$, $J$, $v$
Salida  : $\boldsymbol{f}_Q(\boldsymbol{x})$
Sea $\boldsymbol{f}_0(\boldsymbol{x}) = \left(\max_{1\leq i\leq n}\{y_i^{(1)}\},...,\max_{1\leq i\leq n}\{y_i^{(s)}\}\right)$
para $q = 1$ hasta $Q$ hacer
    Calcular los pseudo-residuos: $\boldsymbol{e}_{iq} = \boldsymbol{y}_j^{(s)} - \boldsymbol{f}^{(s)}(\boldsymbol{x}_i), i = 1,...,n$
    Ajustar un árbol EAT a la muestra $\tilde{\aleph}_q = \{\boldsymbol{x}_i, \boldsymbol{e}_{iq}\}_{i=1,...,n}$ dadas las
      regiones terminales $R_{jq}$, $j = 1,...,J$
    Obtener la predicción del árbol EAT para cada región terminal $R_{jq}$: $\gamma_{jq}$
    Actualizar $\boldsymbol{f}_q(\boldsymbol{x}) = \boldsymbol{f}_{q-1}(\boldsymbol{x}) + v \cdot \sum_{j=1}^{J}\gamma_{jq}I(\boldsymbol{x} \in R_{jq})$
fin
devolver $\boldsymbol{f}_Q(\boldsymbol{x})$
```

Por lo tanto, hay tres hiperparámetros que controlan el rendimiento del modelo. Esos parámetros son el número de iteraciones que realiza el algoritmo (Q), el número de hojas finales de cada árbol EAT en cada iteración (J) y la tasa de aprendizaje (v). Respecto a los valores de los mismos, aumentar el número de iteraciones reduce el error en un conjunto de entrenamiento, pero fijarlo demasiado alto puede llevar a un sobreajuste. Del mismo modo, cuanto mayor sea el número de hojas, más probable será que el modelo sobreajuste los datos de entrenamiento. En cuanto a la tasa de aprendizaje, el uso de tasas de aprendizaje pequeñas produce mejoras en la capacidad de generalización de los modelos, pero tiene el precio de aumentar el tiempo de cálculo, ya que una tasa de aprendizaje más baja requiere más iteraciones. Como consecuencia, a la hora de ajustar los valores de los hiperparámetros, será necesario llegar a un compromiso sobre los valores de los mismos.

El Algoritmo 2 podría considerarse la extensión natural del Gradient Tree Boosting para estimar funciones de producción. Sin embargo, el estimador derivado del algoritmo EATBoosting no sería un estimador válido de una función de producción en microeconomía a menos que satisfaga, como mínimo, los axiomas de la libre disponibilidad y de envoltura de los datos. Estos dos axiomas se traducen en que la función estimada debe ser monótona no decreciente y en que debe envolver las observaciones desde arriba, respectivamente.

Proposición 1 (Proposición 4 A) *Sea $v \in (0,1]$, $f_Q(\boldsymbol{x})$ es una función monótona no decreciente*

Proposición 2 (Proposición 5 A) *Sea $v \in (0,1]$, $f_Q(\boldsymbol{x}_i) \geq y_i$ para todo $i = 1,..,n$.*

Por regla general, los árboles EAT y el algoritmo EATBoosting no generan el mismo estimador de la función de producción. No obstante, el siguiente resultado demuestra que cuando se realiza una sola iteración del algoritmo ($Q = 1$), se fija el número de nodos hoja de cada árbol EAT en cada iteración al número de nodos hoja del árbol EAT entrenado con esa muestra ($J = \tilde{T}(\aleph)$) y no se aplica el factor de regularización ($v = 1$), ambos estimadores coinciden.

Proposición 3 (Proposición 3 A) *Si $Q = 1$, $J = \tilde{T}(\aleph)$ y $v = 1$, $f_Q(\boldsymbol{x}_i) = f_{EAT}(\boldsymbol{x}_i)$*

Además, gracias a la equivalencia entre los árboles EAT y la metodología FDH (Esteve et al., 2020), en el caso mono-*input* la función de producción estimada por EATBoosting y por FDH asumiendo los valores anteriores para los hiperparámetros y modificando la regla de parada de los árboles EAT de modo que se deje crecer los árboles hasta que los nodos hoja contengan una única observación ($n_{min} = 1$), coinciden.

Proposición 4 (Colorario 1 A) *Si $m = 1$, $n_{min} = 1$, $Q = 1$, $J = \tilde{T}(\aleph)$ y $v = 1$, $f_Q(\boldsymbol{x}_i) = f_{EAT}(\boldsymbol{x}_i)$*

En la Figura 3.1 se muestra un ejemplo gráfico comparando la función estimada por FDH y por EATBoosting. Gráficamente se puede observar como el estimador generado por EATBoosting no se sobreajusta a la nube de puntos, sino que intenta predecir la función de producción subyacente.

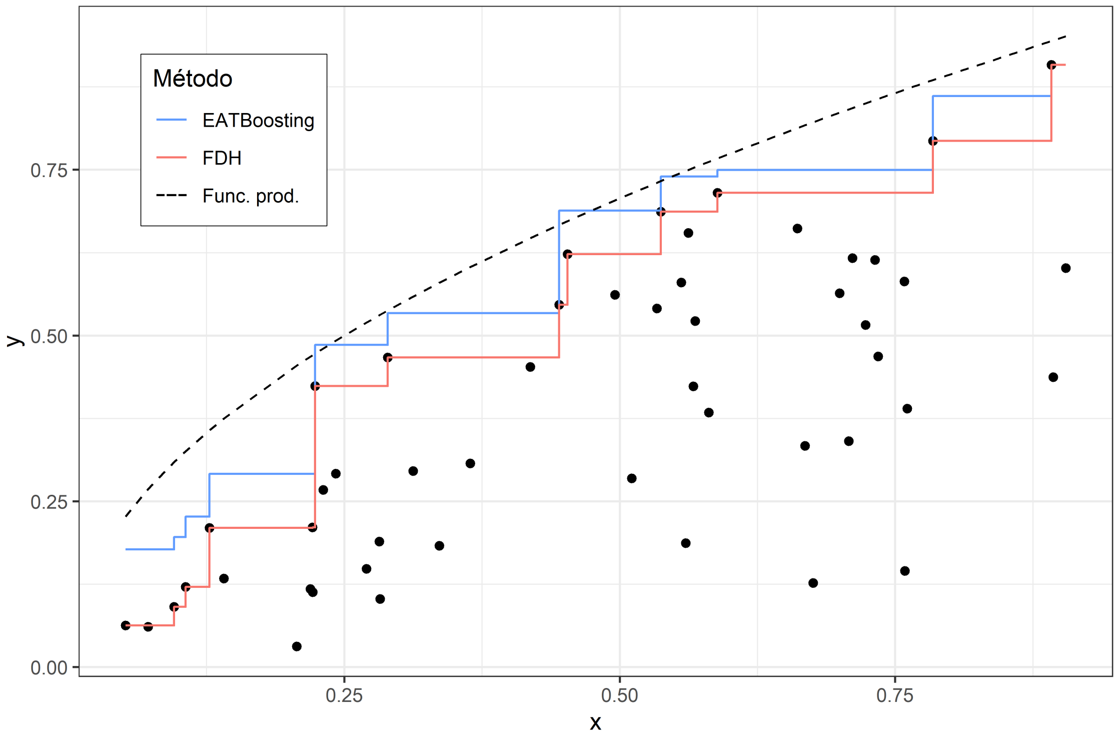

Figura 3.1: Ejemplo de la función de producción estimada por FDH y EATBoosting en un escenario mono-*input* mono-*output* con $y = x^{0{,}5}e^{-u}$, donde $x \sim Uni[0,1]$ y $u \sim |N(0, 0{,}4)|$.

La tecnología derivada del estimador de EATBoosting, $\hat{\Psi}_{EATBoosting}$, se define de la siguiente manera y es un estimador válido de la tecnología de producción, el cual no cumple el principio de mínima extrapolación.

$$\hat{\Psi}_{EATBoosting} := \{(\boldsymbol{x}, \boldsymbol{y}) \in \mathbb{R}_+^{m+s} : \boldsymbol{y} \leq \boldsymbol{f}_Q(\boldsymbol{x})\} \tag{3.1}$$

Proposición 5 (Proposición 6 A) $\hat{\Psi}_{EATBoosting}$ *satisface:*

(i) Libre disponibilidad de inputs y outputs.

(ii) Envoltura de los datos, esto es, $(\boldsymbol{x}_i, \boldsymbol{y}_i) \in \hat{\Psi}_{EATBoosting}$ *para todo* $i = 1, ..., n$.

(iii) $\hat{\Psi}_{FDH} \subseteq \hat{\Psi}_{EATBoosting}$.

Para demostrar el rendimiento del algoritmo EATBoosting, se llevó a cabo una evaluación comparativa entre la función de producción estimada por FDH y la estimada por dicho algoritmo. Véase el Apéndice A para más detalles sobre la experiencia computacional. De forma resumida, a partir de la típica función mono-*output* Cobb-Douglas en microeconomía, se realizaron diferentes simulaciones variando el tamaño de las muestras (50, 75 y 100 DMUs) y el número de *inputs* (6, 9, 12 y 15). En concreto, se realizaron 100 experimentos para cada configuración del escenario. Además, para evaluar los resultados, se utilizaron las métricas habituales de MSE y sesgo. Los resultados indican que tanto el MSE como el sesgo del estimador del método EATBoosting fueron claramente menores que para la técnica FDH en todas las configuraciones de los escenarios. En concreto, las mejoras relativas medias oscilaron entre el 61 % y el 75 % en el caso del MSE, y entre el 18 % y el 53 % en el caso del sesgo. Además, se detectó un patrón en los resultados, ya que dado un tamaño fijo de DMUs, a mayor número de *inputs*, mayor es el porcentaje de mejora. Estos resultados pueden explicarse por el hecho de que la frontera estimada por FDH se sitúa más cerca de los datos que la de EATBoosting debido al cumplimiento del principio de mínima extrapolación. En cambio, EATBoosting intenta estimar la función de producción real que se encuentra detrás de la generación de la muestra de datos.

Finalmente, se ilustra cómo funciona el nuevo enfoque en comparación con la técnica FDH estándar a través de un ejemplo empírico. En particular, utilizamos el conjunto de datos propuesto por Juo et al. (2015), donde se obtuvieron datos de 31 bancos taiwaneses para el año 2010. En concreto, se consideran 3 *inputs* (fondos financieros, mano de obra y capital físico) y 1 *output* (ingresos), y se compara el valor de la medida radial orientación *output*. En este caso, al ser un escenario mono-*output*, este valor se puede calcular para el enfoque EATBoosting directamente como $\phi^{EATBoosting} = f_Q(\boldsymbol{x})/y$. Respecto a los resultados, la técnica FDH muestra que todos los bancos son técnicamente eficientes, con tres excepciones. En contraposición, según EATBoosting solo 11 de los 31 bancos son técnicamente eficientes. De este modo, el algoritmo EATBoosting parece mostrar un mayor poder discriminatorio, permitiendo discriminar entre las DMUs realmente eficientes e ineficientes.

3.2. Medidas de eficiencia bajo la tecnología EATBoosting

Llegados a este punto, conocemos la expresión del estimador de la tecnología generada por la función de producción estimada por el algoritmo EATBoosting. Sin embargo, en la literatura se han definido muchas medidas de eficiencia técnica y necesitamos saber cómo implementar cada una de ellas bajo el nuevo enfoque. En esta sección, introducimos una forma de calcular cualquier medida de eficiencia técnica utilizando la tecnología estimada mediante EATBoosting como base.

En la estructura de cada árbol EAT individual en la iteración q del Algoritmo 2, cada nodo hoja j, $j = 1, ..., J$, se define por el cumplimiento de una serie de condiciones en el espacio de los *inputs* del tipo $\{x^{(p)} < s^{(p)}\}$ o $\{x^{(p)} \geq s^{(p)}\}$, donde p es el *input* utilizado para realizar la división de dicho nodo y $s^{(p)} \in S^{(p)}$, siendo $S^{(p)}$ el conjunto de todos los valores observados para la variable j en $\aleph$. Por tanto, tras ejecutar todas las divisiones y obtener la estructura del árbol final para el árbol correspondiente al paso q, el soporte de dicho nodo hoja tiene el formato: $R_{jq} = \left\{\boldsymbol{x} \in \mathbb{R}_{+}^{m} : a_{jq}^{(p)} \leq x^{(p)} < b_{jq}^{(p)}, p = 1, ..., m\right\}$. Nótese que el parámetro $a_{jq}^{(p)}$ puede ser 0, mientras que el parámetro $b_{jq}^{(p)}$, $+\infty$. Tras ejecutar el algoritmo EATBoosting, el espacio de *inputs* se divide en subconjuntos disjuntos resultantes de la intersección de todas las particiones del espacio de *inputs* asociadas a cada árbol individual EAT. Esta partición final se compone de soportes tales que $\hat{R}_{j_1,...,j_Q} = \bigcap_{q=1}^{Q} R_{j_q q}, \forall j_q = 1, ..., J, \forall q = 1, ..., Q$. Dicha intersección puede reescribir en el mismo formato que cualquier soporte de un árbol individual.

Proposición 6 (Proposición 1 B)

$$\hat{R}_{j_1,...,j_Q} = \left\{\boldsymbol{x} \in \mathbb{R}_{+}^{m} : a_{j_1,...,j_Q}^{(p)} \leq x^{(p)} < b_{j_1,...,j_Q}^{(p)}, p = 1, ..., m\right\}$$

con $a_{j_1,...,j_Q}^{(p)} = \max\limits_{1 \leq q \leq Q} \left\{a_{j_q q}^{(p)}\right\}$ *y* $b_{j_1,...,j_Q}^{(p)} = \min\limits_{1 \leq q \leq Q} \left\{b_{j_q q}^{(p)}\right\}$.

Gracias a esta definición de $\hat{R}_{j_1,...,j_Q}$, se puede definir una muestra de datos "virtual" (con vectores de *inputs* y *outputs* no necesariamente observados) donde el vector de *inputs* es $\hat{\boldsymbol{a}}_{j_1 \cdots j_Q}$, es decir, el vector correspondiente a la esquina inferior de los soportes, y el vector de *outputs* es $\hat{\boldsymbol{f}}\left(\hat{\boldsymbol{a}}_{j_1 \cdots j_Q}\right)$, es decir, el vector correspondiente a la estimación del predictor EATBoosting para el vector $\hat{\boldsymbol{a}}_{j_1 \cdots j_Q}$. Formalmente, esta muestra es $\left\{\left(\hat{\boldsymbol{a}}_{j_1 \cdots j_Q}, \hat{\boldsymbol{f}}\left(\hat{\boldsymbol{a}}_{j_1 \cdots j_Q}\right)\right)\right\}, \forall j_q = 1, ..., J_q, \forall q = 1, ..., Q$. En términos de esta muestra "virtual", se puede definir la tecnología estimada por el algoritmo EATBoosting mediante la siguiente expresión.

$$\begin{aligned}\hat{\Psi}_{EATBoosting} = \Big\{ (\boldsymbol{x}, \boldsymbol{y}) \in R_{+}^{m+s} : \exists \left(j_1, ..., j_Q\right), j_q = 1, ..., J_q \, , q = 1, ..., Q, \\ \text{tal que}\, \boldsymbol{y} \leq \hat{\boldsymbol{f}}\left(\hat{\boldsymbol{a}}_{j_1 ... j_Q}\right), \boldsymbol{x} \geq \hat{\boldsymbol{a}}_{j_1 ... j_Q} \Big\}\end{aligned} \tag{3.2}$$

El resultado clave para poder calcular cualquier medida de eficiencia técnica utilizando como base la tecnología estimada por el algoritmo EATBoosting es que la tecnología estimada por FDH usando la muestra "virtual" coincide con la tecnología estimada por el algoritmo EATBoosting.

Proposición 7 (Proposición 2 B) *La tecnología estimada por FDH,* $\hat{\Psi}_{FDH}$*, usando la muestra de datos* $\left\{ \left(\hat{\boldsymbol{a}}_{j_1 ... j_Q}, \hat{\boldsymbol{f}}\left(\hat{\boldsymbol{a}}_{j_1 ... j_Q}\right) \right) \right\}, \forall j_q = 1, ..., J_q, \forall q = 1, ..., Q,$ *coincide con la tecnología estimada por EATBoosting,* $\hat{\Psi}_{EATBoosting}$.

De este modo, por analogía con el Ecuación 2.5, el siguiente problema de optimización puede usarse para calcular la medida radial orientación *output* usando como base la tecnología estimada por EATBoosting.

$$\begin{aligned} \phi^{EATBoosting}(\boldsymbol{x}_k, \boldsymbol{y}_k) = & \max \phi_k \\ & s.t. \\ & \sum_{j_1=1}^{J} ... \sum_{j_Q=1}^{J} \lambda_{j_1 ... j_Q} \hat{a}_{j_1 ... j_Q}^{(p)} \leq x_k^{(p)}, p = 1, ..., m, \\ & \sum_{j_1=1}^{J} ... \sum_{j_Q=1}^{J} \lambda_{j_1 ... j_Q} \hat{f}^{(r)}(\hat{\boldsymbol{a}}_{j_1 ... j_Q}) \geq \phi_k y_k^{(r)}, r = 1, ..., s, \\ & \sum_{j_1=1}^{J} ... \sum_{j_Q=1}^{J} \lambda_{j_1 ... j_Q} = 1, \\ & \lambda_i \in \{0, 1\}, j_q = 1, ..., J, q = 1, ..., Q \end{aligned} \tag{3.3}$$

Como vemos, para obtener la Ecuación 3.3, en lugar de introducir en la Ecuación 2.5 las DMUs que constituyen la muestra de datos observados, ahora se utilizan las unidades "virtuales". Del mismo modo, cualquier medida de eficiencia definida bajo el enfoque FDH puede ser también calculada bajo el enfoque EATBoosting. En el Apéndice B se incluyen los problemas de optimización asociados a las medidas radiales orientación *input* y *output* (Banker et al., 1984), las medidas Russell orientación *input* y *output* (Färe & Lovell, 1978), la Función de Distancia Direccional (Directional Distance Function, DDF) (Chambers et al., 1998), y la Medida Basada en Slacks (Slacks-Based Measure, SBM) (Tone, 2001).

No obstante, el número de variables de decisión de los programas de optimización asociados es ahora mucho mayor. En concreto, en lugar de haber n variables de decisión λ como ocurre en el enfoque FDH (tantas como DMUs), los programas de optimización basados en el enfoque EATBoosting tendrán J^Q variables. Para hacernos una idea, si consideramos $J = 6$ y $Q = 10$, el número de variables de decisión λ será superior a 60 millones, tantas como soportes finales se han generado. Esto es extremadamente complejo a nivel computacional debido a los requisitos de memoria y de tiempo de computación del programa. Por este motivo, se propone una aproximación heurística a las medidas, donde en la muestra "virtual" los valores correspondientes a los soportes son reemplazados por los perfiles de *inputs* que fueron observados en la muestra de datos original. Es decir, la muestra $\left\{\left(\hat{\boldsymbol{a}}_{j_1...j_Q}, \hat{\boldsymbol{f}}\left(\hat{\boldsymbol{a}}_{j_1...j_Q}\right)\right)\right\}, \forall j_q = 1, ..., J_q, \forall q = 1, ..., Q$ es reemplazada por $\left\{\left(\boldsymbol{x}_i, \hat{\boldsymbol{f}}\left(\boldsymbol{x}_i\right)\right)\right\}, \forall i = 1, ..., n$. En este sentido, se podría determinar una aproximación heurística del valor óptimo de la Ecuación 3.3 mediante la Ecuación 3.4.

$$
\begin{aligned}
\phi_{HEU}^{EATBoosting}(\boldsymbol{x}_k, \boldsymbol{y}_k) = \max\, & \phi_k \\
s.t.\ & \\
& \sum_{i=1}^{n} \lambda_i x_i^{(p)} \leq x_k^{(p)}, p = 1, ..., m, \\
& \sum_{i=1}^{n} \lambda_i \hat{f}^{(r)}(\boldsymbol{x}_i) \geq \phi_k y_k^{(r)}, r = 1, ..., s, \\
& \sum_{i=1}^{n} \lambda_i = 1, \lambda_i \in \{0, 1\}, i = 1, ..., n
\end{aligned}
\tag{3.4}
$$

Obsérvese que en este caso, independientemente de los valores de los hiperparámetros J y Q, el número de variables de decisión λ será siempre n, es decir, el tamaño de la muestra original. Esta simplificación de la Ecuación 3.3 reduce drásticamente la complejidad del modelo y tiene claras implicaciones sobre el tiempo de cálculo y los requisitos de memoria.

Para demostrar la calidad de la medidas heurísticas usando como base la técnica EATBoosting en comparación con la basada en FDH se realizan distintas simulaciones computacionales. Estas simulaciones no se pueden extender al enfoque exacto debido a limitaciones de tiempo y memoria previamente comentadas. Para más detalles sobre esta comparativa, consulte el Apéndice B.

De forma resumida, se propone un primer escenario con 2 *inputs* y 2 *outputs*, siguiendo el enfoque de Perelman y Santín (2009), en el que se varía el porcentaje de DMUs que realmente se encuentran en la frontera de producción (0 %, 10 % y 25 %) y el número de DMUs (50, 100, 200, 300, 400 y 500). Para cada configuración, se ha calculado la diferencia relativa entre el valor de la medida radial orientación *output* (*score*) basada en FDH y el valor de la medida heurística basada en EATBoosting con respecto al valor de eficiencia del output ideal en términos de MSE y sesgo para 100 experimentos. Las mejoras relativas medias en MSE del enfoque EATBoosting respecto al FDH variaron entre 7 % y 74 %, mientras que la mejora media en el sesgo varió entre 6 % y el 48 %, dependiendo del tamaño de la muestra y del número de DMUs en la frontera simulada.

Además, también se propone en un segundo escenario con 3 *inputs* y 3 *outputs*, siguiendo el enfoque de Schaefer y Clermont (2018), con la misma configuración de la frontera y de número de DMUs que en el primer escenario. Al igual que en el escenario anterior, EATBoosting supera a FDH en términos de MSE y sesgo en todas las configuraciones. En este caso, la mejora en MSE varió entre 22 % y 73 %, mientras que la mejora en el sesgo varió entre 13 % y el 83 %. Además, mientras que en FDH la desviación estándar del MSE aumenta cuando aumenta el número de DMUs, en el caso de EATBoosting parece disminuir. De este modo, al igual que ocurría en los escenario mono-*output* del Apéndice A, la frontera estimada por DEA se sitúa más cerca de los datos que la estimada por EATBoosting, intentando estimar la función de producción subyacente.

Sin embargo, es importante destacar una limitación evidente de esta nueva metodología en comparación con las técnicas estándar. Los resultados muestran una clara diferencia entre el tiempo de cálculo entre FDH y EATBoosting. De hecho, mientras que para el escenario más complejo con 3 *inputs*, 3 *outputs* y 500 observaciones, el tiempo medio asociado a FDH para obtener los valores de eficiencia fue alrededor de 10 segundos, el tiempo medio asociado a EATBoosting fue algo menos de 3 minutos.

Para finalizar, se aplica el algoritmo EATBoosting a dos bases de datos reales para evaluar y comparar los métodos exactos y heurísticos de medición de eficiencia basados en EATBoosting. Como se ha explicado anteriormente, no es técnicamente posible comparar las diferencias entre el enfoque exacto y el enfoque heurístico, debido a la complejidad computacional, mediante múltiples simulaciones computacionales. Sin embargo, la comparación es posible a través de ejemplos numéricos concretos. En el primero de los ejemplos empíricos, se usa el mismo conjunto de datos sobre bancos empleado en el Apéndice A, pero considerando 3 *inputs* (fondos financieros, mano de obra y capital físico) y 2 *outputs* (ingresos de la inversión financiera e ingresos de los préstamos). Para dichos bancos, se calculan las medidas radiales orientación *input* y *output*, las medidas Russell orientación *input* y *output*, la DDF y la SBM. Respecto a los resultados, según el enfoque FDH todas las DMUs son técnicamente eficientes excepto una, independientemente de la medida utilizada. En contraposición, tanto las medidas heurísticas como exactas basadas en el algoritmo EATBoosting son capaces de captar una mayor distribución en los valores de eficiencia, mostrando una mayor capacidad de discriminación entre bancos eficientes e ineficientes. De hecho, los métodos exactos y heurísticos presentan valores idénticos en la mayoría de medidas. Además, se aplicó el test de Li (Simar y Zelenyuk, 2006) para verificar si las diferencias entre los valores de eficiencia de los diferentes enfoques eran estadísticamente significativas. Los p-valores indican que los métodos EATBoosting (heurístico y exacto) generan valores de eficiencia claramente distinguibles del método FDH. Además, solo se observan diferencias entre las medidas heurísticas y exactas para las medidas orientación *input* y la SBM [1]. De este modo, aparentemente el enfoque heurístico es una buena aproximación de las medidas exactas

En el segundo ejemplo empírico, se aborda de forma directa el problema de la maldición de la dimensionalidad. Este problema está relacionado con la falta de poder discriminatorio a medida que aumenta la dimensionalidad del espacio de producción. Este inconveniente es especialmente evidente en el caso de FDH cuando la proporción del número de variables (*inputs* y *outputs*) respecto al número de DMUs es baja. Como resultado, un gran número de DMUs se consideran técnicamente eficientes. Para ilustrar este problema, recurrimos a un conjunto de datos que incluye las 15 mejores ciudades de Estados Unidos según la revista Fortune, utilizado en Charles et al. (2019). Para ello, se utiliza información sobre seis *inputs* y seis *outputs* para calcular la medida radial orientación *output*. Analizando los resultados, la técnica FDH clasifica todas las DMUs como técnicamente eficientes. En contraposición, tanto el enfoque exacto como heurístico basado en EATBoosting

[1]Simar y Zelenyuk (2006) introdujeron su adaptación del test de Li para medidas de eficiencia orientación *output*, en la que los valores de eficiencia están en el rango $[1, +\infty)$. Se adaptó el test a las medidas cuyos valores están en el rango $[0, 1]$ realizando las transformaciones de rango necesarias.

coinciden en que solo 3 de las 15 ciudades son técnicamente eficientes. De hecho, ambos enfoques obtienen puntuaciones de eficiencia muy similares para todas las ciudades. De nuevo, el nuevo enfoque basado en el algoritmo EATBoosting sí que es capaz de distinguir entre ciudades técnicamente eficientes e ineficientes, pudiéndose considerar como un posible remedio para la maldición de la dimensionalidad en la medición de eficiencia no paramétrica.

3.3. Paquete en R: `boostingDEA`

En esta sección se presenta el paquete de R llamado `boostingDEA`, el cual permite estimar fronteras de producción y hacer predicciones de la producción ideal en el contexto del análisis de eficiencia no paramétrico mediante los modelos estándar de DEA y FDH y, además, modelos basados en boosting. La finalidad de este paquete es facilitar el uso real de la técnica desarrollada en esta Tesis Doctoral por parte de otros investigadores, gestores y responsables políticos. El paquete está disponible en CRAN, donde además se incluye material suplementario como conjuntos de datos o viñetas para ilustrar su funcionamiento. Así mismo, el código del paquete está alojado en un repositorio de GitHub para que los usuarios lo modifiquen, amplíen y adapten según sus necesidades.

En concreto, `boostingDEA` incluye funciones para crear modelos de eficiencia bajo tres enfoques diferentes: DEA, FDH y EATBoosting. Además, en este paquete se incluyen funciones para calcular diferentes medidas de eficiencia técnica definidas en la literatura utilizando los diferentes enfoques. En particular, se incluyen las medidas radiales orientación *input* y *output* (Banker et al., 1984), las medidas Russell orientación *input* y *output* (Färe & Lovell, 1978), la Función de Distancia Direccional (*Directional Distance Function*, DDF) (Chambers et al., 1998), la Medida Aditiva Ponderada (*Weighted Additive Measure*, WAM) (Lovell & Pastor, 1995) y la Medida Basada en Slacks (Slacks-Based Measure, SBM) (Tone, 2001), equivalente a la *Enhanced Russell Graph Measure* (ERG) (Pastor et al., 1999). Las principales funciones incluidas en el paquete `boostingDEA` se resumen en la tabla 3.1, donde la primera columna contiene el nombre de la función y la segunda ofrece una breve descripción de la misma. Respecto a la información sobre las DMUs, en el paquete estas se manejan a través de estructuras `matrix` y/o `data.frame`, indicando los índices de las *inputs* y *outputs* a través de `vector`.

Dentro del paquete, el modelo DEA se calcula usando la función `DEA(data,x,y)`. Esta función requiere como parámetros de entrada la información sobre las DMUs

Función	**Descripción**
`DEA`	Genera un modelo DEA
`FDH`	Genera un modelo FDH
`EATBoost`	Genera un modelo EATBoosting
`bestEATBoost`	Calcula el error cuadrático medio (MSE) asociado a un conjunto de modelos EATBoosting, los cuales son especificados por el usuario mediante un *grid* de hiperparámetros.
`predict`	Método genérico. Calcula el *ouput* ideal dado un perfil de *inputs* bajo un modelo especificado.
`efficiency`	Calcula los valores de las medidas de eficiencia bajo un modelo especificado. Las medidas incluidas son: • `rad.out`: Medida radial orientación *output*. • `rad.in`: Medida radial orientación *input*. • `Russell.out`: Medida Russell orientación *output*. • `Russell.in`: Medida Russell orientación *input*. • `DDF`: Directional Distance Function. • `WAM`: Weighted Additive Model. • `ERG`: Slacks-Based measure.

Tabla 3.1: Funciones incluidas en el paquete `boostingDEA`

a evaluar (`data`) y los índices de los perfiles de *inputs* y *outputs* (`x` e `y`, respectivamente). De forma similar, el modelo FDH se calcula usando la función `FDH(data,x,y)`.

```
# Creates a DEA model with 3 inputs and 1 output
R> DEAmodel <- DEA(data = banks, x = 1:3, y = 6)
```

```
# Creates a DEA model with 3 inputs and 1 output
R> FDHmodel <- FDH(data = banks, x = 1:3, y = 6)
```

Por su parte, el modelo EATBoosting se calcula mediante la función `EATBoost(data, x, y, num.iterations, num.leaves, learning.rate)`. Esta función requiere aparte de la información sobre las DMUs, el conjunto de hiperparámetros que controlan el rendimiento del algoritmo. Estos parámetros son:

- `num.iterations`, que hace referencia al número de iteraciones que realiza el algoritmo, Q.
- `num.leaves`, que limita el número de nodos hoja de cada árbol EAT en cada iteración, J.
- `learning.rate`, que es la tasa de aprendizaje, v.

```
# Creates an EATBoosting model with 3 inputs and 2 outputs
R> EATBoostModel <- EATBoost(
  data = banks, x = 1:3, y = 4:5,
  num.iterations = 4,
  num.leaves = 4,
  learning.rate = 0.6
)
```

Uno de los aspectos clave del algoritmo EATBoosting es el ajuste de los hiperparámetros. Para tratar de encontrar el mejor valor para los hiperparámetros, podemos recurrir a un *grid* de valores de parámetros que pueden probarse mediante muestras de entrenamiento y de *test* en una proporción especificada por el usuario. Para ello, podemos recurrir a la función `bestEATBoost(training,test,x,y,num.iterations,learning.rate, num.leaves,verbose)`. Esta función recibe un vector de valores para cada hiperparámetro, en vez de un único valor. Además, mediante el argumento `verbose` podemos especificar si queremos "ver" el proceso de entrenamiento. La función devuelve cada configuración ordenada según su MSE.

```
# Search of best hyperparameters for an EATBoosting model
R> grid_EATBoost <- bestEATBoost(
  training = training, test = test,
  x = 1:3, y = 4:5,
  num.iterations = c(5, 6, 7),
  learning.rate = c(0.4, 0.5, 0.6),
  num.leaves = c(6, 7, 8),
  verbose = FALSE
)
# Best EATBoosting model
R> EATBoost_best <- EATBoost(
  data = banks, x = 1:3, y = 4:5,
  num.iterations = grid_EATBoost[1, "num.iterations"],
  learning.rate = grid_EATBoost[1, "learning.rate"],
  num.leaves = grid_EATBoost[1, "num.leaves"]
)
```

Una de las ventajas del algoritmo EATBoosting es que puede utilizarse para tomar decisiones cuando se trabaja con datos que no están incluidos en el conjunto de datos observados. Es decir, en vez de centrarnos en calcular la eficiencia técnica de los datos presentes en la muestra, podemos utilizar la tecnología estimada para hacer predicciones sobre DMUs fuera de la muestra. Por ejemplo, dada una empresa que desarrolla su actividad comercial en un ámbito previamente estudiado, a dicha empresa le puede interesar ver cómo variaría su producción ideal al modificar el valor de ciertos *inputs* (aumentar cierto *input* si se desea invertir o ver en que *input* se pueden ahorrar costes de forma que afecte lo mínimo posible). Esto se puede hacer en el paquete a través de función `predict(object, newdata, x)`, donde `object` hace referencia a los modelos DEA, FDH y EATBoosting, `newdata` a un conjunto de DMUs, que puede ser el mismo que el utilizado para crear los modelo o no, y `x` a los índices del perfil de *inputs*.

```
# EATBoosting prediction
R> predict(object = EATBoost_best, newdata = banks, x = 1:3)
      Financial.investments_pred Loans_pred
   1                    12364.87   102624.5
   2                   904580.00  2091100.0
   3                   691348.93  1549138.7
   4                   122533.01   193094.5
   5                   822648.85  1882858.7
   6                   822648.85  1882858.7
```

Finalmente, dentro del paquete, también se puede calcular diferentes medidas de eficiencia técnica gracias a la función `efficiency(model,measure,data,x,y,...)`. Esta función recibe como parámetros de entrada bajo qué enfoque queremos calcular la medida (`model`, que puede ser DEA, FDH y EATBoosting), la medida concreta a calcular (`measure`) y el conjunto de datos sobre el que queremos dichos valores de eficiencia técnica (`data`, `x`, `y`). Además, dependiendo de la medida, se pueden necesitar parámetros adicionales, como por ejemplo el vector de dirección para la Función de Distancia Direccional (*Directional Distance Function*, DDF) o el vector de pesos en la Medida Aditiva Ponderada (*Weighted Additive Measure*, WAM). Así mismo, dentro del modelo EATBoosting, también es posible especificar si se desea calcular las medidas exactas o bajo el enfoque heurístico.

```
# Radial output measure
R> efficiency(model = EATBoost_best, measure = "rad.out",
              heuristic = FALSE, data = banks, x = 1:3,y = 4:5)
                                   EATBoost.rad.out
Export-Import Bank                         1.251579
Bank of Taiwan                             1.000000
Taipei Fubon Bank                          1.761439
Bank of Kaohsiung                          1.184237
Land Bank                                  1.103045
Cooperative Bank                           1.046176
```

Capítulo 4

Conclusiones y trabajos futuros

La presente Tesis Doctoral muestra cómo estimar fronteras de producción y medir eficiencia técnica utilizando como base la tecnología estimada mediante la adaptación del algoritmo Gradient Tree Boosting, denominada EATBoosting. Las principales contribuciones de la misma se han expuesto en un estilo directo en aras de la brevedad y con la intención de evitar redundancias innecesarias con los apéndices que contienen las publicaciones completas que conforman esta Tesis. En concreto, se ha probado que esta nueva metodología basada en el aprendizaje automático mejora el enfoque FDH no paramétrico estándar, sin dejar de cumplir la libre disponibilidad de *inputs* y *outputs* y proporcionando estimaciones que envuelven la nube de datos desde arriba. Además, se han introducido los programas de optimización lineales que permiten determinar medidas de eficiencia bien conocidas en la literatura. Así mismo, también se ha introducido un enfoque heurístico a estas medidas para hacer frente a limitaciones tanto en tiempo computacional como en requisitos de memoria de las mismas. Finalmente, se ha intentado facilitar su uso a través de una implementación en software libre accesible para la comunidad científica y profesional mediante el paquete en R `boostingDEA`.

La principal ventaja del nuevo enfoque reside en su capacidad para abordar el problema habitual del sobreajuste, que caracteriza a los métodos estándar como DEA o FDH. A diferencia de estas técnicas estándar, nuestra metodología no subestima sistemáticamente la ineficiencia real de las DMUs, por lo que funciona más como una herramienta inferencial que como una mera herramienta descriptiva. En consecuencia, permite un mayor poder discriminatorio, lo que conduce a una identificación más precisa de las ineficiencias. Además, nuestro enfoque ofrece una solución al problema de la maldición de la dimensionalidad, especialmente evidente cuando la relación entre el número de DMU y el número de variables es baja. La aplicación de EATBoosting en estos casos permite realizar un análisis de eficiencia más sólido y preciso.

Respecto a las vías de investigación futuras, hay varios problemas pendientes directamente relacionados con los logros de esta Tesis. Primero, aunque los resultados son alentadores, se debería desarrollar un análisis en mayor profundidad sobre las diferencias entre el enfoque heurístico y el exacto, así como sobre el uso de EATBoosting como posible remedio para la maldición de la dimensionalidad en la medición de eficiencia no paramétrica. Además, sería interesante proponer el uso de otra clase de modelos que generen estimaciones mediante funciones lineales a trozos como modelos base dentro del proceso del algoritmo, de modo que el estimador resultante se pueda comparar directamente con DEA.

En términos de expansión metodológica, otras implementaciones de las técnicas de *boosting* como Stochastic Gradient Boosting (Friedman, 2002) o XGBoost (eXtreme Gradient Boosting) (Chen y Guestrin, 2016) podrían considerarse. Así mismo, otra área de investigación futura podría centrarse en la integración de EATBoosting con otras técnicas de aprendizaje automático, como las redes neuronales profundas (Deep Learning), para explorar la mejora en la capacidad predictiva y la robustez del modelo. Esta integración podría ofrecer nuevas perspectivas para el tratamiento de grandes volúmenes de datos y estructuras de datos complejas.

Por otra parte, se debería ampliar la aplicación del nuevo enfoque a más bases de datos reales en diferentes contextos empíricos, con el objetivo de validar el buen rendimiento del modelo en la práctica. Del mismo modo, podrían considerarse escenarios con presencia de *inputs* y/o *outputs* no deseados, los cuales requieren un tratamiento específico. Así mismo, también sería beneficioso investigar la aplicabilidad de EATBoosting en la evaluación de eficiencia bajo condiciones de incertidumbre, utilizando enfoques de programación robusta o análisis estocástico. Esto permitiría una mejor comprensión del comportamiento del modelo bajo variabilidad e incertidumbre en los datos, proporcionando así una herramienta más robusta para la toma de decisiones en ambientes reales.

Otra línea de investigación futura podría involucrar la comparación del desempeño de EATBoosting con otros enfoques emergentes en la medición de eficiencia, como los métodos bayesianos y los modelos de frontera estocástica. Estas comparaciones pueden proporcionar una perspectiva más amplia sobre la eficacia relativa de EATBoosting y ayudar a identificar sus fortalezas y limitaciones en diferentes contextos de aplicación.

Finalmente, la incorporación de técnicas de visualización avanzada para representar los resultados de EATBoosting podría ser una línea de investigación valiosa. La visualización efectiva de los resultados de eficiencia puede facilitar una mejor interpretación y comunicación de los hallazgos, especialmente en contextos multidimensionales y de alta complejidad.

En resumen, las posibilidades de expansión y profundización en el estudio y aplicación de EATBoosting son amplias y variadas, abarcando desde mejoras metodológicas hasta aplicaciones empíricas en diversos contextos, pasando por la integración con otras técnicas de aprendizaje automático y la exploración bajo condiciones de incertidumbre. Estas líneas de investigación futura no solo tienen el potencial de enriquecer el campo de la medición de eficiencia, sino también de proporcionar herramientas más robustas y precisas para la toma de decisiones en una amplia gama de aplicaciones prácticas.

Bibliografía

Ang, F., Kerstens, K., & Sadeghi, J. (2023). Energy productivity and greenhouse gas emission intensity in Dutch dairy farms: A Hicks–Moorsteen by-production approach under non-convexity and convexity with equivalence results. *Journal of Agricultural Economics*, *74*(2), 492-509.

Aparicio, J., Esteve, M., & Kapelko, M. (2023). Measuring dynamic inefficiency through machine learning techniques. *Expert Systems with Applications*, *228*, 120417.

Aparicio, J., Esteve, M., Rodriguez-Sala, J. J., & Zofio, J. L. (2021). The estimation of productive efficiency through machine learning techniques: Efficiency analysis trees. En *Data-Enabled Analytics: DEA for Big Data* (pp. 51-92). Springer.

Banker, R. D., Charnes, A., & Cooper, W. W. (1984). Some models for estimating technical and scale inefficiencies in data envelopment analysis. *Management science*, *30*(9), 1078-1092.

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). Classification and regression trees. *Taylor & Francis.*

Breiman, L. (2001). Random forests. *Machine learning*, *45*, 5-32.

Chambers, R. G., Chung, Y., & Färe, R. (1998). Profit, directional distance functions, and Nerlovian efficiency. *Journal of optimization theory and applications*, *98*, 351-364.

Charles, V., Aparicio, J., & Zhu, J. (2019). The curse of dimensionality of decision-making units: A simple approach to increase the discriminatory power of data envelopment analysis. *European Journal of Operational Research*, *279*(3), 929-940.

Charnes, A., Cooper, W. W., & Rhodes, E. (1978). Measuring the efficiency of decision making units. *European journal of operational research*, *2*(6), 429-444.

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785-794.

Daouia, A., Noh, H., & Park, B. U. (2016). Data envelope fitting with constrained polynomial splines. *Journal of the Royal Statistical Society: Series B: Statistical Methodology*, 3-30.

Deprins, D., Simar, L., & Tulkens, H. (1984). *Measuring labor-efficiency in post offices*. LIDAM Reprints CORE.

España, V. J., Aparicio, J., Barber, X., & Esteve, M. (2024). Estimating production functions through additive models based on regression splines. *European Journal of Operational Research*, *312*(2), 684-699.

Esteve, M., Aparicio, J., Rabasa, A., & Rodriguez-Sala, J. J. (2020). Efficiency analysis trees: A new methodology for estimating production frontiers through decision trees. *Expert systems with applications*, *162*, 113783.

Esteve, M., Aparicio, J., Rodriguez-Sala, J. J., & Zhu, J. (2022). Random Forests and the measurement of super-efficiency in the context of Free Disposal Hull. *European Journal of Operational Research.*

Färe, R., & Lovell, C. K. (1978). Measuring the technical efficiency of production. *Journal of Economic theory*, *19*(1), 150-162.

Farrell, M. J. (1957). The Measurement of Productive Efficiency. *Journal of the Royal Statistical Society. Series A (General)*, *120*(3), 253-290.

Ferreira, A. J., & Figueiredo, M. A. (2012). Boosting algorithms: A review of methods, theory, and applications. *Ensemble machine learning: Methods and applications*, 35-85.

Friedman, J. H. (1999). Stochastic Gradient Boosting. *Preliminary Report Stanford University.*

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 1189-1232.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational statistics & data analysis*, *38*(4), 367-378.

Guerrero, N. M., Aparicio, J., & Valero-Carreras, D. (2022). Combining Data Envelopment Analysis and Machine Learning. *Mathematics*, *10*(6), 909.

Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2017). *The elements of statistical learning: data mining, inference, and prediction* (Vol. 2). Springer.

Juo, J.-C., Fu, T.-T., Yu, M.-M., & Lin, Y.-H. (2015). Profit-oriented productivity change. *Omega*, *57*, 176-187.

Kerstens, K., Sadeghi, J., Toloo, M., & Van de Woestyne, I. (2022). Procedures for ranking technical and cost efficient units: With a focus on nonconvexity. *European journal of operational research*, *300*(1), 269-281.

Kerstens, K., & Van de Woestyne, I. (2021). Cost functions are nonconvex in the outputs when the technology is nonconvex: convexification is not harmless. *Annals of Operations Research*, *305*(1), 81-106.

Lovell, C. K., & Pastor, J. T. (1995). Units invariant and translation invariant DEA models. *Operations research letters*, *18*(3), 147-151.

Moragues, R., Aparicio, J., & Esteve, M. (2023). Measuring technical efficiency for multi-input multi-output production processes through OneClass Support Vector Machines: a finite-sample study. *Operational Research*, *23*(3), 47.

Olesen, O., & Ruggiero, J. (2022). The hinging hyperplanes: An alternative nonparametric representation of a production function. *European Journal of Operational Research*, *296*(1), 254-266. https://doi.org/https://doi.org/10.1016/j.ejor.2021.03.054

Orea, L., & Zofío, J. L. (2019). Common methodological choices in nonparametric and parametric analyses of firms' performance. *The Palgrave handbook of economic performance analysis*, 419-484.

Parmeter, C. F., & Racine, J. S. (2013). Smooth constrained frontier analysis. *Recent Advances and Future Directions in Causality, Prediction, and Specification Analysis: Essays in Honor of Halbert L. White Jr*, 463-488.

Pastor, J. T., Ruiz, J. L., & Sirvent, I. (1999). An enhanced DEA Russell graph efficiency measure. *European journal of operational research*, *115*(3), 596-607.

Perelman, S., & Santín, D. (2009). How to generate regularly behaved production data? A Monte Carlo experimentation on DEA scale efficiency measurement. *European Journal of Operational Research*, *199*(1), 303-310.

Schaefer, J., & Clermont, M. (2018). Stochastic non-smooth envelopment of data for multi-dimensional output. *Journal of Productivity Analysis*, *50*(3), 139-154.

Simar, L., & Wilson, P. W. (1998). Sensitivity analysis of efficiency scores: How to bootstrap in nonparametric frontier models. *Management science*, *44*(1), 49-61.

Simar, L., & Wilson, P. W. (2000). A general methodology for bootstrapping in non-parametric frontier models. *Journal of applied statistics*, *27*(6), 779-802.

Simar, L., & Zelenyuk, V. (2006). On testing equality of distributions of technical efficiency scores. *Econometric Reviews*, *25*(4), 497-522.

Tone, K. (2001). A slacks-based measure of efficiency in data envelopment analysis. *European journal of operational research*, *130*(3), 498-509.

Tsionas, M. G. (2022). Efficiency estimation using probabilistic regression trees with an application to Chilean manufacturing industries. *International Journal of Production Economics*, 108492.

Valero-Carreras, D., Aparicio, J., & Guerrero, N. M. (2021). Support vector frontiers: A new approach for estimating production functions through support vector machines. *Omega*, *104*, 102490.

Valero-Carreras, D., Aparicio, J., & Guerrero, N. M. (2022). Multi-output Support Vector Frontiers. *Computers & Operations Research*, *143*, 105765.

Apéndice A

Gradient tree boosting and the estimation of production frontiers

Expert Systems With Applications 214 (2023) 119134

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Gradient tree boosting and the estimation of production frontiers

Maria D. Guillen, Juan Aparicio [*], Miriam Esteve

Center of Operations Research (CIO). Miguel Hernandez University of Elche (UMH), Elche, Alicante 03202, Spain

A R T I C L E I N F O

Keywords:
Data envelopment analysis
Free Disposal Hull
Boosting
Technical efficiency

A B S T R A C T

In production theory and engineering, a topic of interest is the determination of technical efficiency of firms from the estimation of a technology. By definition, a technology must satisfy a set of micro-economic postulates. Likewise, a valid estimator of a technology should meet the same set of axioms. In this paper, for the first time, we adapt the Gradient Tree Boosting algorithm with the objective of estimating production technologies, satisfying the required theoretical conditions. The new approach shares similarities with the standard Free Disposal Hull (FDH) methodology, but with the advantage that it avoids the typical problem of overfitting. Finally, the performance of the new approach based on boosting is measured through a computational experience, determining that the technique based upon boosting decreases the mean squared error by more than 35% with respect to FDH.

1. Introduction

Boosting (Kearns, 1988; Kearns and Valiant, 1994) is a recognized technique in the field of machine learning, based upon an ensemble algorithm for both classification and regression in supervised learning. Boosting is a methodology that works on a family of 'weak learners' (i.e., a model that yields predictions that are marginally correlated with the response variable) to convert them into a strong learner (i.e., a model that is strongly correlated with the response). Let us briefly explain the most widespread boosting algorithm by Freund and Schapire (1997), the so-called AdaBoost.M1, to illustrate the main ideas underpinning this methodology. Let us consider a typical classification problem. A weak classifier is a classifier with an error rate slightly better than random guessing. The objective of boosting is to sequentially fit the model on adapted versions of the data, thus generating a sequence of weak predictive models. The predictions of each weak predictive model (classifier) are then combined through a weighted scheme to yield the better prediction. The weights are updated by the boosting algorithm at each step, assigning more weight to the best classifiers in the sequence (Hastie et al., 2009).

Most of the boosting algorithms in the extant literature have focused on using decision trees as the method's base learner (Hastie et al., 2009). In this regard, the most renowned decision trees algorithm is CART (Classification and Regression Techniques) by Breiman et al. (1984). In CART, the algorithm recursively generates (binary) partitions of the original data until a stopping rule is satisfied. The final representation of the algorithm is a tree structure. CART suffers from overfitting problems unless a process of pruning, based on cross-validation or a test sample, is applied. The larger the tree, the more optimistic the predictions are, yielding predictors with high variance. In this regard, Breiman et al. suggested "pruning" the tree structure by minimizing a certain error-complexity measure, based on combining accuracy and tree size. In the case of combining boosting and CART, i.e., when using a decision tree constructed by CART as a base learner for the boosting algorithm, the pruning process is not necessary since variance reduction of the estimator is achieved by applying the iterative updating procedure linked to boosting.

From an empirical perspective, boosting has recently been successfully applied to many different contexts. For example, Guelman (2012) used boosting for modeling auto insurance loss costs, Brown and Mues (2012) utilized boosting in the context of credit scoring, Landry et al. (2016) for wind power forecasting, Lu et al. (2019) for breast cancer prognosis, Carmona et al. (2019) for predicting failures in the US banking sector, Baboota and Kaur (2019) showed that boosting is one of the best methodologies for estimating football results in the English Premier League, Zhang et al. (2019) used boosting for predicting blood pressure in health, and Hew et al. (2020) predicted student satisfaction through gradient boosting.

However, as far as we are aware, no contribution has been published on how to adapt the boosting algorithm to estimate production functions, where the true surface to be estimated must satisfy certain microeconomic properties. The estimation of production functions is

* Corresponding author.
E-mail addresses: maria.guilleng@umh.es (M.D. Guillen), j.aparicio@umh.es (J. Aparicio), miriam.estevec@umh.es (M. Esteve).

https://doi.org/10.1016/j.eswa.2022.119134
Received 3 June 2021; Received in revised form 16 April 2022; Accepted 21 October 2022
Available online 30 October 2022

related to the determination of technical efficiency in the literature.

The main objective of technical efficiency evaluation is to assess the performance of the so-called Decision Making Units (DMUs). Nowadays, in the non-parametric framework, this task is usually carried out by estimating a best practice frontier from a data sample. Then, technical inefficiency of a DMU can be determined as the distance from the unit to the frontier. Within the single-output multiple-input production framework, the key notion for technical efficiency assessment is the concept of production function. Given an input vector, the production function yields the maximum output attainable from that input bundle. Within the multi-output scenario, the concept of production function is substituted by the more general notion of production frontier, which corresponds to a subset of the boundary of the technology. Following the traditional literature on production theory, any technology and, therefore, its corresponding estimation, must satisfy a certain set of basic postulates (Färe et al., 1985, Färe and Primont, 1995). Typical postulates are: convexity and free disposability in inputs and outputs. In particular, convexity says that if an input-output bundle is feasible (producible) and the inputs are increased and/or outputs are decreased, then the modified input-output bundle is also feasible. In case of producing only one type of output, free disposability in inputs and outputs means that the corresponding production must be monotonically non-decreasing, whereas convexity of the technology is translated into a concave production function. These are some of the main conditions that a suitable estimator of production functions should satisfy.

Nowadays, two renowned (non-parametric) approaches for efficiency evaluation are Free Disposal Hull (FDH) and Data Envelopment Analysis (DEA) (see Charnes et al., 1978, Banker et al., 1984, Deprins and Simar, 1984). FDH is based upon the construction of a technology that satisfies only free disposability and 'deterministicness', i.e., all the observations belong to the technology, which means that data have been observed without stochastic noise. Nevertheless, many possible estimators can satisfy these two postulates. Accordingly, additional requirements are needed to select a suitable estimator. Under FDH, the so-called minimal extrapolation principle is applied: the most conservative estimation of the technology would be that satisfying both free disposability and deterministicness but, additionally, being positioned as close as possible to the observations. This procedure finally leads to a stepwise surface as an estimator of the boundary of the underlying technology. In contrast, DEA also imposes the postulate of convexity as an additional axiom, which is related to a piece-wise linear production frontier. Convexity is a postulate broadly used in production theory; however, it is not always applicable (Kerstens et al., 2019). For example, the technology could be associated with increasing returns to scale, which cannot be modelled by convexity. Consequently, FDH may produce a more flexible estimator of the technology than Data Envelopment Analysis. We will focus on FDH-type estimators throughout this paper.

However, by definition, the FDH technique is not endowed with inferential power. The axiom of minimal extrapolation used by FDH is the cause of the problem of overfitting suffered by the standard Free Disposal Hull technique. The inefficiency scores determined by FDH are always biased: the technique systematically underestimates the actual technical inefficiency of the DMUs. Accordingly, regardless of the obvious data-driven nature of Free Disposal Hull, a gap remains between this well-known methodology for efficiency evaluation and the machine learning field, where current techniques focus on generalization error rather than empirical error. It is worth mentioning that Esteve et al. (2020) have recently introduced the so-called Efficiency Analysis Trees (EAT) technique, which is a non-overfitted version of the FDH approach for estimating production frontiers based on regression trees. In particular, it is an adaptation of CART for technical efficiency assessment.

In this paper, for the first time, boosting is modified with the objective of estimating technologies while fulfilling the conventional postulates defined in production theory and linking the machine learning techniques world with Free Disposal Hull for efficiency measurement. The new approach will lead to an adaptation of gradient tree boosting, which will be called the EATBoost algorithm. The name responds to the idea of using an ensemble method as boosting together with the growth of a decision tree. As we mentioned above, in the literature on boosting, it is very usual to combine this technique with a decision tree based methodology, such as CART. In this regard, one of our objectives is to adapt standard boosting by resorting to the EAT algorithm by Esteve et al. (2020) as the base learner. In this paper, we prove that the technology induced by the new EATBoost algorithm satisfies the postulates of free disposability and deterministicness, and always includes the standard FDH technology as a subset, meaning that minimal extrapolation is not fulfilled. This feature will also allow us to show, through a computational experience, that the EATBoost algorithm does not suffer from overfitting and that it is clearly better than FDH regarding mean squared error and bias. Additionally, we want to highlight the existing parallelism between both approaches, i.e., the standard boosting method and that defined for dealing with the estimation of production functions (the EATBoost algorithm), with respect to the necessity of using an appropriate base learner. While the standard gradient tree boosting resorts to CART, EATBoosting uses EAT as a base learner in the corresponding algorithm. In this way, the similarities between EATBoosting and EAT are the same as those between standard gradient tree boosting and CART. Standard gradient tree boosting cannot exist without a decision tree method as a base learner (CART). And this dependency will equally be observed in the case of the new algorithm (EATBoosting), which needs EAT as a key element to work.

The paper is organized as follows. In Section 2, we briefly introduce the main notions used in the paper, related to the Free Disposal Hull technique, the Efficiency Analysis Trees approach and Gradient Tree Boosting. In Section 3, we define EATBoost, an algorithm based upon boosting, which can estimate production frontiers. In Section 4, we compare the new algorithm with respect to the standard FDH through a computational experience. In Section 5, we show the results corresponding to an empirical illustration. Finally, Section 6 shows the main conclusions and future works.

2. Background: Free Disposal Hull, efficiency analysis trees and gradient tree boosting

2.1. Free Disposal Hull

The Free Disposal Hull (FDH) methodology is an approach for determining multi-dimensional production frontiers and technical efficiency of DMUs. In this regard, let us consider n units to be assessed. $\mathrm{DMU_j} = \left(\boldsymbol{x}_j, \boldsymbol{y}_j\right), j = 1, \ldots, n$, consumes $\boldsymbol{x}_j = (x_{1j}, \ldots, x_{mj}) \in R_+^m$ amounts of inputs for the production of $\boldsymbol{y}_j = \left(y_{1j}, \ldots, y_{sj}\right) \in R_+^s$ amounts of outputs.[1] Additionally, a key notion in production theory is the concept of technology, also called production possibility set, defined as the set of all feasible bundles$(\boldsymbol{x}, \boldsymbol{y})$:$\Psi = \left\{(\boldsymbol{x}, \boldsymbol{y}) \in R_+^{m+s} : \boldsymbol{x} \text{ can produce } \boldsymbol{y}\right\}$. The usual microeconomic assumptions on this set are convexity and free disposability of inputs and outputs. In particular, this last axiom states that if $(\boldsymbol{x}, \boldsymbol{y}) \in \Psi$ is a technically feasible bundle, then$(\boldsymbol{x}', \boldsymbol{y}') \in \Psi$, with $\boldsymbol{x}' \geqslant \boldsymbol{x}$ and$\boldsymbol{y}' \leqslant \boldsymbol{y}$, is also a feasible bundle (Färe and Primont, 1995).[2] Deterministicness is also a usual assumption: $(\boldsymbol{x}_j, \boldsymbol{y}_j) \in \Psi$ for all$j = 1, \ldots, n$.

With respect to gauging inefficiency from a technical point of view, the production frontier of Ψ, which is usually defined as$\partial(\Psi) := \{(\boldsymbol{x}, \boldsymbol{y}) \in \Psi : \widehat{\boldsymbol{x}} < \boldsymbol{x}, \widehat{\boldsymbol{y}} > \boldsymbol{y} \Rightarrow (\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{y}}) \notin \Psi\}$, is a subset of the border of the set Ψ and represents the most important subset of the technology. From this notion, technical inefficiency can be established as the distance from an input-output bundle that belongs to Ψ to$\partial(\Psi)$. One traditional technical

[1] Bold will denote vectors and non-bold scalars.

[2] Let $z = (z_1, \ldots, z_g)$ and $\widehat{z} = (\widehat{z}_1, \ldots, \widehat{z}_g)$. In the paper, $z \leqslant \widehat{z}$ will mean $z_h \leqslant \widehat{z}_h$, $\forall h = 1, \ldots, g$.

efficiency measure is the output-oriented radial model, which coincides with the inverse of the well-known Shephard output distance function (Shephard, 1953). The output-oriented radial model calculates the efficiency score corresponding to the bundle $(\boldsymbol{x}_k, \boldsymbol{y}_k)$ by expanding all its outputs equi-proportionally while keeping inputs constant:

$$\phi(\boldsymbol{x}_k, \boldsymbol{y}_k) = \max\{\phi_k \in R : (\boldsymbol{x}_k, \phi_k \boldsymbol{y}_k) \in \Psi\}.$$

Free Disposal Hull by Deprins and Simar (1984) is a technique that provides estimations of production frontiers associated with technologies satisfying a few postulates in microeconomic theory. Following Deprins and Simar (1984), the FDH estimator of Ψ is defined as:

$$\widehat{\Psi}_{FDH} = \left\{ (\boldsymbol{x}, \boldsymbol{y}) \in R_+^{m+s} : \exists j = 1, ..., n, y_r \leqslant y_{rj}, \forall r = 1, ..., s, x_i \geqslant x_{ij}, \forall i = 1, ..., m \right\}$$

Under the single-output production context (i.e., $s = 1$), the corresponding FDH production function can be approximated by the expression$\widehat{f}_{FDH}(x) = \max\limits_{j: x_i \geqslant x_{ij}, \forall i} \left\{ y_{1j} \right\}$..

Applying the FDH approach, a mixed-integer linear optimization program can be used for determining the output-oriented radial efficiency score corresponding to bundle$(\boldsymbol{x}_k, \boldsymbol{y}_k)$, as follows:

$$\phi^{FDH}(\boldsymbol{x}_k, \boldsymbol{y}_k) = \max \phi_k$$

$$s.t.$$

$$\sum_{j=1}^{n} \lambda_j x_{ij} \leqslant x_{ik}, \ i = 1, ..., m$$

$$\sum_{j=1}^{n} \lambda_j y_{rj} \geqslant \phi_k y_{rk}, \ r = 1, ..., s$$

$$\sum_{j=1}^{n} \lambda_j = 1,$$

$$\lambda_j \in \{0, 1\}, \qquad j = 1, ..., n$$

Finally, note that FDH satisfies an additional axiom: minimal extrapolation. Many estimators of a technology in the production context can satisfy free disposability as well as deterministicness. Accordingly, there could be several estimators fulfilling all these axioms at the same time. In this regard, the most cautious estimation of the frontier would be that located as close as possible to the data cloud. However, it also gives rise to overfitting (see Esteve et al., 2020). Other data-driven methodologies present the same problem. CART, for example, suffers from overfitting problems when the tree is allowed to grow for as long as possible. However, this problem can be solved by cross validating and pruning the deep tree. In the same way, FDH is overly optimistic when technical efficiency has to be determined from a data sample. In this regard, FDH works as a descriptive technique of the extreme behavior of the data, with low inferential power, except for big data sizes (Simar and Wilson, 1998).

2.2. Literature on Free Disposal Hull

In this section, we briefly survey some of the most important contributions on efficiency analysis that adapt, work or are based upon Free Disposal Hull. The objective of this subsection is to justify the interest of researchers in this standard technique for efficiency measurement.

From a theoretical point of view, Cazals et al. (2002) introduced a robust FDH-based estimator to deal with extreme values, while Daraio and Simar (2005) introduced the notion of the conditional order-m efficiency measure. Simar and Vanhems (2012) developed asymptotic properties of the FDH estimator of directional distances as well as robust order-m and order-α directional distance estimators. More recently, Daraio et al., (2020) have proposed a new fast and efficient method to compute exact values of the directional distance estimates for all cases (full and partial frontier cases, unconditional or conditional to external factors). Indeed, it can be considered that the FDH estimator is the initiation point of the probabilistic measurement of production analysis with several innovative applications (Mastromarco and Simar, 2015; Tzeremes, 2015; Kevork et al., 2017; among others). Finally, it is worth mentioning that the FDH method has once more attracted the interest of researchers in recent years. For example, Tavakoli and Mostafaee (2019) adapted Network Data Envelopment Analysis models to the FDH technique. Barbosa et al. (2019) developed a Hybrid Evolutionary Genetic Algorithm and Scatter Search for the discrete and dynamic Berth Allocation Problem, where DEA and FDH were exploited to compare the performance of alternative specifications of the algorithm parameters. Kerstens et al. (2019) reconsidered the way metafrontiers are obtained from nonparametric estimates of underlying group-specific frontiers, concluding that the usual convexification strategy consisting in assuming a convex metaset generally leads to erroneous results. Cordero et al. (2021) analyzed the evolution of the technical efficiency of Spanish regional tax offices by resorting to the conditional directional distance function methodology, based on the FDH estimator of the efficient frontier.

2.3. Gradient tree boosting

Gradient boosting is a type of algorithm that belongs to the field of machine learning, based upon the ensemble of multiple models (Natekin and Knoll, 2013). In fact, boosting is considered one of the most influential ideas introduced in the last thirty years in machine learning (Hastie et al., 2009). In contrast to the most frequent approaches to data-driven modelling where a single "strong" predictive model is fitted, the boosting algorithm depends on merging several relatively "weak" predictive models to get a stronger ensemble estimator. However, unlike other popular machine-learning ensembles techniques, such as Random Forest (Breiman, 2001), which is based upon directly averaging the predictions of the predictive models, boosting is grounded on a forward stagewise strategy where a new predictive model is added in sequence.

Boosting machines work as a classification model or as a regression model, subject to the type of the response variable (binary or continuous). In our production context, we will assume that the response variable is the output of a DMU and, therefore, we will focus our analysis on the regression model. Specifically, the gradient boosting approach provides an estimation $\widehat{f}(\boldsymbol{x})$ in the form of a weighted sum of functions $h(\boldsymbol{x})$, called weak or base learners, from a class of models H, which minimizes the expected value of some specified loss function $L(y_1, f(\boldsymbol{x}))$ (e.g., the mean squared error). To do so, boosting starts with an initial model which consists of a constant function $f_0(\boldsymbol{x}) = \underset{\gamma}{\mathrm{argmin}} \sum_{j=1}^{n} L(y_{1j}, \gamma)$, where γ is the value that minimizes the MSE for every observation. Then, for a finite set of iterations $q = 1, 2, ..., Q$ a new model $h_q(\boldsymbol{x})$ is fitted to the components of the negative gradient of the loss function. The components of the gradient $\boldsymbol{e}_q$ are $e_{jq} = -\left[\frac{\partial L(y_{1j}, f(\boldsymbol{x}_j))}{\partial f(\boldsymbol{x}_j)}\right]_{f = f_{q-1}}$. These components are referred to as pseudo-residuals. The current solution is updated $f_q(\boldsymbol{x}) = f_{q-1}(\boldsymbol{x}) + \gamma_q h_q(\boldsymbol{x})$ where γ_q is computed by solving the optimization model $\gamma_q = \underset{\gamma}{\mathrm{argmin}} \sum_{j=1}^{n} L\left(y_{1j}, f_{q-1}(\boldsymbol{x}_j) + \gamma h_q(\boldsymbol{x}_j)\right)$ (Hastie et al., 2009).

Friedman (2001) proposed gradient tree boosting, an adaptation of standard gradient boosting, where CART trees of a fixed size are used as base learners in the algorithm. This algorithm fits a decision tree $h_q(\boldsymbol{x})$ to pseudo-residuals at the q-th step. In this regard, let J_q be the number of its leaves. The tree splits the input space into J_q disjoint regions (supports) $R_{1q}, ..., R_{J_q q}$, where b_{hq} is the value predicted by the tree model in the region R_{hq}. Thus, $h_q(\boldsymbol{x}) = \sum_{h=1}^{J_q} b_{hq} I(\boldsymbol{x} \in R_{hq})$, where $I(\boldsymbol{x} \in R_{hq}) = \begin{cases} 1 & \text{if} \quad \boldsymbol{x} \in R_{hq} \\ 0 & \text{if} \quad \boldsymbol{x} \notin R_{hq} \end{cases}$. Then, the coefficients b_{hq} are multiplied by some values γ_q, computed by solving the optimization model previously described. Friedman (2001) proposes modifying this algorithm so that it selects a different γ_{hq} for each of the regions R_{hq}, rather than just γ_q for the entire tree. Therefore, the model updating rule becomes $f_q(\mathbf{x}) = f_{q-1}(\mathbf{x}) + \sum_{h=1}^{J_q} \gamma_{hq} I(\mathbf{x} \in R_{hq})$ and $\gamma_{hq} = \underset{\gamma}{\text{argmin}} \sum_{x_j \in R_{hq}} L\left(y_{1j}, f_{q-1}(\mathbf{x}_j) + \gamma\right)$. It is worth mentioning that gradient tree boosting applies CART without the typical pruning procedure. At each step q of the boosting algorithm, a tree structure with J_q leaves is built. This represents a small advantage of gradient tree boosting over CART. Boosting is intensive from a computational point of view but, at least, it does not need to apply the pruning procedure.

As a result, Algorithm 1 is obtained for the standard Gradient Tree Boosting when only one response variable is considered (see Hastie et al., 2009):

Algorithm 1: Gradient Tree Boosting.

```
Set $f_0(\mathbf{x}) = \underset{\gamma}{\text{argmin}} \sum_{j=1}^{n} L(y_{1j}, \gamma)$
1. For $q = 1$ to $Q$:
    a. For $j = 1, 2, ..., n$ calculate $e_{jq} = -\left[\frac{\partial L(y_{1j}, f(\mathbf{x}_j))}{\partial f(\mathbf{x}_j)}\right]_{f = f_{q-1}}$
    b. Fit a regression tree to the targets $e_{jq}$ given terminal regions $R_{hq}, h = 1, 2, ..., J_q$
    c. For $h = 1, 2, ..., J_q$ calculate $\gamma_{hq} = \underset{\gamma}{\text{argmin}} \sum_{x_j \in R_{hq}} L\left(y_{1j}, f_{q-1}(\mathbf{x}_j) + \gamma\right)$
    d. Update $f_q(\mathbf{x}) = f_{q-1}(\mathbf{x}) + \sum_{h=1}^{J_q} \gamma_{hq} I(\mathbf{x} \in R_{hq})$
Final estimation: $\hat{f}(\mathbf{x}) = f_Q(\mathbf{x})$
```

In the above algorithm, two hyperparameters are J_q, the number of leaves of each of the trees, for $q = 1, 2, ..., Q$, and M, the number of iterations. To avoid trees becoming too large in early iterations, which decreases performance and increases computation, restricting all trees to the same size is recommended ($J_q = J, \forall q$). Previous computational experiences indicate that $4 \leqslant J \leqslant 10$ works appropriately for the boosting algorithm (Hastie et al., 2009). One can also introduce shrinkage by scaling the contribution of each tree by a factor $0 < \nu < 1$. In that case, replace the updating rule by $f_q(\mathbf{x}) = f_{q-1}(\mathbf{x}) + \nu \sum_{h=1}^{J_q} \gamma_{hq} I(\mathbf{x} \in R_{hq})$. The parameter ν ($\nu \leqslant 0.1$) may control the learning rate in boosting.

2.4. Efficiency analysis trees

EAT is a recent approach based on machine learning that estimates the production frontier of production possibility sets, satisfying the usual axioms assumed by the standard FDH. The main differences of these two techniques for estimating production frontiers and technical efficiency are as follows. First, from a computational point of view, EAT is clearly more complex than FDH, being based upon a recursive algorithm that belongs to the field of regression trees. Second, EAT provides a non-overfitted estimation of the underlying technology, while FDH, due to the principle of minimal extrapolation, is not able to do that. Regarding the common features between them, both approaches generate stepwise functions as estimators. In Fig. 1, a graphical example of these two techniques, in the case of producing an output from an input, is shown.

Now, we introduce the main steps of the algorithm linked to EAT. Given a data sample $\aleph = \left\{ \left(\mathbf{x}_j, \mathbf{y}_j \right) \right\}_{j=1,\ldots,n}$, the first node of the tree, t_0, contains all the data and must be split into two child nodes, which will be split in subsequent steps. Therefore, we are going to explain how the general splitting step works in the algorithm. So, let us assume that we have a node t in tree structure to be split. This node contains a subset of the original data sample $\aleph$. Then, the algorithm has to select an input i, $i = 1, ..., m$, and a value for this input $s_i \in S_i$. The criterion utilized for that selection is based on minimizing the sum of the Mean Square Error (MSE) associated with the observations belonging to the left child node (the data that fulfil the condition $x_i < s_i$) and the mean squared error calculated for the observations of the right child node (the data that satisfy $x_i \geqslant s_i$). Formally, the split consists in selecting the best combination (x_i^*, s_i^*) that minimizes the expression $R(t_L) + R(t_R) = \frac{1}{n} \sum_{(\mathbf{x}_j, \mathbf{y}_j) \in t_L} \sum_{r=1}^{s} \left(y_{rj} - y_r(t_L) \right)^2 + \frac{1}{n} \sum_{(\mathbf{x}_j, \mathbf{y}_j) \in t_R} \sum_{r=1}^{s} \left(y_{rj} - y_r(t_R) \right)^2$, where $y_r(t)$ denotes the estimation of the r-th output of the node t. Additionally, in the algorithm, a node is a leaf node in the tree structure when a stopping rule is fulfilled. In particular, the most usual stopping rule with regression trees is $n(t) \leqslant n_{\min} = 5$, where $n(t)$ denotes the number of observations at node t. In the algorithm, a relevant point is how to define $y_r(t)$ to guarantee that one of the basic properties of microeconomics, the property of free disposability, is satisfied at each node. To do that, we need to introduce some new notation related to nodes. In this regard, after executing each split, a region in the input space is determined: the "support" of node t (denoted as R_t). Mathematically, it is defined as $R_t = \left\{ \mathbf{x} \in R_+^m : a_{it} \leqslant x_i < b_{it}, i = 1, ..., m \right\}$. The parameters a_{it} and b_{it} are created from the various thresholds selected during the splitting process. Given the notion of support of a node, it is possible to establish the concept of (input) Pareto-dominance to ensure that the estimator $y_r(t)$ satisfies free disposability. In this way, let $k = 1, ..., K$ be the splits completed, $T_k(\aleph)$ be the tree structure built after the k-th split, and $\widetilde{T}_k(\aleph)$ be the set of leaf nodes in $T_k(\aleph)$. Also, let $t^* \in \widetilde{T}_k(\aleph)$ be a node to be split, then $T(k|t^* \rightarrow t_L, t_R)$ denotes the tree linked to this specific split. Given node t, its set of

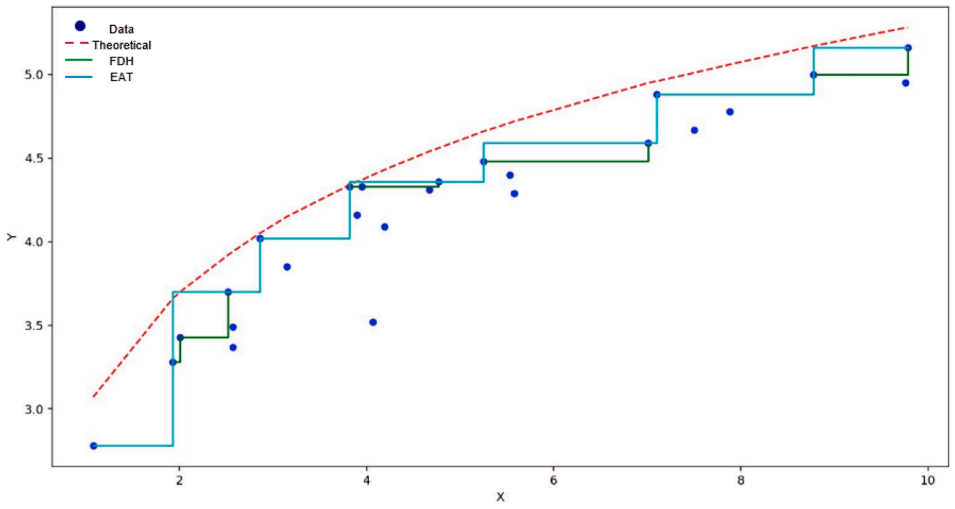

Fig. 1. Graphical illustration of the frontier estimates provided by FDH and EAT (Esteve et al, 2021).

(input) Pareto-dominant nodes is stated as $I_{T_k(\aleph)}(t) = \left\{ t' \in \widetilde{T}_k(\aleph) - t : \exists \boldsymbol{x} \in R_t, \exists \boldsymbol{x}' \in R_{t'} \text{ such that } \boldsymbol{x}' \leqslant \boldsymbol{x} \right\}$. Returning to how to estimate $y_r(t)$ satisfying free disposability, for any node $t^* \in \widetilde{T}_k(\aleph)$ the estimation of the right child node coincides with the estimation of its parent node, i.e., $y_r(t_R) = y_r(t^*)$, $r = 1, \ldots, s$, and the estimation of the left child node is defined as follows:

$$y_r(t_L) = \max\left\{ \max\left\{ y_{rj} : (x_j, y_j) \in t_L \right\}, y_r\left(I_{T(k|t^* \to t_L, t_R)}(t_L) \right) \right\}, r = 1, \ldots, s,$$

where $y_r\left(I_{T(k|t^* \to t_L, t_R)}(t_L)\right) = \max\left\{ y_r(t') : t' \in I_{T(k|t^* \to t_L, t_R)}(t_L) \right\}$, being $y_r(t')$ the estimation of the output y_r at node $t' \in \widetilde{T}(k|t^* \to t_L, t_R)$, $r = 1, \ldots, s$.

Under the standard application of EAT, the algorithm provides a deep tree that can suffer from the same problems as FDH, i.e., overfitting. In this regard, with the objective of improving the model, Esteve et al. (2020) suggested applying the pruning procedure previously introduced by Breiman et al. (1984). However, in this paper, the application of this complex procedure is not necessary since the standard gradient tree boosting, which we intend to adapt, does not resort to pruning but the specification of a particular number of terminal nodes. Accordingly, let $T(\aleph)$ be the generated tree after the application of EAT and let $\boldsymbol{d}^{T(\aleph)}(\boldsymbol{x})$ be the multidimensional output estimator defined from the tree $T(\aleph)$, i.e., $d_r^{T(\aleph)}(\boldsymbol{x}) = \sum_{t \in \widetilde{T}(\aleph)} y_r(t) I(\boldsymbol{x} \in t)$, for all $r = 1, \ldots, s$, with $I(\cdot)$ being the indication function. From this estimator, the technology estimated from the EAT algorithm would be as follows:

$$\widehat{\Psi}_{T(\aleph)} = \left\{ (\boldsymbol{x}, \boldsymbol{y}) \in R_+^{m+s} : \boldsymbol{y} \leqslant \boldsymbol{d}^{T(\aleph)}(\boldsymbol{x}) \right\} \qquad 5$$

Furthermore, $\phi(\boldsymbol{x}_k, \boldsymbol{y}_k)$, i.e., the efficiency score linked to the vector $(\boldsymbol{x}_k, \boldsymbol{y}_k)$, is estimated by the following mixed-integer linear optimization program:

$$\begin{array}{lll} \phi^{EAT}(\boldsymbol{x}_k, \boldsymbol{y}_k) = & \max & \phi \\ & s.t. & \\ & \displaystyle\sum_{t \in \widetilde{T}(\aleph)} \lambda_t a_{it} \leqslant \phi x_{ik}, & i = 1, \ldots, m \\ & \displaystyle\sum_{t \in \widetilde{T}(\aleph)} \lambda_t d_r^{T(\aleph)}(\boldsymbol{a}_t) \geqslant y_{rk}, & r = 1, \ldots, s \\ & \displaystyle\sum_{t \in \widetilde{T}(\aleph)} \lambda_t = 1, & \\ & \lambda_t \in \{0, 1\}, & t \in \widetilde{T}(\aleph) \end{array} \qquad 6$$

3. The EATBoost algorithm

In this section, we adapt Gradient Tree Boosting, by resorting to Efficiency Analysis Trees (Esteve et al., 2020) as base learners, for the estimation of technologies that satisfy free disposability and deterministicness, which will be called EATBoost. We start with the single-output case and, later, we extend the approach to deal with multiple outputs.

3.1. The single-output case

We first must state a specific loss criterion $L(y_1, f(\boldsymbol{x}))$, since different loss criteria result in different algorithms. Due to the fact that the EAT method resorts to the Mean Squared Error, we define our loss function as $\frac{1}{2}\left(y_{1j} - \widehat{f}(\boldsymbol{x}_j)\right)^2, \forall j = 1, \ldots, n$ (Hastie et al., 2009). Now, according to step 1 of Algorithm 1, the value of $f_0(\boldsymbol{x})$ such that it minimizes $\frac{1}{n}\sum_{j=1}^n \left(y_{1j} - f_0(\boldsymbol{x}_j)\right)^2$ must be calculated. However, in the production context that we are dealing with, an additional condition, $f_0(\boldsymbol{x}) \geqslant y_{1j}$ for all $j = 1, \ldots, n$, must be established. It means that the initial estimation of the output envelops all the observations from above. Esteve et al. (2020, Proposition 1) proved that $f_0(\boldsymbol{x}) = \max_{j=1,\ldots,n}\left\{ y_{1j} \right\}$. Now, according to Friedman (2001), for a finite set of iterations $q = 1, 2, \ldots, Q$, a new model $h_q(\boldsymbol{x})$ is fitted to the components of the negative gradient of the loss function (steps 2.a and 2.b). The components of the gradient $\boldsymbol{e}_q$ are now $e_{jq} = -\left[\frac{\partial L(y_{1j}, f(x_j))}{\partial f(x_j)}\right]_{f = f_{q-1}} = y_{1j} - f_{q-1}(\boldsymbol{x}_j)$ and the model $h_q(\boldsymbol{x})$ is defined, in our context, as the tree T that comes from the EAT algorithm with a stopping rule such that J_q is the number of leaves. As established in the standard Gradient Tree Boosting, the tree model $T(\aleph_q)$ is fitted using the dataset $\aleph_q = \left\{ (\boldsymbol{x}_j, e_{jq}) \right\}_{j=1,\ldots,n}$, with the pseudo-residuals as output values. In step 2.c, the value γ_{hq} that minimizes the loss function over the region R_{hq} of the input space is defined as $d_1^{T(\aleph_q)}(R_{hq})$, i.e., the output estimation given by the EAT algorithm for the region $\mathrm{R_{hq}}$. This is defined by analogy with the standard Gradient Tree Boosting when CART is used as the base learner. In that case, the value $\mathrm{\gamma_{hq}}$ that minimizes the loss function over the region $\mathrm{R_{hq}}$ is the estimation of the output given by CART for the final node associated with $\mathrm{R_{hq}}$. Finally, steps 2.d and 3 are not modified from Algorithm 1. As a result, we obtain Algorithm 2: the EATBoost algorithm.

Algorithm 2: EATBoost (single-output version).

```
Set $f_0(\boldsymbol{x}) = \max_{j=1,\ldots,n}\left\{ y_{1j} \right\}$
1. For $q = 1$ to$Q$:
    a. For $j = 1, 2, \ldots, n$ calculate $e_{jq} = y_{1j} - f_{q-1}(\boldsymbol{x}_j)$
    b. Fit a deep EAT to the targets $e_{jq}$ given terminal regions$R_{hq}$,$h = 1, 2, \ldots, J_q$
    c. For $h = 1, 2, \ldots, J_q$ calculate $\gamma_{hq} = d_1^{T(\aleph_q)}(R_{hq})$
    d. Update $f_q(\boldsymbol{x}) = f_{q-1}(\boldsymbol{x}) + v\sum_{h=1}^{J_q} \gamma_{hq} I(\boldsymbol{x} \in R_{hq})$

Final estimation: $\widehat{f}(\boldsymbol{x}) = f_Q(\boldsymbol{x})$
```

Algorithm 2 could be seen as a natural adaptation of Gradient Tree Boosting for estimating production functions, using the Efficiency Analysis Trees as base learners. However, the estimator derived from the EATBoost algorithm would not be a valid estimator of a production function in microeconomics unless it satisfies, at least, free disposability and deterministicness. Under the single-output scenario, these two properties are translated into monotonicity, i.e., the estimated function must be monotonically non-decreasing, and the function must envelop the observations from above. Certainly, the following propositions prove that $\widehat{f}(\boldsymbol{x})$ derived from EATBoost fulfils both properties, even using a strictly less than one learning rate v.

Proposition 1. *Let $v \in (0, 1]$. Then, $\widehat{f}(\boldsymbol{x})$ is a monotone non-decreasing function.*

Proof. Given $q = 1, \ldots, Q$, let us assume that $f_{q-1}(\boldsymbol{x})$ is a monotone non-decreasing function. The application of the EAT algorithm in step 2.b guarantees that the function $h_q(\boldsymbol{x}) = \sum_{h=1}^{J_q} \gamma_{hq} I(\boldsymbol{x} \in R_{hq})$ is a monotone non-decreasing function (Theorem 1 in Esteve et al., 2020). Since $v > 0$, $h_q^v(\boldsymbol{x}) = v\sum_{h=1}^{J_q} \gamma_{hq} I(\boldsymbol{x} \in R_{hq})$ is also a monotone non-decreasing function. Then, $f_q(\boldsymbol{x}) = f_{q-1}(\boldsymbol{x}) + v\sum_{h=1}^{J_q} \gamma_{hq} I(\boldsymbol{x} \in R_{hq})$ is the sum of two monotone non-decreasing functions. Hence, $f_q(\boldsymbol{x})$ is a monotone non-decreasing function. By induction, it is enough to prove that $f_0(\boldsymbol{x})$ is also a monotone non-decreasing function. Regarding this first element, $f_0(\boldsymbol{x}) = \max_{j=1,\ldots,n}\left\{ y_{1j} \right\}$, which is clearly a constant function. Therefore, $f_0(\boldsymbol{x})$ is a monotone non-decreasing function. Finally, we have that $\widehat{f}(\boldsymbol{x}) = f_Q(\boldsymbol{x})$ is also a monotone non-decreasing function.

Proposition 2. *Let $v \in (0, 1]$. Then, $\widehat{f}(\boldsymbol{x}_j) \geqslant y_{1j}$, for all $j = 1, \ldots, n$.*

Proof. Given a certain q, $q = 1, \ldots, Q$, let us suppose that $f_{q-1}(\boldsymbol{x}_j) \geqslant y_{1j}$, for all $j = 1, \ldots, n$. The EAT algorithm applied on the data sample $\aleph_q =$

$\{(x_j, e_{jq})\}_{j=1,\ldots,n}$ satisfies that $h_q(x_j) = \sum_{h=1}^{J_q} \gamma_{hq} I(x_j \in R_{hq}) \geqslant e_{jq}$, for all $j = 1,\ldots,n$. Then, $f_q(x) = f_{q-1}(x) + \nu \sum_{h=1}^{J_q} \gamma_{hq} I(x \in R_{hq}) \geqslant f_{q-1}(x_j) + \nu \cdot e_{jq} = f_{q-1}(x_j) + \nu\left(y_{1j} - f_{q-1}(x_j)\right) = \nu \cdot y_{1j} + (1-\nu) f_{q-1}(x_j)$, which is a convex combination of y_{1j} and $f_{q-1}(x_j) \geqslant y_{1j}$ (by hypothesis) since $\nu \in (0,1]$. This means that $\nu \cdot y_{1j} + (1-\nu) f_{q-1}(x_j) \geqslant y_{1j}$. Hence, $f_q(x_j) \geqslant y_{1j}$, for all $j = 1,\ldots, n$. Finally, by induction, it is enough to prove that $f_0(x_j) \geqslant y_{1j}$, for all $j = 1, \ldots, n$. This last statement is trivial because, by definition, $f_0(x_j) = \max_{j=1,\ldots,n}\{y_{1j}\}$

Although, EATBoost and the EAT algorithm do not generate the same estimator in the most general framework, the next result proves that the two approaches provide the same estimator when only one iteration is made, i.e. $Q = 1$, the number of leaves is the same and the learning rate is set as $\nu = 1$.

Proposition 3. *If $Q = 1$, $J_1 = \left|\widetilde{T}(\aleph)\right|$ and $\nu = 1$, then $\widehat{f}(x) = d_1^{T(\aleph)}(x)$.*

Proof. Let $f_0(x_j) = \max_{j=1,\ldots,n}\{y_{1j}\}$. As a result, $e_{1j} = y_{1j} - f_0(x_j) = y_{1j} - \max_{j=1,\ldots,n}\{y_{1j}\}$. Additionally, if the training data are transformed in the way $y'_{1j} = y_{1j} + k$, $\forall j = 1, \ldots, n$, where k is a constant, then the estimator derived from the deep EAT, $d_1^{T(\aleph')}(x)$, is $d_1^{T(\aleph')}(x) = d_1^{T(\aleph')}(x) + k$. Moreover, in this case, the two tree structures, $T(\aleph)$ and $T(\aleph')$, yield the same splits of the input space (the same regions) since the Mean Squared Error is not affected by the transformation of the data. Note that $\max_{j=1,\ldots,n}\{y_{1j}\}$ is a constant. Therefore, if $J_1 = \left|\widetilde{T}(\aleph)\right|$, then $\gamma_{h1} = d_1^{T(\aleph')}(R_{h1}) = d_1^{T(\aleph)}(R_{h1}) - \max_{j=1,\ldots,n}\{y_{1j}\}$, for all $h = 1,2,\ldots,J_1$. In this way, we have $\widehat{f}(x) = f_1(x) = f_0(x) + \sum_{h=1}^{J_1} \gamma_{h1} I(x \in R_{h1}) = \max_{j=1,\ldots,n}\{y_{1j}\} + \sum_{h=1}^{J_1}\left(d_1^{T(\aleph)}(R_{h1}) - \max_{j=1,\ldots,n}\{y_{1j}\}\right) I(x \in R_{h1}) = \max_{j=1,\ldots,n}\{y_{1j}\} - \max_{j=1,\ldots,n}\{y_{1j}\} + \sum_{h=1}^{J_1} d_1^{T(\aleph)}(R_{h1}) I(x \in R_{h1}) = \sum_{h=1}^{J_1} d_1^{T(\aleph)}(R_{h1}) I(x \in R_{h1}) = d_1^{T(\aleph)}(x)$. ■.

Furthermore, due to the equivalence of EAT and Free Disposal Hull when the number of inputs is one, i.e., $m = 1$, and the stopping rule is set to $n_{\min} = 1$ (Esteve et al., 2020, Proposition 3), we can prove that under the same hypotheses both EATBoost and FDH generate the same prediction function.

Corollary 1. *If $m = 1$, $n_{\min} = 1$, $Q = 1$, $J_1 = \left|\widetilde{T}(\aleph)\right|$ and $\nu = 1$, then $\widehat{f}(x) = \widehat{f}_{FDH}(x)$.*

Proof. It is straightforward using Proposition 3 in Esteve et al. (2020) and our previous proposition.

3.2. The multi-output case

In this section, we extend the univariate EATBoost version to the more general and usual multi-output production context. To do so, it is enough to define the multi-dimensional pseudo-residuals as $e_{rjq} = -\left[\frac{\partial L(y_j, f(x_j))}{\partial f_r(x_j)}\right]_{f_r(x_j) = f_{rq-1}(x_j)} = y_{rj} - f_r(x_j)$ $r = 1, \ldots, s$ where s is the number of outputs. As a result, γ_{hq} is the vector predicted by EAT from the pseudo-residuals of step q. The final prediction derived by the EATBoost algorithm, $\widehat{f}(x)$, would be a vector of s components: $\widehat{f}(x) = \left(\widehat{f}_1(x), \ldots, \widehat{f}_s(x)\right)$. The multi-output version of the EATBoost algorithm is summarized in Algorithm 3.

Algorithm 3: EATBoost (multi-output version).

```
Set $f_0(x) = \left(\max_{j=1,\ldots,n}\{y_{1j}\}, \ldots, \max_{j=1,\ldots,n}\{y_{sj}\}\right)$
1. For $q = 1$ to $Q$:
   a. For $j = 1, 2, \ldots, n$ calculate $e_{jq} = y_j - f_{q-1}(x)$
   b. Fit a deep EAT to the targets $e_{jq}$ given terminal regions $R_{hq}$, $h = 1, 2, \ldots, J_q$
   c. For $h = 1, 2, \ldots, J_q$ calculate $\gamma_{hq} = d^{T(\aleph_q)}(R_{hq})$
   d. Update $f_q(x) = f_{q-1}(x) + \nu \sum_{h=1}^{J_q} \gamma_{hq} I(x \in R_{hq})$
Final estimation: $\widehat{f}(x) = f_Q(x)$
```

Just as the EATBoost algorithm was naturally extended from the single-output case to the multi-output scenario, it is also possible to prove that each component of the vector $\widehat{f}(x)$ also satisfies monotonicity, as happens in the one-dimensional case. This result is proved in Proposition 4.

Proposition 4. *$\widehat{f}_r(x)$ is a monotone non-decreasing function for all $r = 1, \ldots, s$.*

Proof. It is straightforward invoking Proposition 1.

Additionally, each component of the vector containing the value of the estimated outputs defines a function that envelops the observations from above in its corresponding dimension, as established in Proposition 5.

Proposition 5. *$\widehat{f}_r(x_j) \geqslant y_{rj}$, for all $j = 1, \ldots, n$, and for all $r = 1, \ldots, s$.*

Proof. The proof of this result is straightforward invoking Proposition 2. ■.

In the standard literature, when the multi-output scenario is considered, the cornerstone notion of production function is substituted by the more complex concept of production possibility set or technology. If Algorithm 3, the multi-output version of EATBoost, would be working in a suitable way within the production theory context, then one should be able to define an estimated technology satisfying at least free disposability and deterministicness from $\widehat{f}(x)$. Next, we show the induced technology that can be derived from the EATBoost algorithm.

Let $\widehat{f}(x)$ be the vector of estimated outputs from the input bundle $x \in R_+^m$ once the EATBoost algorithm, Algorithm 3, has been applied. Then, the technology estimated by boosting would be as follows:

$$\widehat{\Psi}_B = \left\{(x, y) \in R_+^{m+s} : y \leqslant \widehat{f}(x)\right\}$$

Next, we prove that $\widehat{\Psi}_B$ meets the axiom of free disposability and other properties.

Proposition 6. *$\widehat{\Psi}_B$ satisfies:*

(i) Free disposability in inputs and outputs.
(ii) Deterministicness, i.e., $\left(x_j, y_j\right) \in \widehat{\Psi}_B$, for all $j = 1, \ldots, n$;
(iii) $\widehat{\Psi}_{FDH} \subseteq \widehat{\Psi}_B$.

Proof. (i) Let $(x, y) \in \widehat{\Psi}_B$ and let $(x', y') \in R_+^{m+s}$ such that $x' \geqslant x$ and $y' \leqslant y$. First, we have that $y \leqslant \widehat{f}(x)$ because $(x, y) \in \widehat{\Psi}_B$. Second, by Proposition 4, we have that if $x' \geqslant x$, then $\widehat{f}(x') \geqslant \widehat{f}(x)$. Therefore, $y' \leqslant y \leqslant \widehat{f}(x) \leqslant \widehat{f}(x')$, which implies that $(x', y') \in \widehat{\Psi}_B$ by. (ii) See Proposition 5. (iii) By definition, $\widehat{\Psi}_{FDH}$ is the smallest set that satisfies (i) and (ii). As a result, $\widehat{\Psi}_{FDH} \subseteq \widehat{\Psi}_B$. ■.

As Proposition 6 (iii) states, the EATBoost algorithm no longer satisfies the minimal extrapolation principle. Therefore, generally speaking, it does not coincide with the smallest set that satisfies both free disposability and the deterministic property. This also means that the new technique could avoid overfitting to the data sample, also referred to as overlearning in the machine learning field, something that systematically happens with FDH. At this point, the difficulty associated with the technology estimator linked to the result of the EATBoost algorithm would be where to locate the production frontier while enveloping the data from above. The main objective is to correctly estimate the underlying technology from which the data were generated. However, in practice, this set is unknown to researchers. In machine learning, a usual way of checking the goodness of the proposed estimation is resorting to cross validation or a test sample. It has a double effect. First, this process allows the quality of the estimations provided by the model to be checked. Second, in boosting algorithms where the results of the model depend on several parameters (as, for example, the number of iterations or the number of leaves), it allows a grid of parameter values to be assessed and the best solution to be selected, i.e., the best combination of parameters.

In this paper, we propose to randomly divide the original dataset into a training (70 %) and a test sample (30 %).[3] Let us assume, without loss of generality, that the training set contains the observations with indices $j = 1, ..., n_1$, whereas the test sample contains the data with indices $j = n_1 + 1, ..., n$, being $n_1 < n$. Given a combination of parameters (Q, J, v); put in words, the number of iterations, the number of leaves and the learning rate, the training sample is used for training the EATBoost algorithm, obtaining the function $\widehat{f}(x|Q, J, v)$. Then, the 'quality' of this estimator is assessed by calculating the difference between the observed output y_j and the estimated output $\widehat{f}(x_j|Q, J, v)$, for all $j = n_1 + 1, ..., n$, i.e., for the observations belonging to the test sample. All these values are also aggregated through the MSE. A grid of different combinations of values for (Q, J, v) is checked. Following the literature (see Hastie et al., 2009), $4 \leqslant J \leqslant 10$ and $0 < v \leqslant 0.1$. In this way, a final combination (Q^*, J^*, v^*) is obtained as that which achieves the minimum value for MSE. Then, as a final step, the full original data sample is used for training the EATBoost algorithm with parameters (Q^*, J^*, v^*), obtaining the estimation function $\widehat{f}(x|Q^*, J^*, v^*)$, which will be used for estimating the output vector of a firm given any certain input bundle.

4. Computational experience

In this section, we present an evaluation of FDH and EATBoost for the estimation of a production function simulated from the typical Cobb-Douglas function in microeconomics (see Table 1), playing with different sample sizes (50, 75 and 100 units) and number of inputs (3, 6, 9, 12 and 15). The inputs were randomly sampled from $Uni[1, 10]$, while the inefficiency term followed a truncated normal distribution $|N(0, 0.4)|$. Two hundred trials were executed for each combination of number of inputs and sample size. To check the results, the usual MSE and bias were used. For each technique, the corresponding efficiency score can be determined through the ratio $\widehat{f}(x)/y$..

Table 2 reports the results of the simulations. We have shown, using brackets, the relative difference between FDH and EATBoost regarding mean squared error and bias. These values represent the percentage of decrease of the mean squared error and bias when the new technique EATBoost is applied in comparison with FDH. Regarding the results, MSE and bias of the EATBoost method were clearly smaller than the same performance measures for the FDH technique. The improvements ranged from 35 % to 77 % in the case of MSE, and from 23 % to 57 % in the case of bias. Additionally, a pattern is detected in Table 2: the bigger the sample size, the greater the improvement. Nevertheless, from 100 units, the enhancement increases very slightly (the percentages are similar for $n = 100$ and $n = 150$). Moreover, for the biggest sample sizes ($n = 100$ and $n = 150$), we observe that the number of inputs also affects the results. In particular, the worst results (i.e., the lowest improvement percentages) are observed for the maximum number of inputs considered in the analysis ($m = 15$). In the case of $n = 50$ (the smallest sample size considered), the best results are observed for $m = 9$. Overall, we observe better results with the new approach than with the standard FDH, with respect to both MSE and bias. Nevertheless, and although the results derived from the two methods seem different (better in the case of EATBoost), we also check whether the score vectors associated with the two techniques (FDH and EATBoost) show statistically significant differences or not. To this end, we apply the Li test (Simar and Zelenyuk, 2006). Regarding the results, all p-values obtained are less than 0.05 (the typical significance level) except in the case of tests corresponding to sample sizes of 50 units and 3 inputs. In this particular scenario, we reject the hypothesis that the two score distributions (associated with FDH and EATBoost) are equal for 76 % of the comparisons performed. However, the same cannot be claimed for 24 % of the cases.

Finally, let us mention a particular shortcoming of the new methodology in comparison with the standard Free Disposal Hull: the computational time. The simulations for obtaining Table 2 were performed on a PC with a 1.8 GHz dual-core Intel Core i7 processor, 8 Gigabyte of RAM and a Microsoft Windows 10 Enterprise operating system. Our algorithm was implemented in Python. The mean time (in seconds) associated with each technique is also reported in Table 2, showing clear differences between FDH and EATBoost. Additionally, Fig. 2 shows the computational time (mean) related to the application of the EATBoost algorithm across the scenarios considered in the simulations.

5. An empirical illustration

In this section, we illustrate how the new approach works in comparison with the standard FDH through an empirical example. In particular, we use the dataset proposed by Juo et al. (2015) for the year 2010, where data from 31 Taiwanese banks were obtained.[4] We use financial funds, labor and physical capital as inputs; while revenue obtained from financial investment and loans is used as output.

The results are shown in Table 3, where the first column represents the name of the bank and the subsequent four columns, the corresponding value of its inputs and output. In the table, the radial output score for each bank is shown by resorting to the new approach (column labelled as 'EATBoost score') and the standard Free Disposal Hull (column labelled as 'FDH score'). The FDH technique shows that virtually all the banks are technically efficient. Exceptions are First Bank, Hua Nan Bank and Sunny Bank. In contrast, the new technique, based on boosting, helps to discriminate between the efficient and inefficient companies. In this case, only 11 out of the 31 banks are classified as technically efficient.

Moreover, we apply the Li test (Simar and Zelenyuk, 2006) to check if the score vectors for the two methods considered in this empirical example show statistically significant differences. In this case, the p-value was zero and, consequently, the null hypothesis of the equality of efficiency score distributions can be rejected. The densities of both score distributions have been represented in Fig. 3.

6. Conclusions and future work

Our paper represents the first adaptation of boosting techniques, from a methodological and computational viewpoint, for providing estimates of production possibility sets fulfilling some usual axioms from production theory. Thus, the new approach endows standard techniques like FDH with certain features from the field of machine learning. Nowadays, the interest in building new bridges between machine

[3] The 70–30 rule is very common in machine learning in practice (see, for example, Hastie et al., 2009). Additionally, a recent contribution (Xu and Goodacre, 2018) claims that if the researcher chooses a reasonable balance between training and test sets (50–70% for training), it is likely that a good final model will be obtained.

[4] We are grateful to these authors for sharing the data.

Table 1
Scenarios to be simulated.

Num. inputs	$f(\boldsymbol{x})$
3	$f(\boldsymbol{x}) = 3 \cdot x_1^{0.05} \cdot x_2^{0.05} \cdot x_3^{0.4} \cdot e^{-u}$
6	$f(\boldsymbol{x}) = 3 \cdot x_1^{0.05} \cdot x_2^{0.001} \cdot x_3^{0.004} \cdot x_4^{0.045} \cdot x_5^{0.1} \cdot x_6^{0.3} \cdot e^{-u}$
9	$f(\boldsymbol{x}) = 3 \cdot x_1^{0.05} \cdot x_2^{0.001} \cdot x_3^{0.004} \cdot x_4^{0.005} \cdot x_5^{0.001} \cdot x_6^{0.004} \cdot x_7^{0.08} \cdot x_8^{0.075} \cdot x_9^{0.28} \cdot e^{-u}$
12	$f(\boldsymbol{x}) = 3 \cdot x_1^{0.005} \cdot x_2^{0.001} \cdot x_3^{0.004} \cdot x_4^{0.005} \cdot x_5^{0.001} \cdot x_6^{0.004} \cdot x_7^{0.08} \cdot x_8^{0.05} \cdot x_9^{0.05} \cdot x_{10}^{0.075} \cdot x_{11}^{0.025} \cdot x_{12}^{0.2} \cdot e^{-u}$
15	$f(\boldsymbol{x}) = 3 \cdot x_1^{0.005} \cdot x_2^{0.001} \cdot x_3^{0.004} \cdot x_4^{0.005} \cdot x_5^{0.001} \cdot x_6^{0.004} \cdot x_7^{0.08} \cdot x_8^{0.05} \cdot x_9^{0.05} \cdot x_{10}^{0.05} \cdot x_{11}^{0.025} \cdot x_{12}^{0.025} \cdot x_{13}^{0.025} \cdot x_{14}^{0.025} \cdot x_{15}^{0.15} \cdot e^{-u}$

Table 2
Simulation results.

n	*m*	Mean Square Error			BIAS			Mean Time (*sec*)	
		FDH	EATBoost		FDH	EATBoost		FDH	EATBoost
50	3	1.086	0.630	(42 %)	1.086	0.627	(23 %)	0.52	202.79
50	6	3.795	1.919	(49 %)	3.795	1.157	(29 %)	0.83	450.08
50	9	3.980	1.325	(67 %)	3.980	0.939	(43 %)	1.24	751.47
50	12	3.927	2.341	(40 %)	3.927	1.202	(27 %)	2.41	1037.57
50	15	3.958	2.587	(35 %)	3.958	1.178	(29 %)	2.32	1365.97
100	3	0.831	0.201	(76 %)	0.831	0.374	(47 %)	2.79	418.40
100	6	3.447	1.140	(67 %)	3.447	0.852	(46 %)	5.11	952.49
100	9	4.073	1.274	(69 %)	4.073	0.873	(48 %)	8.11	1459.58
100	12	4.037	1.611	(60 %)	4.037	0.970	(42 %)	8.88	2150.14
100	15	4.004	1.726	(57 %)	4.004	1.093	(35 %)	11.67	2898.28
150	3	0.695	0.202	(71 %)	0.695	0.341	(47 %)	5.30	954.04
150	6	3.236	0.747	(77 %)	3.236	0.652	(57 %)	8.71	2027.86
150	9	3.865	1.177	(70 %)	3.865	0.838	(49 %)	12.04	3235.60
150	12	3.982	1.384	(65 %)	3.982	0.938	(44 %)	16.47	4349.84
150	15	3.956	1.670	(58 %)	3.956	1.053	(37 %)	17.66	5291.47

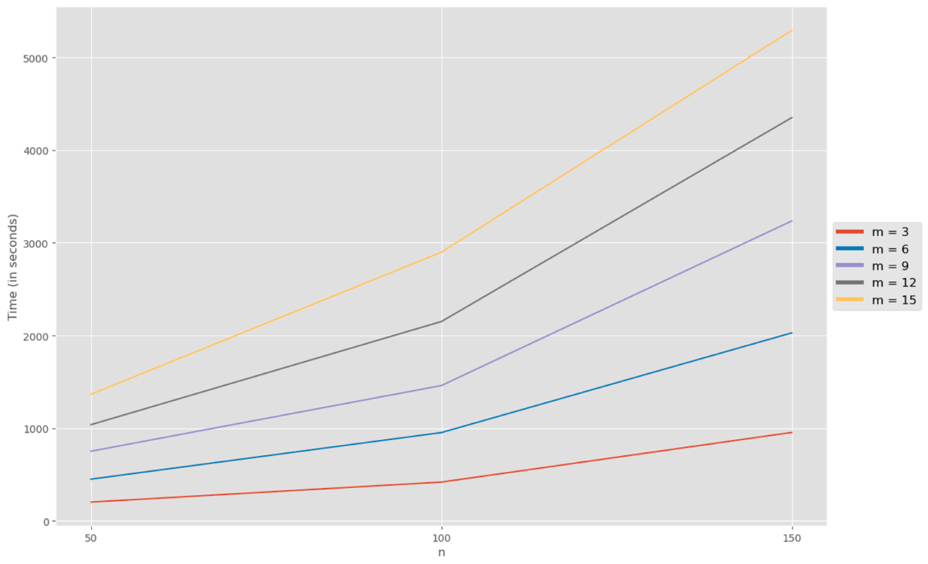

Fig. 2. Computational time for EATBoost.

learning techniques and efficiency measurement is increasing (see, for example, Khezrimotlagh et al., 2019, and Zhu, 2019; who work on big data and DEA, the book on data science and productivity by Charles et al., 2020, the preference learning contribution on preference inference and DEA by Pereira et al., 2020, Tsolas et al., 2020; who propose a two-stage hybrid model that integrates Artificial Neural Network with DEA, and Thaker et al., 2021; who combine DEA and Random Forest regression to examine the impact of corporate governance in the Indian banking sector). Our contribution is in this research line. In particular, we introduce a new approach for efficiency analysis by adapting the well-known Gradient Tree Boosting algorithm, defining the so-called EATBoost algorithm, which uses EAT as the base learner. Specifically, the standard Gradient Tree Boosting algorithm was modified to deal with the axiom of free disposability in inputs and outputs and to provide estimates that envelop the data cloud from above. These two same postulates are also key in the definition of the Free Disposal Hull (FDH) estimator of a technology, which is a traditional and well-known non-parametric technique. This was why we compared the performance of the new approach with respect to FDH in this paper. Our computational experience showed that the results linked to the EATBoost algorithm clearly outperform FDH with regard to the reduction in bias and mean squared error. In contrast to Free Disposal Hull, the EATBoost algorithm does not assume the principle of minimal extrapolation, which allows the technique to overcome the problem of overfitting. However, we would like to highlight a clear limitation associated with the new approach. The EATBoost algorithm is linked to a very intensive

Table 3
Empirical illustration.

Bank	Financial funds	Labor	Capital	Revenue	EATBoost score	FDH score
Export-Import Bank	25,019	202	505	1056	1.01	1.00
Bank of Taiwan	3,171,493	7951	76,576	41,007	1.00	1.00
Taipei Fubon Bank	1,222,499	6434	12,082	19,402	1.04	1.00
Bank of Kaohsiung	189,169	914	2237	2957	1.00	1.00
Land Bank	1,846,028	5732	22,634	31,506	1.05	1.00
Cooperative Bank	2,220,071	8835	33,638	35,510	1.02	1.00
First Bank	1,602,733	7048	22,843	27,084	1.09	1.09
Hua Nan Bank	1,595,039	7126	24,907	25,668	1.15	1.15
Chang Hwa Bank	1,267,731	6428	23,778	21,638	1.06	1.00
Mega Bank	1,589,474	5033	13,568	29,489	1.00	1.00
Cathay United Bank	1,349,708	6062	25,285	21,904	1.35	1.00
The Shanghai Bank	577,127	2265	10,190	9215	1.01	1.00
Union Bank	295,386	2975	8051	8708	1.00	1.00
Far Eastern Bank	344,499	2378	2855	6616	1.08	1.00
E. Sun Bank	936,612	4583	14,195	16,911	1.03	1.00
Cosmos Bank	109,728	1685	6162	5251	1.00	1.00
Taishin Bank	741,883	6236	17,278	16,319	1.03	1.00
Ta Chong Bank	322,836	3215	3208	7191	1.14	1.00
Jih Sun Bank	182,228	1529	4238	3219	1.01	1.00
Entie Bank	298,330	1945	2114	5410	1.00	1.00
China Trust Bank	1,335,080	9538	32,436	29,185	1.01	1.00
Sunny Bank	213,037	1790	9116	4128	1.31	1.27
Bank of Panhsin	143,268	1270	7416	2824	1.02	1.00
Taiwan Business Bank	1,035,800	5010	14,185	18,056	1.00	1.00
Taichung Bank	313,451	1829	3243	5770	1.00	1.00
China Development	77,242	567	1219	1873	1.01	1.00
Hwatai Bank	105,657	856	1667	2284	1.00	1.00
Cota Bank	108,685	1078	1113	2283	1.00	1.00
Ind. Bank of Taiwan	76,383	282	2605	1577	1.01	1.00
Bank SinoPac	936,418	4670	8721	17,922	1.00	1.00
Shin Kong Bank	428,995	3146	7123	8226	1.04	1.00

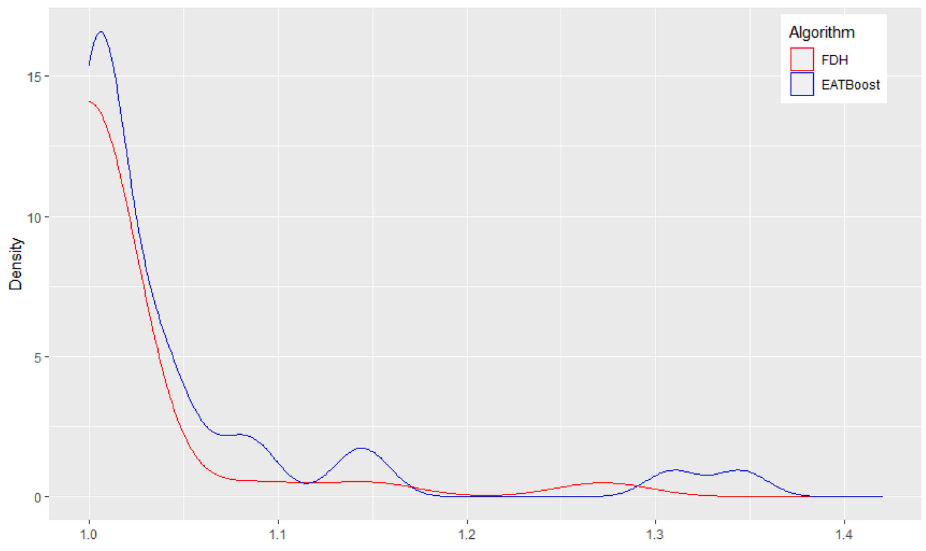

Fig. 3. Efficiency score distributions.

computational procedure. This feature contrasts sharply with the simplicity of the standard Free Disposal Hull technique, which is based on Mixed Integer Linear Programming.

We conclude by pointing out some possible future lines of research with respect to boosting and efficiency measurement. In this paper, we exclusively resorted to the output-oriented radial model to gauge technical efficiency. However, there are other alternatives in the literature on efficiency measurement. In this line, we suggest extending the new approach to the context of using different types of technical measures (Aparicio et al., 2018; Pastor et al., 2012) as, for example, the directional distance function (Chambers et al., 1998) or the weighted additive model (Lovell and Pastor, 1995; Aparicio et al., 2016). A clear future line of research could be the improvement in the computational time linked to the application of the EATBoost algorithm. In this regard, the parallelization of the assessment of the different values considered for the parameters (Q, J, v) could reduce the execution time of the algorithm. Another remarkable line of research could be adapting alternative boosting algorithms to the production context. In this regard, an interesting methodology to consider would be XGboost (eXtreme Gradient boosting) (Chen and Guestrin, 2016) for regression problems. Finally, applying the new methodology on real datasets could be a good avenue for further research.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We thank three anonymous reviewers and an associate editor for providing constructive comments and help in improving the contents and presentation of this paper. J. Aparicio and M. Esteve thank the grant PID2019-105952GB-I00 funded by Ministerio de Ciencia e Innovación/Agencia Estatal de Investigación/10.13039/501100011033. Additionally, M. D. Guillen and J. Aparicio thank the grant PROMETEO/2021/063 funded by the Valencian Community (Spain), which partially supported this work. Finally, M. Esteve thanks the grant FPU17/05365, supported by the Spanish Ministry of Science, Innovation and Universities.

References

Aparicio, J., Pastor, J. T., & Vidal, F. (2016). The weighted additive distance function. *European Journal of Operational Research, 254*(1), 338–346.
Aparicio, J., Cordero, J. M., Gonzalez, M., & Lopez-Espin, J. J. (2018). Using non-radial DEA to assess school efficiency in a cross-country perspective: An empirical analysis of OECD countries. *Omega, 79*, 9–20.
Baboota, R., & Kaur, H. (2019). Predictive analysis and modelling football results using machine learning approach for English Premier League. *International Journal of Forecasting, 35*(2), 741–755.
Banker, R. D., Charnes, A., & Cooper, W. W. (1984). Some models for estimating technical and scale inefficiencies in data envelopment analysis. *Management Science, 30*(9), 1078–1092.
Barbosa, F., Rampazzo, P. C. B., Yamakami, A., & Camanho, A. S. (2019). The use of frontier techniques to identify efficient solutions for the Berth Allocation Problem solved with a hybrid evolutionary algorithm. *Computers & Operations Research, 107*, 43–60.
Breiman, L. (2001). *Random forests. Machine learning, 45*(1), 5–32.
Breiman, L., Friedman, J., Olshen, R., & Stone, C. (1984). *Classification And Regression Trees*. Taylor & Francis.
Brown, I., & Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Systems with Applications, 39*(3), 3446–3453.
Carmona, P., Climent, F., & Momparler, A. (2019). Predicting failure in the US banking sector: An extreme gradient boosting approach. *International Review of Economics & Finance, 61*, 304–323.
Cazals, C., Florens, J. P., & Simar, L. (2002). Nonparametric frontier estimation: A robust approach. *Journal of Econometrics, 106*(1), 1–25.
Chambers, R. G., Chung, Y., & Färe, R. (1998). Profit, directional distance functions, and Nerlovian efficiency. *Journal of optimization theory and applications, 98*(2), 351–364.
Charles, V., Aparicio, J., & Zhu, J. (2020). *Data Science and Productivity Analytics*. Springer.
Charnes, A., Cooper, W. W., & Rhodes, E. (1978). Measuring the efficiency of decision making units. *European Journal of Operational Research, 2*(6), 429–444.
Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
Cordero, J. M., Díaz-Caro, C., Pedraja-Chaparro, F., & Tzeremes, N. G. (2021). A conditional directional distance function approach for measuring tax collection efficiency: Evidence from Spanish regional offices. *International Transactions in Operational Research, 28*(2), 1046–1073.
Daraio, C., & Simar, L. (2005). Introducing environmental variables in nonparametric frontier models: A probabilistic approach. *Journal of Productivity Analysis, 24*(1), 93–121.
Daraio, C., Simar, L., & Wilson, P. W. (2020). Fast and efficient computation of directional distance estimators. *Annals of Operations Research, 288*(2), 805–835.
Deprins, D., & Simar, L. (1984). *Measuring labor efficiency in post offices*. Concepts and Measurements, M. Marchand, P. Pestieau and H. Tulkens: The Performance of Public Enterprises.
Esteve, M., Aparicio, J., Rabasa, A., & Rodriguez-Sala, J. J. (2020). Efficiency analysis trees: A new methodology for estimating production frontiers through decision trees. *Expert Systems with Applications, 162*, Article 113783.
Esteve, M., Rodriguez-Sala, J. J., Lopez-Espin, J. J., & Aparicio, J. (2021). Heuristic and Backtracking Algorithms for Improving the Performance of Efficiency Analysis Trees. *IEEE Access, 9*, 17421–17428.
Färe, R., Grosskopf, S., & Lovell, C. K. (1985). *The measurement of efficiency of production*. Springer Science & Business Media.
Färe, R., & Primont, D. (1995). *Multiple-Output Production and Duality: Theory and Applications*. Boston: Kluwer Academic Publishers.
Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences, 55*(1), 119–139.
Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of statistics*, 1189–1232.
Guelman, L. (2012). Gradient boosting trees for auto insurance loss cost modeling and prediction. *Expert Systems with Applications, 39*(3), 3659–3667.
Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction*. Springer Science & Business Media.
Hew, K. F., Hu, X., Qiao, C., & Tang, Y. (2020). What predicts student satisfaction with MOOCs: A gradient boosting trees supervised machine learning and sentiment analysis approach. *Computers & Education, 145*, Article 103724.
Kearns, M. (1988). Thoughts on hypothesis boosting. Unpublished manuscript, 45, 105.
Kearns, M., & Valiant, L. (1994). Cryptographic limitations on learning Boolean formulae and finite automata. *Journal of the ACM (JACM), 41*(1), 67–95.
Kerstens, K., O'Donnell, C., & Van de Woestyne, I. (2019). Metatechnology frontier and convexity: A restatement. *European Journal of Operational Research, 275*(2), 780–792.
Kevork, I. S., Pange, J., Tzeremes, P., & Tzeremes, N. G. (2017). Estimating Malmquist productivity indexes using probabilistic directional distances: An application to the European banking sector. *European Journal of Operational Research, 261*(3), 1125–1140.
Khezrimotlagh, D., Zhu, J., Cook, W., & Toloo, M. (2019). Data envelopment analysis and big data. *European Journal of Operational Research, 274*(3), 1047–1054.
Landry, M., Erlinger, T. P., Patschke, D., & Varrichio, C. (2016). Probabilistic gradient boosting machines for GEFCom2014 wind forecasting. *International Journal of Forecasting, 32*(3), 1061–1066.
Lovell, C. K., & Pastor, J. T. (1995). Units invariant and translation invariant DEA models. *Operations research letters, 18*(3), 147–151.
Lu, H., Wang, H., & Yoon, S. W. (2019). A dynamic gradient boosting machine using genetic optimizer for practical breast cancer prognosis. *Expert Systems with Applications, 116*, 340–350.
Mastromarco, C., & Simar, L. (2015). Effect of FDI and time on catching up: New insights from a conditional nonparametric frontier analysis. *Journal of Applied Econometrics, 30*(5), 826–847.
Natekin, A., & Knoll, A. (2013). Gradient boosting machines, a tutorial. *Frontiers in neurorobotics, 7*, 21.
Pastor, J. T., Lovell, C. A., & Aparicio, J. (2012). Families of linear efficiency programs based on Debreu's loss function. *Journal of Productivity Analysis, 38*(2), 109–120.
Pereira, M. A., Figueira, J. R., & Marques, R. C. (2020). Using a Choquet integral-based approach for incorporating decision-maker's preference judgments in a data envelopment analysis model. *European Journal of Operational Research, 284*(3), 1016–1030.
Shephard, R. W. (1953). *Cost and production functions*. Princeton University Press.
Simar, L., & Vanhems, A. (2012). Probabilistic characterization of directional distances and their robust versions. *Journal of Econometrics, 166*(2), 342–354.
Simar, L., & Zelenyuk, V. (2006). On testing equality of distributions of technical efficiency scores. *Econometric Reviews, 25*(4), 497–522.
Tavakoli, I. M., & Mostafaee, A. (2019). Free disposal hull efficiency scores of units with network structures. *European Journal of Operational Research, 277*(3), 1027–1036.
Thaker, K., Charles, V., Pant, A., & Gherman, T. (2021). A DEA and random forest regression approach to studying bank efficiency and corporate governance. *Journal of the Operational Research Society*, 1–28.
Tsolas, I. E., Charles, V., & Gherman, T. (2020). Supporting better practice benchmarking: A DEA-ANN approach to bank branch performance assessment. *Expert Systems with Applications, 160*, Article 113599.
Tzeremes, N. G. (2015). Efficiency dynamics in Indian banking: A conditional directional distance approach. *European Journal of Operational Research, 240*(3), 807–818.
Xu, Y., & Goodacre, R. (2018). On splitting training and validation set: A comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. *Journal of analysis and testing, 2*(3), 249–262.
Zhang, B., Ren, J., Cheng, Y., Wang, B., & Wei, Z. (2019). Health data driven on continuous blood pressure prediction based on gradient boosting decision tree algorithm. *IEEE Access, 7*, 32423–32433.
Zhu, J. (2019). DEA under big data: Data enabled analytics and network data envelopment analysis. *Annals of Operations Research*, 1–23.

Apéndice B

Performance Evaluation of Decision-Making Units Through Boosting Methods in the Context of Free Disposal Hull: Some Exact and Heuristic Algorithms

International Journal of Information Technology & Decision Making
(2023)
© World Scientific Publishing Company
DOI: 10.1142/S0219622023500050

Performance Evaluation of Decision-Making Units Through Boosting Methods in the Context of Free Disposal Hull: Some Exact and Heuristic Algorithms

Maria D. Guillen*,‡, Juan Aparicio*,†,§ and Miriam Esteve*,¶

*Center of Operations Research (CIO)
Miguel Hernandez University of Elche (UMH)
03202 Elche (Alicante), Spain

†Valencian Graduate School and Research Network
of Artificial Intelligence (valgrAI), Spain
‡maria.guilleng@umh.es
§j.aparicio@umh.es
¶miriam.estevec@umh.es

Received 9 February 2022
Revised 5 December 2022
Accepted 7 December 2022
Published 31 January 2023

This paper aims to show how to calculate different efficiency measures using a technology estimator defined through the adaptation of the Gradient Tree Boosting algorithm. This adaptation shares some features with the standard nonparametric Free Disposal Hull (FDH) approach, but it overcomes data overfitting problems. Nevertheless, from a computational point of view, the new approach presents thousands of decision variables, making it difficult to solve. To tackle this problem, we also propose and check a heuristic approximation to the exact measures. To demonstrate the applicability of the proposed method, the exact and the heuristic approaches are compared through two empirical applications. The main contributions of this paper are as follows: we build a new bridge between machine learning techniques and technical efficiency measurement. In this framework, we show how to determine the output-oriented and input-oriented radial models, the Russell measure of output efficiency and the Russell measure of input efficiency, as well as the directional distance function and the Enhanced Russell Graph measure. We also prove that the new technique is better, in terms of bias and squared mean error, than the standard FDH technique. Furthermore, we show that the new approach may be seen as a possible remedy for solving the curse of dimensionality problem.

Keywords: Data envelopment analysis; free disposal hull; decision-making units; technical efficiency; boosting.

1. Introduction

The main purpose of efficiency analysis is to understand how inputs are translated into outputs. The literature on productive efficiency analysis and frontier estimation

§Corresponding author.

is extensive and growing, with hundreds of articles in applied economics, engineering, operations research, and statistics (as shown, for instance, in Ref. 1; where image fusion based on kernel estimation and data envelopment analysis is studied, Ref. 2; where a performance evaluation of Taiwanese hospitals by multi-activity network data envelopment analysis was carried out, and Ref. 3; where performance evaluation of real estate investment trusts was carried out by using a combination of modern approaches). From a methodological point of view, efficiency analysis aims to evaluate the technical efficiency of a set of entities, commonly known as Decision-Making Units (DMUs), which use multiple inputs to produce multiple outputs, whereby their performance in relation to the production possibility set, also called production technology, is compared. This unknown set must be estimated from sample data. In this context, both parametric and nonparametric methodologies have been developed to estimate the production frontiers. However, given that the specific mathematical formulation of the production function being estimated does not need to be identified under the nonparametric methodology, it is therefore more appealing than its parametric counterpart. Furthermore, nonparametric models can cope with multiple-output scenarios, in contrast to the parametric approach that aggregates the outputs into a single production index or attempts to model the technology using a dual cost function.[4]

With regard to nonparametric technologies, Farrell[5] was the first to introduce the concept of a piece-wise linear enveloping surface for multi-input single-output production contexts. In the same production framework, Afriat[6] showed how to determine a production function estimator on the assumption of several typical postulates in microeconomics (nondecreasing and concave) and admitted by the observations. Later, under a general multi-input multi-output production context, Charnes *et al.*[7] and Banker *et al.*[8] continued this line of research, leading to the Data Envelopment Analysis (DEA) technique being developed, in which the determination of the technology is only restricted by its axiomatic system, comprising convexity, free disposability, envelopment (i.e., including observations) and minimal extrapolation. In parallel, Deprins *et al.*[9] proposed the Free Disposal Hull (FDH) approach, which, unlike DEA, is based solely on free disposability, envelopment and minimal extrapolation. Other recent nonparametric techniques for estimating production frontiers include, for example, Simar and Wilson,[10–12] who showed how confidence intervals for efficiency scores could be determined through an adaptation of Efron's[13] bootstrapping methodology to the FDH and DEA framework.

Over the past few decades, numerous distinct technological efficiency metrics have been introduced into the efficiency measuring area. These efficiency measures use FDH and DEA technologies as a basis to define the optimization programs that allow the corresponding efficiency scores to be calculated, but differ in the paths selected to project each DMU onto the frontier. Some of these efficiency measures are the output-oriented and input-oriented Banker, Charnes and Cooper (BCC) radial models (see Ref. 8), based on equiproportional movements of technically inefficient observations to the estimated production frontier; the output-oriented and input-oriented Russell measures,[14] which also consider that there may be slacks in some outputs and/or inputs; and the directional distance function,[15] which projects the inefficient units onto the frontier in a pre-established direction.

Now, putting the spotlight on Machine Learning (ML) techniques in general, it is worth mentioning that gradient-boosting machines[16] are a type of sophisticated and modern ML techniques that have been very successful in a wide range of applications.[17] They are extremely configurable to the application's specific demands, thanks to being learned with respect to various loss functions. The main advantage of gradient-boosting machines is that, unlike the most frequent approach to data-driven models where only one single strong predictive model is created, an ensemble of relatively weak simple models is built instead. These models can be then further combined altogether to produce a better prediction. In contrast to other common ensemble approaches, such as Random Forest,[18] which rely on simple ensemble averaging, boosting approaches are based on a distinct, constructive ensemble-building strategy. In particular, a new weak model is trained with respect to the error of the entire ensemble learned so far at each iteration. The loss functions applied can be arbitrary, however if the error function is the classic squared-error loss, the learning approach will result in successive error-fitting.[19] Gradient Tree Boosting[20] is a modification of the standard Gradient Boosting algorithm to use Classification and Regression Analysis Trees[21] as the weak model.

Despite being a topic of growing academic interest, only a few papers devoted to the measurement of technical efficiency are based on Machine Learning techniques. In this regard, before going on, let us include a literature review on contributions related to technical efficiency measurement through the adaptation of ML techniques. We ruled out some papers where the authors considered the two worlds, i.e., efficiency evaluation and ML, but which simply applied each technique in a different stage of the analysis without really mixing them. Instead, a recent stream of the literature on efficiency frontiers is devoted to introducing new approaches, endowed with certain features of ML techniques. In this sense, Kuosmanen and Johnson[22,23] showed that DEA may be interpreted as nonparametric least-squares regression subject to shape constraints on the production frontier and sign constraints on residuals. These authors introduced the Corrected Concave Nonparametric Least Squares (CCNLS). Under this approach, many hyperplanes, *a priori* one for each observation, must be fitted with the data cloud while convexity of the generated production possibility set is guaranteed by the Afriat inequality constraints.[6] An advantage of CCNLS is that this technique performs well regarding bias and mean squared error in comparison with the standard DEA technique. Nevertheless, although concavity of the production function, which is equivalent to convexity of the technology (see p. 81 in Ref. 24), is a usual postulate in microeconomics, it is not always a valid assumption in practice. The technology might admit increasing returns to scale (i.e., outputs increase at a faster rate than inputs, which cannot be graphically modelled by concavity), or there might be lumpy goods (i.e., fractional values of inputs or outputs do not exist, which is incompatible with convexity). In this sense, stepwise estimators of the production frontier could yield more general and flexible estimators than piecewise linear estimators (see, for example, Ref. 25). However, CCNLS by Kuosmanen and Johnson focuses on piecewise linear estimators, which might not fit well with certain production contexts, as we have just pointed out. Additionally, Parmeter and Racine[26] introduced nonparametric kernel frontier estimators. As a benefit of the technique, let us mention that this approach provides a smooth approximation to the production frontier. However, this method is not able to deal with multi-output production contexts, which is a clear weakness from an empirical point of view. Daouia *et al.* (2016)[52] introduced a method that can be used for production function estimation through quadratic and cubic splines with shape constraints (monotonicity and/or concavity). The model is regularized by resorting to the determination of optimal knots in the input space, which is an advantage from a statistical point of view. However, this methodology is restricted to the single input–singe output case, which can be understood as a drawback. More recently, in Ref. 27, the Classification and Regression Trees (CART) technique was adapted for estimating efficient frontiers through step functions, so competing against the standard FDH technique; but avoiding the overfitting problem that FDH suffers because of the minimal extrapolation principle, which can be seen as an advantage in comparison with the standard FDH technique. Unfortunately, regression trees usually provide estimators with low bias and high variance; a feature that should be improved. Tsionas[28] introduced smooth monotone concave probabilistic regression trees for the estimation of efficiency. This author also indicates how to adapt the technique for dealing with panel data, which is a benefit of the approach. However, Tsionas considers the production of only one output and imposes concavity, restricting the flexibility of the frontier estimator. Valero-Carreras *et al.*[29] adapted Support Vector Machines to determine technical efficiency; introducing a new robust notion of epsilon-insensitivity, with interesting implications, but only in the single-output production framework. Guerrero *et al.*[30] introduced a methodology that allows for estimating polyhedral technologies following the Structural Risk Minimization (SRM) principle, which focuses on both the empirical and generalization error, which could be understood as an advantage from a machine-learning perspective. However, these authors impose convexity on the production possibility set, something that can be considered very demanding in practice (see Ref. 25). Finally, Olesen and Ruggiero[31,32] proposed the use of hinging hyperplanes as a flexible nonparametric representation of a production function, which is an interesting characteristic of this method. Regrettably, this last approach only considers the production of one output and can suffer from overfitting since no cross-validation method is applied.

In this paper, for the first time, we show how to measure technical efficiency by adapting the Gradient Tree Boosting algorithm by Friedman.[20] Our approach is based on the idea of using boosting to ensemble Efficiency Analysis Trees (EAT) models[27] as base learners, which are a direct adaptation of CART for estimating production technologies. The new algorithm will be called “EATBoosting”. The technology created by the EATBoosting algorithm will satisfy free disposability and envelops the observations from above. Additionally, it always contains the standard FDH technology as a subset. This feature will allow the EATBoosting algorithm to be compared to FDH through simulations and to show that EATBoosting outperforms FDH in terms of mean squared error and bias. However, from a computational point of view, the mixed-integer linear programs that solve the corresponding efficiency models based on the EATBoosting algorithm have thousands of variables and are extremely computationally complex due to the Random Access Memory (RAM) requirements. Accordingly, in this paper, we also propose a heuristic approximation to these problems, where both computational time and memory requirements are reduced. Regarding the contributions of this paper in comparison with the previous literature review, where we mentioned some pros and cons of the different approaches on ML and efficiency, we next list the most important ones. First, our approach considers the general multi-input multi-output production framework. In particular, we show how to implement in our context many of the usual efficiency measures traditionally utilized in Data Envelopment Analysis. Second, we apply a cross-validation strategy, which prevents the overfitting problems suffered by other approaches. Third, with respect to the recently-introduced technique known as Efficiency Analysis Trees,[27] our approach resorts to ensembles of trees that improve the performance of techniques based on fitting a single tree, as is the case of Esteve *et al.*'s method.

There are two main reasons for comparing the new technique with respect to the results obtained through the application of FDH rather than DEA. The first one is related to the nature of regression trees (specifically, Efficiency Analysis Trees), which is part of the basis of our approach. In this paper, we are going to show an adaptation of the Gradient Tree Boosting algorithm to the context of technical efficiency measurement. In particular, the Gradient Tree Boosting algorithm is based on regression trees, which yield stepwise estimators of the target functions. Consequently, the graphical relationship between FDH, which also produces stepwise estimators of efficient frontiers in the production context, and the new technique is apparent. The second reason is that FDH has been recently highlighted in the literature as an appealing technique, even in comparison with DEA, due to its flexibility in not having to assume convexity (see the recent papers of Refs. 33–35).

Finally, to contextualize our findings, let us mention some recent and interesting contributions related to this paper's research field. In this regard, Kou *et al.*[36] evaluated financial technology-based investments of European banking services with an application of an original methodology that considers interval type-2 (IT2) fuzzy decision-making trial and evaluation laboratory and IT2 fuzzy Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS) models. Kou *et al.*[37] generated inventive problem-solving map of innovative carbon emission reduction strategies for transportation investment projects. In that paper, an extension of group decision-making (GDM) and spherical fuzzy numbers was also proposed regarding solar energy projects. Chao *et al.*[38] introduced efficiency measures into the consensus reaching process of GDM. From the perspective of efficiency, these authors proposed a benchmark in consensus reaching by DEA without explicit input benchmark models. Li *et al.*[39] introduced a consensus-reaching process approach for large-scale GDM based on bounded confidence and social network to manage experts' opinions. Additionally, Panwar *et al.*[40] recently reviewed the main contributions that combines DEA and machine learning. Most of them consider the two fields (efficiency evaluation and machine learning) without really mixing them, by simply applying each method at a different and independent stage of the analysis. In contrast, our approach adapts a standard machine learning technique, as Gradient Tree Boosting, to endow it with shape constraints coming from production theory. Moreover, Panwar *et al.*[40] concluded that: "Integration of DEA with Machine Learning models is one of the emerging areas of DEA which may have a lot of potential in the coming years due to the increase in volume of data, availability of good computing facilities and increase in the complexities of real-world problems." In this regard, the approach that we introduce in this paper builds a new bridge between frontier analysis and machine learning; and can be included in one of the emerging areas of DEA highlighted by Panwar *et al.*[40]

The remainder of the paper is organized as follows. Section 2 introduces FDH and its related efficiency measures, as well as Gradient Boosting and Efficiency Analysis Trees. In Sec. 3, we show how to measure technical efficiency through Gradient Boosting, resorting to exact and heuristic methods. Performance of the heuristic method is investigated via simulations in Sec. 4, while, in Sec. 5, a comparison of the differences between the exact and the heuristic approaches is made using two empirical applications. Finally, Sec. 6 gives concluding remarks and establishes future research lines.

2. Background

This section briefly summarizes the main concepts related to FDH, the different measures of technical efficiency, the standard Gradient Boosting and the recent EAT algorithm.[27] Regarding notation, we use nonbold for scalars and bold for vectors. Additionally, let $\boldsymbol{z} = (z^{(1)}, \ldots, z^{(g)})$ and $\hat{\boldsymbol{z}} = (\hat{z}^{(1)}, \ldots, \hat{z}^{(g)})$, from now on, $\boldsymbol{z} \leq \hat{\boldsymbol{z}}$ indicates $z^{(j)} \leq \hat{z}^{(j)}$ for all $j = 1, \ldots, g$, and $\boldsymbol{z} < \hat{\boldsymbol{z}}$ means that $z^{(j)} < \hat{z}^{(j)}$ for all $j = 1, \ldots, g$.

2.1. *FDH and its related efficiency measures*

Given a set of n DMUs to be assessed, where DMU$_i$ consumes $\boldsymbol{x}_i = (x_i^{(1)}, \ldots, x_i^{(m)}) \in R_+^m$ amounts of inputs to produce $\boldsymbol{y}_i = (y_i^{(1)}, \ldots, y_i^{(s)}) \in R_+^s$ amounts of outputs, $i = 1, \ldots, n$, the relative efficiency of each DMU in the sample must be evaluated in reference to the production technology, also known as the production possibility set. The production technology contains all technically feasible combinations of $(\boldsymbol{x}, \boldsymbol{y})$ and is formally defined as

$$\Psi = \{(\boldsymbol{x}, \boldsymbol{y}) \in R_+^{m+s} : \boldsymbol{x} \text{ can produce } \boldsymbol{y}\}. \tag{1}$$

The efficient frontier of Ψ, defined as $\partial(\Psi) := \{(\boldsymbol{x}, \boldsymbol{y}) \in \Psi : \hat{\boldsymbol{x}} < \boldsymbol{x}, \hat{\boldsymbol{y}} > \boldsymbol{y} \Rightarrow (\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}}) \notin \Psi\}$ comprises part of the boundary of the technology. Technical inefficiency is thus established as the distance to this boundary from any interior point. Notwithstanding, there are numerous potential paths to the frontier from any point located inside Ψ. Each path is associated with a particular technical efficiency measure. Indeed, in the literature, the output-oriented radial model stands out as one of the most extensively used technical efficiency measures.[8,7] The latter conditions the efficiency score for the point being evaluated $(\boldsymbol{x}_k, \boldsymbol{y}_k)$ by maintaining inputs constant while increasing all outputs equiproportionately:

$$\phi(\boldsymbol{x}_k, \boldsymbol{y}_k) = \max\{\phi : (\boldsymbol{x}_k, \phi\boldsymbol{y}_k) \in \Psi\}. \tag{2}$$

FDH was introduced by Deprins *et al.*[9] as a technique for estimating production frontiers and technical efficiency based upon a few axioms: free disposability [if $(\boldsymbol{x}, \boldsymbol{y}) \in \Psi$, $\boldsymbol{x}' \geq \boldsymbol{x}$ and $\mathbf{0} \leq \boldsymbol{y}' \leq \boldsymbol{y}$, then $(\boldsymbol{x}', \boldsymbol{y}') \in \Psi$] and deterministicness [$(\boldsymbol{x}_i, \boldsymbol{y}_i) \in \Psi$ for all $i = 1, \ldots, n$]. In line with Ref. 9, the FDH estimator of Ψ is defined as

$$\hat{\Psi}_{\text{FDH}} = \{(\boldsymbol{x}, \boldsymbol{y}) \in R_+^{m+s} : \exists i = 1, \ldots, n \text{ such that } \boldsymbol{y} \leq \boldsymbol{y}_i, \boldsymbol{x} \geq \boldsymbol{x}_i\}. \tag{3}$$

We can estimate the efficiency score $\phi(\boldsymbol{x}_k, \boldsymbol{y}_k)$ by plugging $\hat{\Psi}_{\text{FDH}}$ into (2) instead of Ψ. This substitution allows the estimation to be determined by the following mixed-integer linear optimization program:

$$\begin{aligned}
\hat{\phi}^{\text{FDH}}(\boldsymbol{x}_k, \boldsymbol{y}_k) = \quad & \max \phi \\
& \text{s.t.} \\
& \sum_{i=1}^{n} \lambda_i x_i^{(p)} \leq x_k^{(p)}, \quad p = 1, \ldots, m \\
& \sum_{i=1}^{n} \lambda_i y_i^{(l)} \geq \phi y_k^{(l)}, \quad l = 1, \ldots, s \\
& \sum_{i=1}^{n} \lambda_i = 1, \\
& \lambda_i \in \{0, 1\}, \qquad i = 1, \ldots, n
\end{aligned} \tag{4}$$

Other possible efficiency measures are the input-oriented radial model,[7,8] the Russell measure of output efficiency and the Russell measure of input efficiency,[14] the directional distance function[15] and the Enhanced Russell Graph measure.[41] See more efficiency measures in Ref. 42.

2.2. *Efficiency analysis trees*

Efficiency Analysis Trees (EAT) by Esteve *et al.*[27,a] is a CART-based machine learning technique for estimating production frontiers. Similar to CART, in the EAT algorithm, a tree structure is built through binary recursive partitioning. In each

[a] See also the R package eat: https://cran.r-project.org/web/packages/eat/index.html.

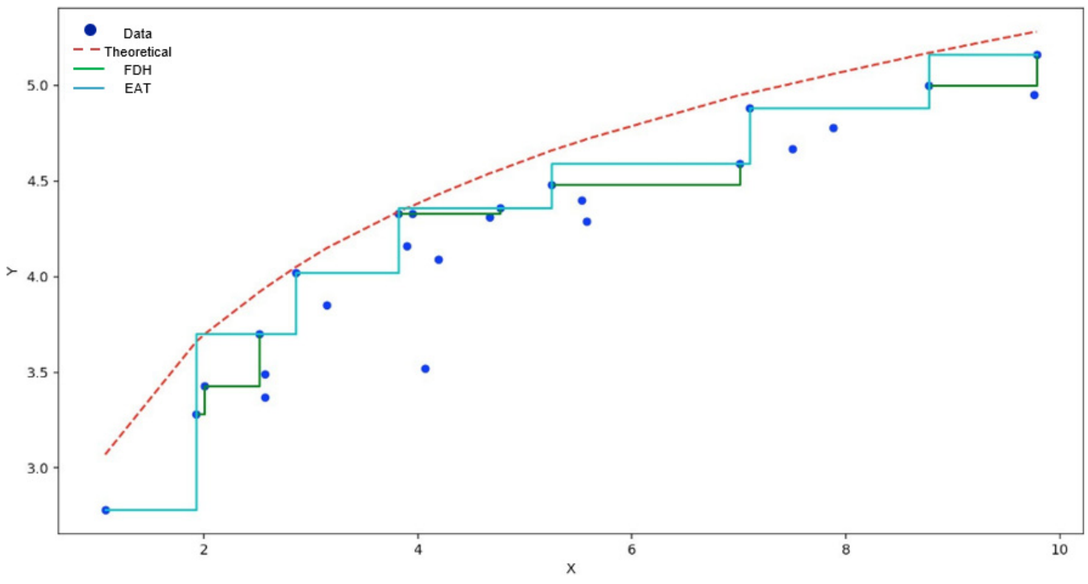

Fig. 1. Example of theoretical, FDH and EAT frontiers.[43]

step, the algorithm needs to select an input variable as well as a threshold, with the objective of splitting the data contained in a node of the tree into two new child nodes, such that the sum of the Mean Square Error (MSE) is minimized.

The tree generated by this process, denominated as deep EAT, suffers from overfitting. To overcome this weakness, Esteve *et al.*[27] proposed to prune the deep tree through a process that is based on an error-complexity measure; generating the final EAT tree and the final predictor of the output variable. Fig. 1 shows an example of the differences between the FDH frontier, which is closer to the data, and the EAT frontier, which is closer to the theoretical function. The EAT approach has the advantage that it generates a frontier that satisfies free disposability, but it does not suffer from overfitting[43]; something that does happen with FDH. See Ref. 27 for more details on the EAT algorithm.

2.3. *Gradient boosting*

Gradient boosting is a machine learning algorithm based on the ensemble of multiple models.[17] The basic principles of gradient boosting are as follows: given a loss function $L(\boldsymbol{y}, f(\boldsymbol{x}))$ (e.g., MSE for regression) and a weak learner $h(\boldsymbol{x})$ (a model that may only have slightly better classification abilities than random guessing), the algorithm seeks to find an additive model that minimizes the loss function. The algorithm is typically initialized with the best guess of the response ($f_0(\boldsymbol{x}) = \arg\min_\gamma \sum_{i=1}^n L(\boldsymbol{y}_i, \gamma)$), for example, the mean of the response in regression. Then, the gradient of pseudo-residuals is calculated ($\boldsymbol{r}_q = -[\frac{\partial L(\boldsymbol{y}_i, f(\boldsymbol{x}_i))}{\partial f(\boldsymbol{x}_i)}]_{f=f_{q-1}}$), and a model is fitted to those residuals to minimize the loss function ($\gamma_q = \arg\min_\gamma \sum_{i=1}^n L(\boldsymbol{y}_i, f_{q-1}(\boldsymbol{x}_i) + \gamma h_q(\boldsymbol{x}_i))$). This model is added to the previous model ($f_q(\boldsymbol{x}) = f_{q-1}(\boldsymbol{x}) + \gamma_q h_q(\boldsymbol{x})$), and the procedure continues for a finite set of iterations $q = 1, 2, \ldots, Q$.[44] To reduce overfitting, shrinkage is introduced as a regularization technique by scaling the contribution of each tree by a factor (known as learning rate) $0 < v < 1$.[19] This means replacing the updating rule by $f_q(\boldsymbol{x}) = f_{q-1}(\boldsymbol{x}) + v\gamma_q h_q(\boldsymbol{x})$, $0 < v < 1$.

Gradient boosting is typically used with CART of a fixed size as base learners. For this case, Friedman[20] proposes a variation to the gradient-boosting method called Gradient Tree Boosting, which improves the quality of fit of each base learner. In particular, as the tree splits the predictor space into J_q disjoint regions (supports), $R_{1q}, \ldots, R_{J_q q}$, instead of choosing a single γ_q for the whole tree, Friedman suggests modifying this algorithm so that it selects a separate optimal value γ_{jq} for each of the tree's regions. As a result, the model updating rule becomes $f_q(\boldsymbol{x}) = f_{q-1}(\boldsymbol{x}) + v\sum_{j=1}^{J_q} \gamma_{jq} I(\boldsymbol{x} \in R_{jq})$, where $I(\boldsymbol{x} \in R_{jq}) = \left\{\begin{matrix} 1 \text{ if } \boldsymbol{x} \in R_{jq} \\ 0 \text{ if } \boldsymbol{x} \notin R_{jq} \end{matrix}\right\}$. The process is executed Q times. As a result of all these steps, Algorithm 1 is derived for the standard Gradient Tree Boosting (see Ref. 19):

Algorithm 1. Pseudo code for Gradient tree boosting.

```
$Q, J_q, v$ // Hyperparameters of the algorithm
begin
    $f_0(\boldsymbol{x}) := \arg\min_\gamma \sum_{i=1}^{n} L(\boldsymbol{y}_i, \gamma)$ // Initialize with the best guess
    for $q := 1$ to $Q$ :
        for $i := 1$ to $n$ :
            $r_{iq} := -[\frac{\partial L(\boldsymbol{y}_i, f(\boldsymbol{x}_i))}{\partial f(\boldsymbol{x}_i)}]_{f=f_{q-1}}$ // Calculate the pseudo-residuals
        end for
        Fit a regression tree (without pruning) to the targets $r_{iq}$ given terminal
        regions $R_{jq}$, $j = 1, 2, \ldots, J_q$
        for $j := 1$ to $J_q$:
            $\gamma_{jq} := \arg\min_\gamma \sum_{\boldsymbol{x}_i \in R_{jm}}^{L(\boldsymbol{y}_i, f_{q-1}(\boldsymbol{x}_i)+\gamma)}$ // Prediction for each terminal
          region
        end for
        $f_q(\boldsymbol{x}) = f_{q-1}(\boldsymbol{x}) + v\sum_{j=1}^{J_q} \gamma_{jq} I(\boldsymbol{x} \in R_{jq})$ // Update prediction
    end for
    return $\hat{f}(\boldsymbol{x}) := f_Q(\boldsymbol{x})$ // Final estimation
end
```

3. Measuring Technical Efficiency Through Gradient Tree Boosting

This section is devoted to the adaptation of the Gradient Tree Boosting algorithm[20] to estimate production technologies while satisfying the axiomatic requisites, linked to production theory, at the same time.

By way of an overview of the new methodology that we introduce in this section, let us mention that, while the standard Gradient Tree Boosting for regression resorts to CART to determine tree structures at each step of the algorithm, we will suggest using EAT (as the base learner) to create tree structures in the new approach, where the aim is to determine a production frontier estimation. This fact will allow us to prove that the new method yields production frontier estimators capable of satisfying basic assumptions in microeconomics, such as free disposability. Otherwise, the use of CART does not guarantee the satisfaction of certain postulates from production theory, since the predictor surface generated by this technique through Gradient Tree Boosting is not conditioned to these shape constraints. Next, we will delve into the details of the new approach, which we will call the EATBoosting algorithm.

Within the adaptation process of the standard Gradient Tree Boosting technique to the production theory field, we first must guarantee that the initial estimation of the outputs envelops all the observations from above (i.e., $\boldsymbol{f}_0(\boldsymbol{x}) \geq \boldsymbol{y}_i$, for all $i = 1, \ldots, n$). In this regard, the estimation for initializing the boosting algorithm is defined in our case as $\boldsymbol{f}_0(\boldsymbol{x}) = (\max_{i=1,\ldots,n}\{y_i^{(1)}\}, \ldots, \max_{i=1,\ldots,n}\{y_i^{(s)}\})$. In addition, as the EAT method resorts to the Mean Squared Error for node splitting, the loss function used in the EATBoosting algorithm is naturally defined as $L(\boldsymbol{y}_i, \boldsymbol{f}(\boldsymbol{x}_i)) = \frac{1}{2}\sum_{l=1}^{s}(\boldsymbol{y}_i^{(l)} - \hat{\boldsymbol{f}}^{(l)}(\boldsymbol{x}_i))^2, \forall\, i = 1, \ldots, n$. Thus, the components of the gradient $\boldsymbol{r}_q$ are now $r_{iq}^{(l)} = -[\frac{\partial L(\boldsymbol{y}_i, \boldsymbol{f}(\boldsymbol{x}_i))}{\partial \boldsymbol{f}^{(l)}(\boldsymbol{x}_i)}]_{\boldsymbol{f}^{(l)}(\boldsymbol{x}_i) = \boldsymbol{f}_{q-1}^{(l)}(\boldsymbol{x}_i)} = \boldsymbol{y}_i^{(l)} - \boldsymbol{f}^{(l)}(\boldsymbol{x}_i)$, $l = 1, \ldots, s$. In each iteration, a new EAT model $h_q(\boldsymbol{x})$ is fitted to the components of the negative gradient of the loss function in the previous one. In this case, the vector $\boldsymbol{\gamma}_{jq}$ that minimizes the loss function over the region R_{jq} corresponds to the estimation of the outputs generated by EAT for the final node associated with R_{jq} (i.e., $\boldsymbol{\gamma}_{jq} = \hat{\boldsymbol{y}}_{T(\aleph_q)}(R_{jq})$, where $\hat{\boldsymbol{y}}_{T(\aleph_q)}(R_{jq})$ denotes the multi-output predictor yielded by the EAT technique in the step q of the algorithm for any input profile that belongs to the region R_{jq}). As a result, Algorithm 2 is obtained as the natural adaptation of the standard Gradient Tree Boosting algorithm (Algorithm 1) to estimate production frontiers.

In the above algorithm, the hyperparameters to be tuned by, for example, cross-validation are Q, i.e., the number of iterations, J_q, i.e., the sizes of each of the trees for $q = 1, 2, \ldots, Q$, and $0 < v < 1$, i.e., the learning rate.

The algorithm is graphically illustrated through a flowchart in Fig. 2.

Moreover, let $\hat{\boldsymbol{f}}(\boldsymbol{x})$ be the vector of estimated outputs from the input profile $\boldsymbol{x} \in R_+^m$ after executing the Algorithm 2. Then, the technology or production possibility set induced by the algorithm could be established as follows:

$$\hat{\Psi}_B = \{(\boldsymbol{x}, \boldsymbol{y}) \in R_+^{m+s} : \boldsymbol{y} \leq \hat{\boldsymbol{f}}(\boldsymbol{x})\}. \tag{5}$$

The set $\hat{\Psi}_B$, just as $\hat{\Psi}_{\text{FDH}}$, satisfies: (i) the free disposability axiom with regard to inputs/outputs, and (ii) any observation $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ belongs to the set; since these

Algorithm 2. Pseudo code for the EATBoosting algorithm.

```
$Q, J_q, v$ // Hyperparameters of the algorithm
begin
    $\boldsymbol{f}_0(\boldsymbol{x}) = (\max_{i=1,\dots,n}\{y_i^{(1)}\}, \dots, \max_{i=1,\dots,n}\{y_i^{(s)}\})$ // Initialize with the maximum for each output
    for $q := 1$ to $Q$
        for $i := 1$ to $n$ :
            $\boldsymbol{r}_{iq} = \boldsymbol{y}_i - \boldsymbol{f}_{q-1}(\boldsymbol{x})$// Calculate the pseudo-residuals
        end for
        Fit an EAT tree (without pruning) to the targets $\boldsymbol{r}_{iq}$ given terminal regions $R_{jq}$, $j = 1, 2, \dots, J_q$
        for $j := 1$ to $J_q$:
            $\boldsymbol{\gamma}_{jq} = \hat{\boldsymbol{y}}_{T(\aleph_q)}(R_{jq})$ // Prediction for each terminal region
        end for
        $\boldsymbol{f}_q(\boldsymbol{x}) = \boldsymbol{f}_{q-1}(\boldsymbol{x}) + v\sum_{j=1}^{J_q}\boldsymbol{\gamma}_{jq}I(\boldsymbol{x} \in R_{jq})$ // Update prediction
    end for
    return $\hat{\boldsymbol{f}}(\boldsymbol{x}) := \boldsymbol{f}_Q(\boldsymbol{x})$ // Final estimation
end
```

properties are inherited from the application of the EAT technique at each step of the algorithm. Furthermore, $\hat{\Psi}_{\text{FDH}} \subseteq \hat{\Psi}_B$ because the EATBoosting technique satisfies the same axioms as FDH except one: minimal extrapolation.

3.1. *Efficiency measures*

At this point, we know the expression of the technology estimator determined by the adaptation of the Gradient Tree Boosting algorithm. However, as we showed in Sec. 2.1, many measures of technical efficiency have been defined in the literature and we need to know how to implement each of them under the new approach. In this section, we introduce a way of calculating any measure of technical efficiency using the technology estimated by boosting as a basis, as follows.

In the tree structure of each individual EAT tree in step q of Algorithm 2, each leaf node j, $j = 1, \dots, J_q$, is defined by the fulfillment of a series of conditions in the input space of the type $\{x_p < \delta_p\}$ or $\{x_p \geq \delta_p\}$, where p is the input used to make the split and $\delta_p \in \Delta_p$, where Δ_p is the set of possible thresholds for input p. Therefore, after executing all the splits and getting the final tree structure for the tree corresponding to step q, the support of its leaf node j has the format: $R_{jq} = \{\boldsymbol{x} \in R_+^m : a_{jq}^{(p)} \leq x^{(p)} < b_{jq}^{(p)}, p = 1, \dots, m\}$. The parameter $a_{jq}^{(p)}$ could be zero, whereas the parameter $b_{jq}^{(p)}$ could be $+\infty$. After executing the EATBoosting algorithm, the input space is split into disjoint subsets resulting from the intersection of all the input space partitions associated with each individual EAT-based tree. This final partition is made up of supports such that $\hat{R}_{j_1 \cdots j_Q} = \bigcap_{q=1}^{Q} R_{j_q q}$, for all

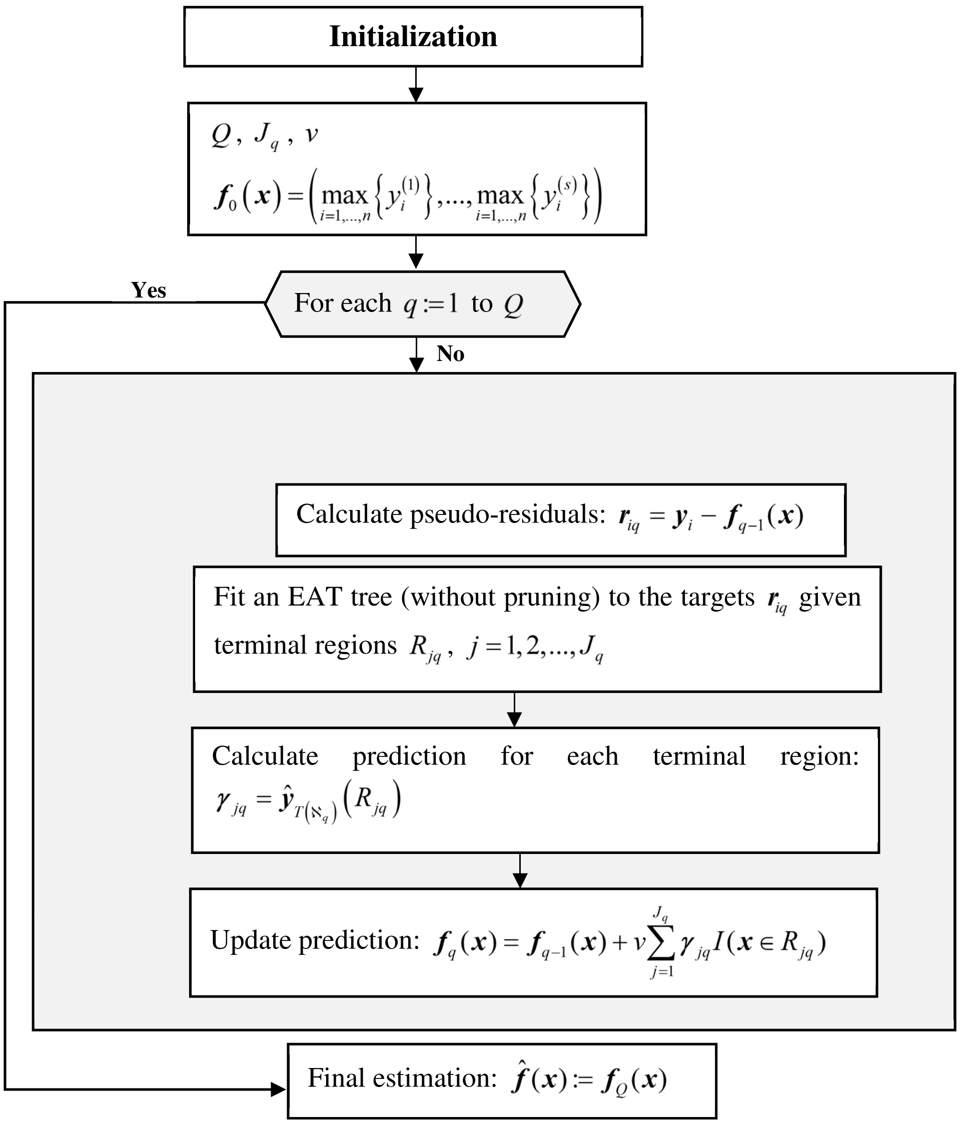
Initialization
Q, J_q, v
$f_0(x) = \left(\max_{i=1,\dots,n}\{y_i^{(1)}\},\dots,\max_{i=1,\dots,n}\{y_i^{(s)}\}\right)$
Yes
For each $q := 1$ to Q
No
Calculate pseudo-residuals: $r_{iq} = y_i - f_{q-1}(x)$
Fit an EAT tree (without pruning) to the targets r_{iq} given terminal regions R_{jq}, $j = 1,2,\dots,J_q$
Calculate prediction for each terminal region: $\gamma_{jq} = \hat{y}_{T(\aleph_q)}(R_{jq})$
Update prediction: $f_q(x) = f_{q-1}(x) + v\sum_{j=1}^{J_q}\gamma_{jq} I(x \in R_{jq})$
Final estimation: $\hat{f}(x) := f_Q(x)$

Fig. 2. Flowchart of the EATBoosting algorithm.

$j_q = 1, \dots, J_q$, and for all $q = 1, \dots, Q$. In Proposition 1, it is proved that the final supports $\hat{R}_{j_1 \cdots j_Q}$ can be rewritten in the same format as any support of an individual tree.

Proposition 1. *$\hat{R}_{j_1 \cdots j_Q} = \{\boldsymbol{x} \in R_+^m : \hat{a}_{j_1 \cdots j_Q}^{(p)} \leq x^{(p)} < \hat{b}_{j_1 \cdots j_Q}^{(p)}, p = 1, \dots, m\}$ with $\hat{a}_{j_1 \cdots j_Q}^{(p)} = \max_{1 \leq q \leq Q}\{a_{j_q q}^{(p)}\}$ and $\hat{b}_{j_1 \cdots j_Q}^{(p)} = \min_{1 \leq q \leq Q}\{b_{j_q q}^{(p)}\}$, $j_q = 1, \dots, J_q$, $q = 1, \dots, Q$.*

Proof. Let $A = \{\boldsymbol{x} \in R_+^m : \hat{a}_{j_1 \cdots j_Q}^{(p)} \leq x^{(p)} < \hat{b}_{j_1 \cdots j_Q}^{(p)}, p = 1, \dots, m\}$. If $\boldsymbol{x} \in \hat{R}_{j_1 \cdots j_Q} = \bigcap_{q=1}^{Q} R_{j_q q}$, then $\boldsymbol{x} \in R_{j_q q}$, $\forall\, q = 1, \dots, Q$, which implies that $a_{j_q q}^{(p)} \leq x^{(p)} < b_{j_q q}^{(p)}$, $\forall\, p = 1, \dots, m$ and $\forall\, q = 1, \dots, Q$. Consequently, $\max_{1 \leq q \leq Q}\{a_{j_q q}^{(p)}\} \leq x^{(p)} < \min_{1 \leq q \leq Q}$ $\{b_{j_q q}^{(p)}\}$. Hence, $\boldsymbol{x} \in A$. Now, if $\boldsymbol{x} \in A$, then $\hat{a}_{j_1 \cdots j_Q}^{(p)} \leq x^{(p)} < \hat{b}_{j_1 \cdots j_Q}^{(p)}$, $\forall\, p = 1, \dots, m$.

This means that $\max_{1\leq q\leq Q}\{a^{(p)}_{j_q q}\} \leq x^{(p)} < \min_{1\leq q\leq Q}\{b^{(p)}_{j_q q}\}$, $\forall\, p = 1,\dots,m$, which implies that $a^{(p)}_{j_q q} \leq x^{(p)} < b^{(p)}_{j_q q}$, $\forall\, p = 1,\dots,m$ and $\forall\, q = 1,\dots,Q$. Then, by the definition of each set $R_{j_q q}$, we have that $\boldsymbol{x} \in \bigcap_{q=1}^{Q} R_{j_q q} = \hat{R}_{j_1\cdots j_Q}$.

The similarities between standard FDH and the set $\hat{\Psi}_B$, derived from the EAT-Boosting algorithm, can be summed up by the following result, which determines that the FDH technology produced by a particular "virtual" learning sample concurs with the estimation of the underlying technology given by the boosting algorithm. This virtual or unobserved learning sample is as follows: $\{(\hat{\boldsymbol{a}}_{j_1\cdots j_Q}, \hat{\boldsymbol{f}}(\hat{\boldsymbol{a}}_{j_1\cdots j_Q}))\}$, $\forall\, j_q = 1,\dots,J_q$, $\forall\, q = 1,\dots,Q$. In other words, they represent "virtual" units, meaning they may not be observed in the original learning sample, which consumes the input vector $\hat{\boldsymbol{a}}_{j_1\cdots j_Q}$, corresponding to the bottom corner of the support $\hat{R}_{j_1\cdots j_Q}$, and produces the output vector $\hat{\boldsymbol{f}}(\hat{\boldsymbol{a}}_{j_1\cdots j_Q})$, corresponding to the EATBoosting estimation of the set of outputs for input vector $\hat{\boldsymbol{a}}_{j_1\cdots j_Q}$. This result is formally set out in Proposition 2. □

Proposition 2. *The FDH technology constructed from the set of points* $\{(\hat{\boldsymbol{a}}_{j_1\cdots j_Q}, \hat{\boldsymbol{f}}(\hat{\boldsymbol{a}}_{j_1\cdots j_Q}))\}_{\substack{j_q = 1,\dots,J_q \\ q = 1,\dots,Q}}$ *coincides with* $\hat{\Psi}_B$:

$$\hat{\Psi}_B = \{(\boldsymbol{x},\boldsymbol{y}) \in R^{m+s}_{+} : \exists (j_1,\dots,j_Q), j_q = 1,\dots,J_q,\ q = 1,\dots,Q, \text{ such that } \boldsymbol{y} \leq \hat{\boldsymbol{f}}(\hat{\boldsymbol{a}}_{j_1\cdots j_Q}), \boldsymbol{x} \geq \hat{\boldsymbol{a}}_{j_1\cdots j_Q}\}. \tag{6}$$

Proof. Let us denote the right-hand side of expression (6) as $\Psi^{\text{EATBoost}}_{\text{FDH}}$. Let $(\boldsymbol{x},\boldsymbol{y}) \in \hat{\Psi}_B$. Then, by definition, $\boldsymbol{y} \leq \hat{\boldsymbol{f}}(\boldsymbol{x})$. Let $\hat{R}_{\hat{j}_1\cdots \hat{j}_Q}$ such that $\boldsymbol{x} \in \hat{R}_{\hat{j}_1\cdots \hat{j}_Q}$. Moreover, $\hat{\boldsymbol{f}}(\boldsymbol{x}') = \hat{\boldsymbol{f}}(\boldsymbol{x})$ for all $\boldsymbol{x}' \in \hat{R}_{\hat{j}_1\cdots \hat{j}_Q}$. Then, $\hat{\boldsymbol{f}}(\hat{\boldsymbol{a}}_{\hat{j}_1\cdots \hat{j}_Q}) = \hat{\boldsymbol{f}}(\boldsymbol{x})$ since $\hat{\boldsymbol{a}}_{\hat{j}_1\cdots \hat{j}_Q} \in \hat{R}_{\hat{j}_1\cdots \hat{j}_Q}$. Additionally, we have that $\hat{\boldsymbol{a}}_{\hat{j}_1\cdots \hat{j}_Q} \leq \boldsymbol{x}$ because $\boldsymbol{x} \in \hat{R}_{\hat{j}_1\cdots \hat{j}_Q}$. Consequently, $\exists (j_1,\dots,j_Q), j_q = 1,\dots,J_q, q = 1,\dots,Q$, such that $\boldsymbol{y} \leq \hat{\boldsymbol{f}}(\hat{\boldsymbol{a}}_{j_1\cdots j_Q})$ and $\boldsymbol{x} \geq \hat{\boldsymbol{a}}_{j_1\cdots j_Q}$. Therefore, $(\boldsymbol{x},\boldsymbol{y}) \in \Psi^{\text{EATBoost}}_{\text{FDH}}$. Let now $(\boldsymbol{x},\boldsymbol{y}) \in \Psi^{\text{EATBoost}}_{\text{FDH}}$. Then, $\exists (j_1,\dots,j_Q), j_q = 1,\dots,J_q, q = 1,\dots,Q$, such that $\boldsymbol{y} \leq \hat{\boldsymbol{f}}(\hat{\boldsymbol{a}}_{j_1\cdots j_Q})$ and $\boldsymbol{x} \geq \hat{\boldsymbol{a}}_{j_1\cdots j_Q}$. As we know that $\hat{\Psi}_B$ satisfies free disposability, we have that if $\boldsymbol{x}' \geq \boldsymbol{x}$, then $\hat{\boldsymbol{f}}(\boldsymbol{x}') \geq \hat{\boldsymbol{f}}(\boldsymbol{x})$. Hence, $\hat{\boldsymbol{f}}(\boldsymbol{x}) \geq \hat{\boldsymbol{f}}(\hat{\boldsymbol{a}}_{j_1\cdots j_Q})$. Then, we have $\boldsymbol{y} \leq \hat{\boldsymbol{f}}(\hat{\boldsymbol{a}}_{j_1\cdots j_Q}) \leq \hat{\boldsymbol{f}}(\boldsymbol{x})$, which implies that, by the definition of $\hat{\Psi}_B$, $(\boldsymbol{x},\boldsymbol{y}) \in \hat{\Psi}_B$. □

Proposition 2 has more significant implications. This result demonstrates how any measure of technical efficiency based on the estimator $\hat{\Psi}_B$ can be calculated. In our case, and thanks to the link between the FDH and the EATBoosting technique, by analogy with the FDH model of Eq. (4), the mixed-integer linear program for calculating the output-oriented radial model would be as follows:

$$\begin{array}{rll} \hat{\phi}^{\text{EATBoost}}(\boldsymbol{x}_k, \boldsymbol{y}_k) = & \max\ \phi & \\ & \text{s.t.} & \\ & \displaystyle\sum_{j_1=1}^{J_1} \cdots \sum_{j_Q=1}^{J_Q} \lambda_{j_1 \cdots j_Q} \hat{a}^{(p)}_{j_1 \cdots j_Q} \leq x^{(p)}_k, & p = 1, \ldots, m \\ & \displaystyle\sum_{j_1=1}^{J_1} \cdots \sum_{j_Q=1}^{J_Q} \lambda_{j_1 \cdots j_Q} \hat{f}^{(l)}(\hat{\boldsymbol{a}}_{j_1 \cdots j_Q}) \geq \phi y^{(l)}_k, & l = 1, \ldots, s \\ & \displaystyle\sum_{j_1=1}^{J_1} \cdots \sum_{j_Q=1}^{J_Q} \lambda_{j_1 \cdots j_Q} = 1, & \\ & \lambda_{j_1 \cdots j_Q} \in \{0, 1\}, & j_q = 1, \ldots, J_q, \\ & & q = 1, \ldots, Q \end{array} \tag{7}$$

As we can see, to obtain Eq. (7), $\hat{\Psi}_B$ is plugged into (2) instead of Ψ, unlike in (4) where $\hat{\Psi}_{FDH}$ was used. That is, instead of plugging the DMUs that constitute the learning sample into (2), the "virtual" units are now used. As a result, the number of λ variables is now much bigger. More specifically, if we consider the same size for all trees ($J_q = J, \forall\, q$), then instead of n variables (just as many as DMUs), there are now J^Q.

In the same way, in the context of EATBoosting, the input-oriented radial model can be determined by solving the following mixed-integer linear program:

$$\begin{array}{rll} \hat{\theta}^{\text{EATBoost}}(\boldsymbol{x}_k, \boldsymbol{y}_k) = & \min \theta & \\ & \text{s.t.} & \\ & \displaystyle\sum_{j_1=1}^{J_1} \cdots \sum_{j_Q=1}^{J_Q} \lambda_{j_1 \cdots j_Q} \hat{a}^{(p)}_{j_1 \cdots j_Q} \leq \theta x^{(p)}_k, & p = 1, \ldots, m \\ & \displaystyle\sum_{j_1=1}^{J_1} \cdots \sum_{j_Q=1}^{J_Q} \lambda_{j_1 \cdots j_Q} \hat{f}^{(l)}(\hat{\boldsymbol{a}}_{j_1 \cdots j_Q}) \geq y^{(l)}_k, & l = 1, \ldots, s \\ & \displaystyle\sum_{j_1=1}^{J_1} \cdots \sum_{j_Q=1}^{J_Q} \lambda_{j_1 \cdots j_Q} = 1, & \\ & \lambda_{j_1 \cdots j_Q} \in \{0, 1\}, & j_q = 1, \ldots, J_q, \\ & & q = 1, \ldots, Q \end{array} \tag{8}$$

In the case of the Russell measures, the Russell measure of output efficiency should be calculated as follows:

$$\begin{array}{rll} \hat{R}^{\text{EATBoost}}_{\text{out}}(\boldsymbol{x}_k, \boldsymbol{y}_k) = & \max\ \displaystyle\frac{1}{s}\sum_{l=1}^{s} \phi_l & \\ & \text{s.t.} \displaystyle\sum_{j_1=1}^{J_1} \cdots \sum_{j_Q=1}^{J_Q} \lambda_{j_1 \cdots j_Q} \hat{a}^{(p)}_{j_1 \cdots j_Q} \leq x^{(p)}_k, & p = 1, \ldots, m \end{array}$$

$$
\begin{aligned}
&\sum_{j_1=1}^{J_1}\cdots\sum_{j_Q=1}^{J_Q}\lambda_{j_1\cdots j_Q}\hat{f}^{(l)}(\hat{\boldsymbol{a}}_{j_1\cdots j_Q}) \geq \phi_l y_k^{(l)}, \quad l=1,\ldots,s\\
&\sum_{j_1=1}^{J_1}\cdots\sum_{j_Q=1}^{J_Q}\lambda_{j_1\cdots j_Q} = 1,\\
&\lambda_{j_1\cdots j_Q}\in\{0,1\}, && j_q=1,\ldots,J_q,\\
& && q=1,\ldots,Q\\
&\phi_l \geq 1, && l=1,\ldots,s
\end{aligned}
\tag{9}
$$

while the Russell measure of input efficiency would be as follows:

$$
\begin{aligned}
\hat{R}_{\text{in}}^{\text{EATBoost}}(\boldsymbol{x}_k,\boldsymbol{y}_k) = \ & \min\ \frac{1}{m}\sum_{p=1}^{m}\theta_p\\
& \text{s.t.} \sum_{j_1=1}^{J_1}\cdots\sum_{j_Q=1}^{J_Q}\lambda_{j_1\cdots j_Q}\hat{a}_{j_1\cdots j_Q}^{(p)} \leq \theta_p x_k^{(p)}, \quad p=1,\ldots,m\\
& \sum_{j_1=1}^{J_1}\cdots\sum_{j_Q=1}^{J_Q}\lambda_{j_1\cdots j_Q}\hat{f}^{(l)}(\hat{\boldsymbol{a}}_{j_1\cdots j_Q}) \geq y_k^{(l)}, \quad l=1,\ldots,s\\
& \sum_{j_1=1}^{J_1}\cdots\sum_{j_Q=1}^{J_Q}\lambda_{j_1\cdots j_Q} = 1,\\
& \lambda_{j_1\cdots j_Q}\in\{0,1\}, \qquad j_q=1,\ldots,J_q,\\
& \qquad\qquad\qquad\qquad q=1,\ldots,Q\\
& \theta_p \leq 1, \qquad\qquad\quad p=1,\ldots,m
\end{aligned}
\tag{10}
$$

Regarding the directional distance function (DDF), it would be calculated by the following optimization problem, where $(\boldsymbol{g}_k^-,\boldsymbol{g}_k^+)\in R_+^{m+s}$ is the corresponding directional vector:

$$
\begin{aligned}
\hat{\beta}^{\text{EATBoost}}(\boldsymbol{x}_k,\boldsymbol{y}_k) = \ & \max\beta\\
& \text{s.t.} \sum_{j_1=1}^{J_1}\cdots\sum_{j_Q=1}^{J_Q}\lambda_{j_1\cdots j_Q}\hat{a}_{j_1\cdots j_Q}^{(p)}\\
& \quad \leq x_k^{(p)} - \beta g_k^{+(p)}, \qquad p=1,\ldots,m\\
& \sum_{j_1=1}^{J_1}\cdots\sum_{j_Q=1}^{J_Q}\lambda_{j_1\cdots j_Q}\hat{f}^{(l)}(\hat{\boldsymbol{a}}_{j_1\cdots j_Q})\\
& \quad \geq y_k^{(l)} + \beta g_k^{-(l)}, \qquad l=1,\ldots,s\\
& \sum_{j_1=1}^{J_1}\cdots\sum_{j_Q=1}^{J_Q}\lambda_{j_1\cdots j_Q} = 1,\\
& \lambda_{j_1\cdots j_Q}\in\{0,1\}, \qquad j_q=1,\ldots,J_q,\\
& \qquad\qquad\qquad\qquad q=1,\ldots,Q
\end{aligned}
\tag{11}
$$

Finally, the Enhanced Russell Graph (ERG) measure, once it has been linearized,[41] could be determined through the following optimization model:

$$
\begin{aligned}
\hat{R}^{\text{EATBoost}}_{\text{ERG}}(\boldsymbol{x}_k, \boldsymbol{y}_k) = \quad & \min \ \beta - \frac{1}{m}\sum_{p=1}^{m} \frac{t^{-(p)}}{x_k^{(p)}} \\
& \text{s.t.} \quad \beta + \frac{1}{s}\sum_{l=1}^{s} \frac{t^{+(l)}}{y_k^{(l)}} = 1, \\
& \quad -\beta x_k^{(p)} + \sum_{j_1=1}^{J_1} \cdots \sum_{j_Q=1}^{J_Q} \lambda_{j_1 \cdots j_Q} \hat{a}^{(p)}_{j_1 \cdots j_Q} + t^{-(p)} = 0, \qquad p = 1, \ldots, m \\
& \quad -\beta y_k^{(l)} + \sum_{j_1=1}^{J_1} \cdots \sum_{j_Q=1}^{J_Q} \lambda_{j_1 \cdots j_Q} \hat{f}^{(l)}(\hat{\boldsymbol{a}}_{j_1 \cdots j_Q}) - t^{+(l)} = 0, \quad l = 1, \ldots, s \\
& \sum_{j_1=1}^{J_1} \cdots \sum_{j_Q=1}^{J_Q} \lambda_{j_1 \cdots j_Q} = \beta, \\
& \lambda_{j_1 \cdots j_Q} \in \{0, 1\}, \qquad j_q = 1, \ldots, J_q, \\
& \qquad\qquad\qquad\qquad q = 1, \ldots, Q \\
& \beta \geq 0, \\
& t^{-(p)} \geq 0, \qquad p = 1, \ldots, m \\
& t^{+(l)} \geq 0, \qquad l = 1, \ldots, s
\end{aligned}
\tag{12}
$$

3.2. *A heuristic approach*

As shown in the previous section, we have already indicated how to calculate a list of well-known technical efficiency measures by resorting to an estimation of the production possibility set, based on the EATBoosting algorithm. However, from a computational point of view, the corresponding mixed-integer linear programs will have thousands of variables. For example, if we consider $J_q = 6$, $\forall\, q = 1, \ldots, Q$, and $Q = 10$, the number of decision variables λ will be over 60 thousand million, as many as the final supports that are generated. This is extremely complex at a computational level due to the program's computer memory requirements. Therefore, in this subsection, we propose a heuristic approximation to this problem, where the final supports are replaced by the input profiles that were observed in the data sample of DMUs. Accordingly, let us next show how to implement the heuristic approach for the case of the output-oriented radial model, since the method to be applied on the remaining technical efficiency measures is very similar. In this regard, a heuristic approximation of the optimal value of model (7) could be determined by the following optimization program:

$$\begin{array}{rll}\hat{\phi}^{\text{EATBoost}-\text{HEU}}(\boldsymbol{x}_k, \boldsymbol{y}_k) = & \max & \phi \\ & \text{s.t.} & \\ & \sum_{i=1}^{n} \lambda_i x_i^{(p)} \leq x_k^{(p)}, & p = 1, \ldots, m \\ & \sum_{i=1}^{n} \lambda_i \hat{f}^{(l)}(\boldsymbol{x}_i) \geq \phi y_k^{(l)}, & l = 1, \ldots, s \\ & \sum_{i=1}^{n} \lambda_i = 1, & \\ & \lambda_i \in \{0, 1\}, & i = 1, \ldots, n \end{array} \tag{13}$$

In model (7), the virtual units $\{(\hat{\boldsymbol{a}}_{j_1 \cdots j_Q}, \hat{\boldsymbol{f}}(\hat{\boldsymbol{a}}_{j_1 \cdots j_Q}))\}$, $\forall\, j_q = 1, \ldots, J_q$, $\forall\, q = 1, \ldots, Q$, have been substituted by $\{(\boldsymbol{x}_i, \hat{\boldsymbol{f}}(\boldsymbol{x}_i))\}$, $\forall\, i = 1, \ldots, n$, leading to the constraints of model (13). Notice that, regardless of the values of the hyperparameters J_q and Q, the number of decision variables λ will be always n, i.e., the sample size. This simplification of the original program (7) drastically reduces the complexity of the model and has clear implications on computational time and memory requirements.

Hereinafter, we will refer to (7) as the exact method to determine technical efficiency, whereas (13) will be the heuristic method. Due to the computational difficulties for solving any of the programs linked to the exact methods, both approaches will be compared (exact and heuristic) through two empirical applications in Sec. 5.

4. Computational Experience

In this section, we perform numerous simulations to compare the quality of the heuristic output-oriented radial measure based on the EATBoosting technique and the FDH method. This comparison cannot be extended to the exact approach due to time and computer memory limitations. Additionally, we check the output-oriented radial measure because we know its theoretical value for the data generating process considered in this section, something that does not happen for the remaining measures. We primarily propose a multi-input ($m = 2$) multi-output ($s = 2$) framework, in line with the concepts suggested by Perelman and Santin.[45] Following that simulation context, inputs are generated following a uniform distribution $U[5, 50]$ while the outputs are generated through the following formula:

$$\begin{aligned} -\ln(y^{(1)}) = {} & -1 + 0.5\left(\ln\frac{y^{(2)}}{y^{(1)}}\right) + 0.25\left(\ln\frac{y^{(2)}}{y^{(1)}}\right)^2 - 1.5(\ln x^{(1)}) - 0.6(\ln x^{(2)}) \\ & + 0.2(\ln x^{(1)})^2 + 0.05(\ln x^{(2)})^2 - 0.1(\ln x^{(1)})(\ln x^{(2)}) + 0.05(\ln x^{(1)}) \\ & \times \left(\ln\frac{y^{(2)}}{y^{(1)}}\right) - 0.05(\ln x^{(2)})\left(\ln\frac{y^{(2)}}{y^{(1)}}\right) \end{aligned} \tag{14}$$

A half-normal distribution is also considered for generating the inefficiency term and 0%, 10% and 25% of simulated DMUs are located on the true frontier. Moreover, 100 trials were run ($t = 1, \ldots, 100$) for each combination of sample size and percentage of units on the true frontier. To select the best hyperparameters of the algorithm, in each trial we divide the original simulated dataset into a training and a test sample in a 70%–30% proportion. In our computations, we consider the same size for all trees, i.e., $J_q = J, \forall\, q$. Additionally, we select the best hyperparameters from the following sets: $Q = \{6, 7, 8, 9, 10\}$, $J = \{6, 7, 8, 9, 10\}$ and $v = \{0.3, 0.4, 0.5\}$. Mean squared error (MSE) and bias are used to assess the performance of each approach.

Table 1 sets out the results obtained from the two approaches being assessed (FDH and heuristic EATBoosting). The first two columns indicate the sample size and the percentage of DMUs on the true frontier, respectively. The third and fourth columns indicate the MSE associated with the FDH and heuristic EATBoosting and its standard deviation, while the following two columns show the bias for the methods employed. Moreover, we have calculated (and reported in brackets) the relative difference between FDH and heuristic EATBoosting with respect to MSE and bias to facilitate the comparison. These percentages correspond to the reduction of the MSE and bias when the new technique is applied as opposed to FDH. The improvements in MSE ranged from 7% to 74%, while the bias improvement ranged from 6% to 48%, depending on the sample size and the number of DMUs on the true frontier. In general, a smaller number of DMUs renders greater improvement.

In addition, we propose in this computational section a second multi-output multi-input scenario with three inputs and three outputs ($m = 3$ and $s = 3$) following the ideas of Gstach,[46] later enhanced by Schaefer and Clermont.[47] In this case, the inputs are obtained following a uniform distribution $U[5, 15]$ and the outputs are generated from a Cobb-Douglas function. Additionally, we used the same configuration as before with respect to trials, percentage of units on the real frontier and sets of hyperparameters.

Following the same structure as Table 1, Table 2 reports the results for the two techniques evaluated (FDH and heuristic EATBoosting). As in the 2-input/2-output settings, EATBoosting outperforms FDH in all 3-input/3-output scenarios. In this case, the improvement in MSE ranged from 22% to 73%, while the bias improvement ranged from 13% to 83%. Furthermore, while in FDH the standard deviation of the MSE increases when the number of DMUs increases, reaching its highest value with 500 DMUs (0.904), in EATBoosting it only ranges from 0.101 to 0.289. In fact, it seems to decrease when the number of DMUs increases. The results obtained in both tables demonstrate how the new approach reflects a significant improvement over FDH, obtaining similar improvement percentages both in MSE and bias in the two scenarios tested.

The experiments were carried out on a Personal Computer (PC) with a 3.2 GHz 8-core Ryzen 7 processor, 16 GB RAM and with a Windows 10 Pro operating system. The algorithm was executed in C code, using CPLEX 20.1 (default options) to solve the optimization problems. Regarding the comparison of the two techniques, FDH and EATBoosting, with respect to the execution time spent, the last two columns of Tables 1 and 2 show the average time (in seconds) and the standard deviation

Table 1. Simulation results (two inputs and two outputs).

Num. DMUs	Frontier	MSE		Bias		Time (s)	
		FDH	EATBoosting	FDH	EATBoosting	FDH	EATBoosting
50	25	2.482 ± 6.173	0.698 ± 1.261 (72%)	0.871 ± 0.903	0.456 ± 0.370 (48%)	0.289 ± 0.015	13.784 ± 0.505
50	10	4.131 ± 10.781	1.087 ± 2.052 (74%)	1.215 ± 1.404	0.630 ± 0.601 (48%)	0.290 ± 0.039	13.857 ± 0.669
50	0	2.142 ± 6.716	1.919 ± 6.346 (10%)	0.284 ± 3.489	0.248 ± 3.383 (13%)	0.292 ± 0.007	14.071 ± 0.317
100	25	1.534 ± 3.251	0.896 ± 1.584 (42%)	0.666 ± 0.663	0.489 ± 0.323 (27%)	0.598 ± 0.059	28.773 ± 1.057
100	10	4.324 ± 21.668	1.335 ± 5.475 (69%)	0.635 ± 2.306	0.346 ± 1.045 (46%)	0.612 ± 0.022	29.628 ± 1.142
100	0	2.260 ± 5.871	1.887 ± 5.206 (17%)	0.411 ± 2.609	0.346 ± 2.450 (16%)	0.636 ± 0.042	31.159 ± 1.397
200	25	0.925 ± 1.139	0.400 ± 0.980 (57%)	0.540 ± 0.258	0.383 ± 0.203 (29%)	1.380 ± 0.110	65.175 ± 2.366
200	10	2.075 ± 4.949	1.599 ± 3.230 (23%)	0.872 ± 0.863	0.649 ± 0.511 (26%)	1.521 ± 0.105	73.571 ± 3.496
200	0	2.697 ± 8.487	1.850 ± 7.572 (31%)	0.567 ± 2.292	0.459 ± 2.113 (19%)	1.552 ± 0.084	74.362 ± 2.537
300	25	2.246 ± 3.919	1.004 ± 1.991 (55%)	0.752 ± 0.474	0.521 ± 0.449 (31%)	1.589 ± 0.183	75.816 ± 1.628
300	10	3.186 ± 6.629	2.639 ± 4.600 (17%)	1.023 ± 1.025	0.795 ± 0.603 (22%)	1.619 ± 0.093	75.888 ± 1.859
300	0	5.110 ± 12.966	3.947 ± 11.290 (23%)	1.200 ± 2.113	0.952 ± 1.937 (21%)	1.636 ± 0.179	79.589 ± 2.384
400	25	2.203 ± 3.091	0.916 ± 1.656 (58%)	0.742 ± 0.440	0.494 ± 0.395 (33%)	1.964 ± 0.032	94.481 ± 1.189
400	10	1.568 ± 2.402	1.375 ± 2.236 (12%)	0.733 ± 0.585	0.645 ± 0.391 (12%)	1.971 ± 0.039	94.835 ± 0.766
400	0	4.880 ± 9.818	4.552 ± 9.334 (7%)	0.410 ± 5.682	0.386 ± 5.547 (6%)	1.992 ± 0.038	95.781 ± 1.021
500	25	1.910 ± 2.921	0.614 ± 1.101 (68%)	0.409 ± 0.827	0.720 ± 0.540 (43%)	2.468 ± 0.045	120.989 ± 1.377
500	10	1.391 ± 1.532	1.060 ± 1.551 (24%)	0.648 ± 1.125	0.731 ± 0.713 (11%)	2.485 ± 0.043	121.621 ± 1.390
500	0	1.380 ± 3.870	0.946 ± 3.094 (31%)	0.408 ± 4.321	0.729 ± 4.012 (44%)	2.549 ± 0.093	123.988 ± 1.959

Table 2. Simulation results (three inputs and three outputs).

Num. DMUs	Frontier	MSE		Bias		Time (s)	
		FDH	EATBoosting	FDH	EATBoosting	FDH	EATBoosting
50	20	0.486 ± 0.231	0.174 ± 0.263 (64%)	0.376 ± 0.242	0.188 ± 0.105 (50%)	0.324 ± 0.013	14.2797 ± 0.514
50	10	0.534 ± 0.267	0.253 ± 0.249 (53%)	0.392 ± 0.209	0.287 ± 0.192 (27%)	0.331 ± 0.020	15.552 ± 0.963
50	0	0.551 ± 0.269	0.159 ± 0.247 (71%)	0.403 ± 0.219	0.180 ± 0.197 (55%)	0.349 ± 0.021	16.874 ± 0.549
100	20	0.743 ± 0.488	0.277 ± 0.289 (63%)	0.676 ± 0.281	0.115 ± 0.246 (83%)	0.741 ± 0.053	35.826 ± 2.643
100	10	0.589 ± 0.424	0.266 ± 0.288 (55%)	0.557 ± 0.286	0.112 ± 0.282 (80%)	0.750 ± 0.063	36.384 ± 2.153
100	0	0.863 ± 0.576	0.237 ± 0.277 (73%)	0.236 ± 0.471	0.122 ± 0.212 (48%)	0.769 ± 0.059	37.098 ± 2.372
200	20	0.534 ± 0.241	0.174 ± 0.169 (67%)	0.177 ± 0.445	0.141 ± 0.399 (20%)	1.278 ± 0.097	61.992 ± 3.947
200	10	0.578 ± 0.780	0.212 ± 0.177 (63%)	0.188 ± 0.406	0.161 ± 0.147 (14%)	1.384 ± 0.058	63.232 ± 2.021
200	0	0.717 ± 0.460	0.289 ± 0.134 (60%)	0.347 ± 0.648	0.134 ± 0.236 (61%)	1.396 ± 0.129	66.046 ± 4.140
300	20	0.742 ± 0.243	0.291 ± 0.182 (61%)	0.321 ± 0.459	0.213 ± 0.354 (34%)	2.297 ± 0.165	110.295 ± 3.665
300	10	0.746 ± 0.239	0.302 ± 0.128 (59%)	0.321 ± 0.497	0.219 ± 0.350 (32%)	2.437 ± 0.308	115.754 ± 3.157
300	0	1.019 ± 0.730	0.412 ± 0.105 (60%)	0.472 ± 0.416	0.350 ± 0.118 (26%)	2.499 ± 0.387	118.421 ± 3.476
400	20	0.879 ± 0.147	0.419 ± 0.101 (52%)	0.280 ± 0.534	0.244 ± 0.372(13%)	3.126 ± 0.473	127.996 ± 4.048
400	10	0.841 ± 0.192	0.389 ± 0.118 (54%)	0.271 ± 0.674	0.236 ± 0.504(13%)	3.400 ± 0.388	128.377 ± 4.505
400	0	1.235 ± 0.676	0.366 ± 0.117 (70%)	0.414 ± 0.409	0.150 ± 0.109 (64%)	3.629 ± 0.476	151.215 ± 4.745
500	20	0.890 ± 0.348	0.463 ± 0.189 (22%)	0.303 ± 0.117	0.237 ± 0.089 (22%)	3.711 ± 0.897	164.452 ± 5.153
500	10	0.989 ± 0.904	0.514 ± 0.181 (23%)	0.343 ± 0.368	0.265 ± 0.147 (23%)	3.869 ± 0.859	167.064 ± 5.161
500	0	1.282 ± 0.862	0.419 ± 0.126 (65%)	0.457 ± 0.422	0.159 ± 0.222 (65%)	3.977 ± 0.861	174.387 ± 5.614

associated with each method. The results show a clear difference between the computation time associated with FDH and EATBoosting. The average time spent by EATBoosting is between 44 and 48 times longer than the average time associated with applying FDH. Although it may be considered a weakness of the new technique, it is worth highlighting that, in the more complex scenario, with three inputs, three outputs and 500 observations, the average time associated with the new approach to obtain the efficiency scores was less than 3 min (with a standard deviation of approximately 5 s) using a desktop PC.

5. Empirical Applications

In this section, we will illustrate the new approach by applying it on two empirical databases. The first subsection is devoted to comparing the exact and heuristic approaches through a real scenario. In the second subsection, both methods are again tested in another real situation, but in this case putting the spotlight on the curse of dimensionality problem, which is related to the capacity of the models to discriminate between efficient and inefficient DMUs.

5.1. *Empirical comparison of the exact and heuristic EATBoosting methods*

As previously explained, a simulation experience cannot be used to compare the differences between the exact EATBoosting efficiency measures and the heuristic approach due to the computational complexity. However, the comparison is possible through a particular numerical example. For this reason, in this subsection, we utilize a dataset to prove their performances in a real scenario and compare them with respect to the FDH measures. In particular, we select the dataset proposed by Juo *et al.*[48] for year 2010, where data from 31 banks were obtained.[b] We use financial funds, labor and physical capital as inputs; while financial investment and loans are used as outputs. In Table 3, a summary of the main descriptive statistics of the data are presented.

Table 3. Summary statistics of Taiwanese banks for year 2010.

	Financial funds	Labor	Physical capital	Financial investment	Loans
Min	25,019	202	505	1,681	66,947
1st Qu.	185,698	1,607	3,032	15,634	146,481
Median	428,995	3,146	8,721	157,870	328,574
Mean	795,536	3,826	13,393	196,808	609,489
3rd Qu.	1,301,406	6,149	19,956	317,859	942,599
Max.	3,171,493	9,538	76,576	904,580	2,091,100
St. dev.	768,008	2,729	15,185	215,062	582,653

[b] We are grateful to these authors for sharing the data with us.

Table 4. Efficiency scores for Taiwanese banks dataset (first part).

DMU	Radial output			Radial input			Russell output		
	FDH	Exa.	Heu.	FDH	Exa.	Heu.	FDH	Exa.	Heu.
Export-Import Bank	1.000	1.040	1.040	1.000	1.000	1.000	1.000	1.257	1.257
Bank of Taiwan	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.257
Taipei Fubon Bank	1.000	1.692	1.692	1.000	1.000	0.847	1.000	1.733	1.733
Bank of Kaohsiung	1.000	1.067	1.063	1.000	1.000	1.000	1.000	3.991	3.779
Land Bank	1.000	1.096	1.096	1.000	1.000	1.000	1.000	2.349	2.335
Cooperative Bank	1.000	1.040	1.040	1.000	0.832	0.832	1.000	1.334	1.327
First Bank	1.000	1.220	1.220	1.000	0.711	0.646	1.000	1.286	1.286
Hua Nan Bank	1.000	1.223	1.223	1.000	0.703	0.649	1.000	1.485	1.485
Chang Hwa Bank	1.000	1.450	1.450	1.000	0.817	0.817	1.000	2.060	2.060
Mega Bank	1.000	1.153	1.153	1.000	1.000	0.652	1.000	1.540	1.540
Cathay United Bank	1.000	1.724	1.724	1.000	0.826	0.767	1.000	2.975	2.975
The Shanghai Bank	1.000	1.044	1.000	1.000	1.000	1.000	1.000	1.164	2.975
Union Bank	1.000	1.082	1.081	1.000	1.000	1.000	1.000	4.715	4.617
Far Eastern Bank	1.000	1.004	1.004	1.000	1.000	1.000	1.000	1.314	1.314
E. Sun Bank	1.000	1.003	1.003	1.000	1.000	1.000	1.000	1.084	1.084
Cosmos Bank	1.266	1.313	1.304	0.228	0.228	0.228	34.555	35.081	34.555
Taishin Bank	1.000	1.103	1.103	1.000	1.000	1.000	1.000	1.375	1.375
Ta Chong Bank	1.000	1.036	1.000	1.000	1.000	1.000	1.000	1.122	1.375
Jih Sun Bank	1.000	1.026	1.026	1.000	1.000	1.000	1.000	2.681	2.681
Entie Bank	1.000	1.047	1.045	1.000	1.000	0.990	1.000	2.360	2.317
China Trust Bank	1.000	1.355	1.355	1.000	0.776	0.776	1.000	1.487	1.487
Sunny Bank	1.000	1.019	1.000	1.000	0.888	1.000	1.000	10.970	1.487
Bank of Panhsin	1.000	1.087	1.087	1.000	1.000	1.000	1.000	26.014	26.014
Taiwan Business Bank	1.000	1.644	1.644	1.000	1.000	1.000	1.000	2.926	2.926
Taichung Bank	1.000	1.013	1.013	1.000	1.000	1.000	1.000	4.490	4.490
China Development	1.000	1.011	1.011	1.000	1.000	1.000	1.000	1.138	1.138
Hwatai Bank	1.000	1.037	1.029	1.000	0.731	0.731	1.000	19.060	19.056
Cota Bank	1.000	1.045	1.045	1.000	1.000	0.711	1.000	6.225	6.225
Industrial Bank of Taiwan	1.000	1.304	1.304	1.000	1.000	0.716	1.000	1.752	1.752
Bank SinoPac	1.000	1.003	1.003	1.000	1.000	1.000	1.000	1.131	1.131
Shin Kong Bank	1.000	1.186	1.186	1.000	0.803	0.803	1.000	8.998	8.998

We executed the EATBoosting algorithm using $Q = 8$, $J_q = 8 \forall\, q = 1, \dots, 8$ and $v = 0.6$ as the hyperparameters.[c] In Table 4, the results for the output-oriented radial model, input-oriented radial model and the Russell measure of output efficiency for each bank are shown, while the Russell measure of input efficiency, the directional distance function and the Enhanced Russell Graph scores are presented in Table 5. For every measure except for the Enhanced Russell Graph, under the FDH technique, all the DMUs are technically efficient, except one (Cosmos Bank). Additionally, both the exact and the heuristic methods present identical score values for many measures and banks. Although in the case of the Enhanced Russell Graph measure, the exact and the heuristic approaches only present the same score for the Export-Import Bank. Moreover, it seems that the scores obtained from the

[c]We used other hyperparameter settings and obtained similar results.

Table 5. Efficiency scores for Taiwanese banks dataset (second part).

DMU	Russell input			DDF			ERG		
	FDH	Exa.	Heu.	FDH	Exa.	Heu.	FDH	Exa.	Heu.
Export-Import Bank	1.000	1.000	1.000	0.000	0.000	0.000	1.000	0.796	0.796
Bank of Taiwan	1.000	1.000	1.000	0.000	0.000	0.000	1.000	0.126	1.000
Taipei Fubon Bank	1.000	1.000	0.307	0.000	0.000	0.153	1.000	0.177	0.524
Bank of Kaohsiung	1.000	1.000	0.482	0.000	0.000	0.000	0.217	0.085	0.164
Land Bank	1.000	1.000	0.353	0.000	0.000	0.000	1.000	0.132	0.396
Cooperative Bank	1.000	0.718	0.290	0.000	0.040	0.040	1.000	0.207	0.522
First Bank	1.000	0.659	0.232	0.000	0.220	0.220	1.000	0.184	0.518
Hua Nan Bank	1.000	0.641	0.233	0.000	0.223	0.223	0.792	0.160	0.436
Chang Hwa Bank	1.000	0.731	0.290	0.000	0.183	0.183	0.636	0.144	0.355
Mega Bank	1.000	1.000	0.243	0.000	0.000	0.153	1.000	0.158	0.579
Cathay United Bank	1.000	0.718	0.274	0.000	0.174	0.233	0.407	0.093	0.239
The Shanghai Bank	1.000	1.000	0.274	0.000	0.000	0.233	0.743	0.197	0.507
Union Bank	1.000	1.000	0.377	0.000	0.000	0.000	0.066	0.037	0.057
Far Eastern Bank	1.000	1.000	0.421	0.000	0.000	0.000	1.000	0.319	0.761
E. Sun Bank	1.000	1.000	0.681	0.000	0.000	0.000	0.783	0.180	0.475
Cosmos Bank	0.143	0.143	0.143	0.168	0.296	0.296	0.012	0.008	0.012
Taishin Bank	1.000	1.000	0.354	0.000	0.000	0.000	0.468	0.159	0.310
Ta Chong Bank	1.000	1.000	0.354	0.000	0.000	0.000	0.584	0.209	0.423
Jih Sun Bank	1.000	1.000	0.473	0.000	0.000	0.000	0.187	0.104	0.156
Entie Bank	1.000	1.000	0.444	0.000	0.000	0.010	0.265	0.092	0.205
China Trust Bank	1.000	0.579	0.271	0.000	0.224	0.224	1.000	0.185	0.392
Sunny Bank	1.000	0.548	0.271	0.000	0.001	0.224	0.041	0.020	0.031
Bank of Panhsin	1.000	1.000	0.441	0.000	0.000	0.000	0.019	0.011	0.016
Taiwan Business Bank	1.000	1.000	0.359	0.000	0.000	0.000	0.721	0.122	0.328
Taichung Bank	1.000	1.000	0.422	0.000	0.000	0.000	0.123	0.043	0.094
China Development	1.000	1.000	0.785	0.000	0.000	0.000	1.000	0.662	0.876
Hwatai Bank	1.000	0.708	0.423	0.000	0.029	0.029	0.041	0.029	0.037
Cota Bank	1.000	1.000	0.451	0.000	0.000	0.045	0.058	0.038	0.051
Industrial Bank of Taiwan	1.000	1.000	0.504	0.000	0.000	0.284	1.000	0.261	0.532
Bank SinoPac	1.000	1.000	0.367	0.000	0.000	0.000	0.773	0.173	0.515
Shin Kong Bank	1.000	0.653	0.313	0.000	0.186	0.186	0.094	0.035	0.071

EATBoosting algorithm help to discriminate more between efficient and inefficient banks. For example, in the case of the Enhanced Russell Graph measure, under the FDH technique, only 11 banks are considered technically efficient. In contrast, using the heuristic approach, only one unit is considered as technically efficient (the bank of Taiwan), while using the exact approach, none of them is identified as efficient.

Furthermore, we apply the Li test[49] to check if the score vectors for every measure show statistically significant differences. Following the Li test principles, given the densities of two random variables U^A, $U^Z \in R$ for which the random samples, $\{u^{A,k} : k = 1, \ldots, n_A\}$ and $\{u^{Z,k} : k = 1, \ldots, n_Z\}$, representing the two subgroups A and Z in a population are available, if f_l denotes the density of the distribution of a random variable $U^l (l = A, Z)$, then our null hypothesis is that $H_0 : f_A(u^A) = f_Z(u^Z)$.[49] The results of the Li-test p-values are shown in Table 6, whereas the distribution functions are represented in Fig. 3.

Table 6. Li test results (p-values).

	Radial output	Radial input	Russell output	Russell input	DDF	ERG
FDH versus Heu.	0.018	0.000	0.000	0.089	0.017	0.000
FDH versus Exact	0.001	0.000	0.000	0.000	0.017	0.000
Exact versus Heu.	0.960	0.006	0.981	0.000	0.099	0.002

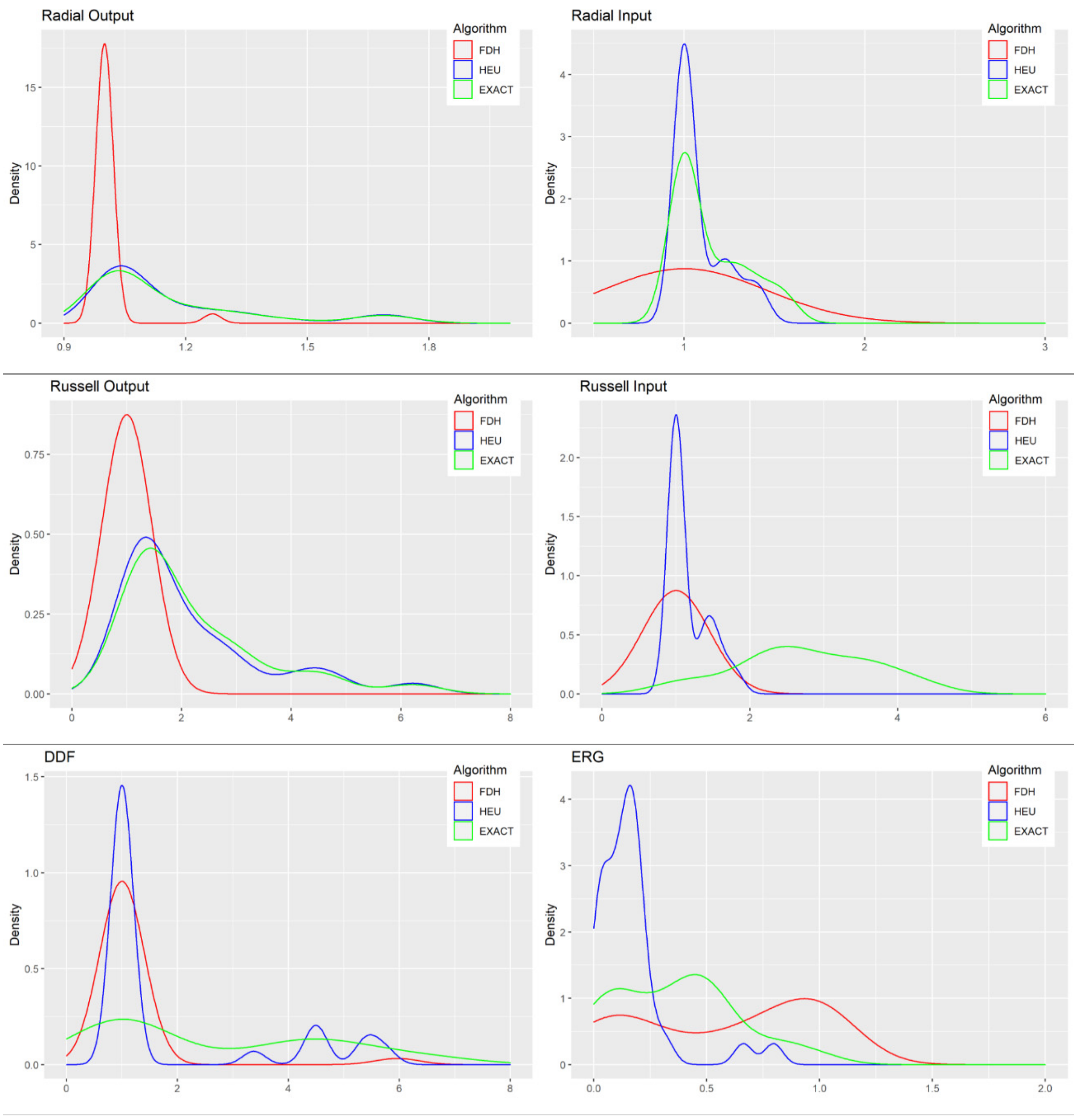

Fig. 3. Efficiency scores distributions.

Both the heuristic and exact EATBoosting methods generate efficiency scores which are clearly distinguishable from the FDH method (p-value < 0.05). These differences are evident in the graphs, where the distributions of the performance scores show quite different patterns. An extreme case, in this sense, is the graph

Table 7. Fortune's best US cities data with its corresponding output-oriented radial efficiency scores.

DMU	$x1$	$x2$	$x3$	$x4$	$x5$	$x6$	$y1$	$y2$	$y3$	$y4$	$y5$	$y6$	FDH	Exact	Heu.
Seattle	586,000	581	1.45	4.5	21	542.3	46,928	0.297	4.49	7	117	22	1.000	1.057	1.044
Denver	475,000	558	0.97	4	14	595.6	42,879	0.291	2.79	5	60	71	1.000	1.093	1.053
Philadelphia	201,000	600	1.5	4.75	21	693.6	43,576	0.227	3.64	25	216	166	1.000	1.000	1.000
Minneapolis	299,000	609	1.49	4	24	496.5	45,673	0.27	2.67	6	131	125	1.000	1.001	1.001
Ral-Durham	318,000	613	0.99	4.5	18	634.7	40,990	0.319	4.94	7	33	47	1.000	1.025	1.025
St. Louis	265,000	558	0.89	3	18	263	39,079	0.206	3.4	10	104	62	1.000	1.021	1.021
Cincinnati	467,000	580	1.25	3.75	20	551.5	38,455	0.199	2.8	4	71	94	1.000	1.101	1.101
Washington	583,000	625	1.29	3.75	33	714.5	54,291	0.373	3.35	30	148	105	1.000	1.000	1.000
Pittsburgh	347,000	535	0.99	3.75	17	382.1	34,534	0.188	3.66	8	124	112	1.000	1.108	1.108
Dallas-FW	296,000	650	1.5	5	18	825.4	41,984	0.271	1.96	3	98	77	1.000	1.075	1.075
Atlanta	600,000	740	1.19	6.75	20	846.6	43,249	0.263	2.23	9	118	102	1.000	1.053	1.053
Baltimore	575,000	775	0.99	3.99	18	1296.3	43,291	0.233	4.02	8	102	45	1.000	1.053	1.053
Boston	351,000	888	1.09	4.25	34	686.6	46,444	0.325	5.69	25	240	55	1.000	1.000	1.000
Milwaukee	283,000	727	1.53	3.5	26	518.9	41,841	0.214	3.11	6	52	50	1.000	1.110	1.121
Nashville	431,000	695	1.19	4	26	1132.5	40,221	0.215	3.25	4	37	37	1.000	1.292	1.281

corresponding to the output-oriented radial model. An exception is the case of the comparison by means of the Li test of the distribution of scores of the heuristic method with respect to the FDH in the Russell input-oriented measure. In that case, the p-value is low but is above the 5% error threshold (p-value $= 0.089 > 0.05$). Additionally, differences between heuristic and exact methods were only observed for input-oriented models and the Enhanced Russell Graph measure. Nevertheless, we are aware that many more real databases should be analyzed before arriving at any clear conclusion regarding this comparison.

5.2. *Curse of dimensionality*

Finally, we turn our attention to the curse of dimensionality problem. This issue is related to lack of discriminatory power as the dimensionality of the production space increases. This drawback is especially evident in the case of FDH when the ratio of the number of variables (inputs and outputs) to the number of DMUs is low. In this case, a huge number of the DMUs are considered as technically efficient. To show this issue, we resort to Fortunes's US 15 best cities, a dataset used in Ref. 50. In this real scenario, 15 cities with six inputs and six outputs were studied. In Table 7, the values of these variables are shown as well as the output-oriented radial output scores for each city using the standard FDH, the exact EATBoosting method and the corresponding heuristic approach. We executed the EATBoosting algorithm using $Q = 8$, $J_q = 8 \forall\, q = 1, \ldots, 8$ and $v = 0.6$ as the hyperparameters[d].

In this empirical example (see Table 7), both the exact method and the heuristic approach get very similar efficiency scores. Additionally, and regarding the curse of dimensionality problem, the FDH technique labels all DMUs as technically efficient. In contrast, the new approach, based on the EATBoosting algorithm, is able to distinguish between efficient and inefficient cities. According to the new methodology, only 3 out of the 15 cities (Philadelphia, Washington, and Boston) are technically efficient. In this regard, the EATBoosting technique could be seen as a possible remedy to the curse of dimensionality in nonparametric efficiency measurement. Nevertheless, while these results are encouraging, further analysis for the future is needed to arrive at a final conclusion on that question.

6. Conclusion and Future Research

To the best of our knowledge, this paper is the first to show how to theoretically measure technical efficiency using the technology estimated by the adaptation of Gradient Tree Boosting (called the EATBoosting technique); a machine learning methodology that improves on the standard nonparametric FDH approach while still meeting free disposability in inputs and outputs and providing estimates that envelop the data cloud from above. In this context, we introduced mixed-linear programs that allow well-known measures in the literature to be determined, such as the

[d]We used other hyperparameter settings and obtained similar results.

output-oriented and input-oriented radial models, the Russell measure of output efficiency and the Russell measure of input efficiency, the directional distance function and the Enhanced Russell Graph measure. Moreover, we also introduced a heuristic approach to deal with certain extremely computationally complex scenarios linked to EATBoosting, where both computational time and memory requirements were improved compared to the exact alternative. Our computational experience proved that the results associated with EATBoosting clearly surpass FDH when it comes to reducing mean squared error and bias. Also, through two empirical applications, we showed that heuristic models are good estimators for exact ones, especially in the case of output-oriented measures. Furthermore, we also demonstrated through an empirical example that EATBoosting could be understood as a possible remedy for solving the curse of dimensionality problem. Overall, we appreciated that the introduction of EATBoosting measures in the literature breaks new ground for adapting and integrating machine learning techniques to the whole context of efficiency analysis and measurement. However, we would like to highlight a clear limitation associated with the new approach. The EATBoosting algorithm is linked to a very intensive computational procedure. This feature contrasts sharply with the simplicity of the standard Free Disposal Hull technique, which is based on Mixed Integer Linear Programming.

We conclude by suggesting interesting avenues for further research in efficiency measurement. We firstly recommend looking further into how technical efficiency can be measured using the boosting algorithm within a production framework. In this regard, XGboost (eXtreme Gradient boosting)[51] could be considered as an interesting method when dealing with regression problems as well as LS-Boosting.[20] Additionally, further comparisons of the results obtained with the exact and heuristic approaches are necessary to ascertain more similarities and differences in both methods. Regarding the curse of dimensionality, this issue alone could be the subject of a paper due to its relevance in the efficiency measurement field. The new approach should be assessed with respect to other possible solutions pointed out in the literature and more databases. Moreover, an additional possible extension of the new approach could be assuming convexity of the technology and comparing the results of the new adaptation with respect to those obtained by applying Data Envelopment Analysis. Another noteworthy line of research could be the enhancement of the technique regarding the computation time associated with EATBoosting. In this sense, the parallelization of the corresponding algorithm (dealing with different values for the hyperparameters Q, J_q and v) could decrease the computation time of the approach. Finally, a patent research line is to extend the application of the new approach to more real databases in different empirical contexts, thus verifying the effectiveness of the model in practice.

Acknowledgments

The authors would like to thank the grant PID2019-105952GB-I00 funded by Ministerio de Ciencia e Innovación/Agencia Estatal de Investigación/10.13039/

501100011033. This work was also supported by the Generalitat Valenciana under Grant CIACIF/2021/345. Additionally, the authors would like to thank the grant PROMETEO/2021/063 funded by the governments of Spain and the Valencian Community. Moreover, M. Esteve would like to thank the grant FPU17/05365 supported by the Spanish Ministry of Science, Innovation and Universities.

References

1. Q. Xie, X. Chen, L. Li, K. Rao, L. Tao and C. Ma, Image fusion based on kernel estimation and data envelopment analysis, *International Journal of Information Technology & Decision Making* **18**(02) (2019) 487–515.
2. B. Hsiao and L. H. Chen, Performance evaluation for Taiwanese hospitals by multi-activity network data envelopment analysis, *International Journal of Information Technology & Decision Making* **18**(3) (2019) 1009–1043.
3. M. K. Yilmaz, A. O. Kusakci, E. Tatoglu, O. Icten and F. Yetgin, Performance evaluation of real estate investment trusts using a hybridized interval Type-2 Fuzzy AHP-DEA approach: The case of Borsa Istanbul, *International Journal of Information Technology & Decision Making* **18**(06) (2019) 1785–1820.
4. L. Orea and J. L. Zofío, Common methodological choices in nonparametric and parametric analyses of firms' performance, in *The Palgrave Handbook of Economic Performance Analysis* (Palgrave Macmillan, Cham, 2019), pp. 419–484.
5. M. J. Farrell, The measurement of productive efficiency, *Journal of the Royal Statistical Society: Series A (General)* **120**(3) (1957) 253–281.
6. S. N. Afriat, The construction of utility functions from expenditure data, *International Economic Review* **8**(1) (1967) 67–77.
7. A. Charnes, W. W. Cooper and E. Rhodes, Measuring the efficiency of decision-making units, *European Journal of Operational Research* **2**(6) (1978) 429–444.
8. R. D. Banker, A. Charnes and W. W. Cooper, Some models for estimating technical and scale inefficiencies in data envelopment analysis, *Management Science* **30**(9) (1984) 1078–1092.
9. D. Deprins, L. Simar and H. Tulkens, Measuring labor inefficiency in post offices, in *The Performance of Public Enterprises: Concepts and Measurements*, eds. M. Marchand, P. Pestieau and H. Tulkens (North-Holland, Amsterdam, 1984), pp. 243–267.
10. L. Simar and P. W. Wilson, Sensitivity analysis of efficiency scores: How to bootstrap in nonparametric frontier models, *Management Science* **44**(1) (1998) 49–61.
11. L. Simar and P. W. Wilson, A general methodology for bootstrapping in non-parametric frontier models, *Journal of Applied Statistics* **27**(6) (2000) 779–802.
12. L. Simar and P. W. Wilson, Statistical inference in nonparametric frontier models: The state of the art, *Journal of Productivity Analysis* **13**(1) (2000) 49–78.
13. B. Efron, Bootstrap methods: Another look at the jackknife, *Annals of Statistics* **7**(1) (1979) 1–26.
14. R. Färe and C. K. Lovell, Measuring the technical efficiency of production, *Journal of Economic Theory* **19**(1) (1978) 150–162.
15. R. Chambers, Y. Chung and R. Färe, Profit, directional distance functions, and Nerlovian efficiency, *Journal of Optimization Theory and Applications* **98**(2) (1998) 351–364.
16. J. Friedman, T. Hastie and R. Tibshirani, Additive logistic regression: A statistical view of boosting (with discussion and a rejoinder by the authors), *The Annals of Statistics* **28**(2) (2000) 337–407.

17. A. Natekin and A. Knoll, Gradient boosting machines, a tutorial, *Frontiers in Neurorobotics* **7** (2013) 21.
18. L. Breiman, Random forests, *Machine Learning* **45**(1) (2001) 5–32.
19. T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Springer Science & Business Media, 2009).
20. J. H. Friedman, Greedy function approximation: A gradient boosting machine, *Annals of Statistics* **29**(5) (2001) 1189–1232.
21. L. Breiman, J. Friedman, R. Olshen and C. Stone, *Classification and Regression Trees* (Taylor & Francis, 1984).
22. T. Kuosmanen and A. L. Johnson, Data envelopment analysis as nonparametric least-squares regression, *Operations Research* **58**(1) (2010) 149–160.
23. T. Kuosmanen and A. Johnson, Modeling joint production of multiple outputs in StoNED: Directional distance function approach, *European Journal of Operational Research* **262**(2) (2017) 792–801.
24. P. Madden, *Concavity and Optimization in Microeconomics* (Blackwell, 1986).
25. Y. Aragon, A. Daouia and C. Thomas-Agnan, Nonparametric frontier estimation: A conditional quantile-based approach, *Econometric Theory* **21**(2) (2005) 358–389.
26. C. F. Parmeter and J. S. Racine, Smooth constrained frontier analysis, in *Recent Advances and Future Directions in Causality, Prediction, and Specification Analysis* (Springer, New York, NY, 2013), pp. 463–488.
27. M. Esteve, J. Aparicio, A. Rabasa and J. J. Rodriguez-Sala, Efficiency analysis trees: A new methodology for estimating production frontiers through decision trees, *Expert Systems with Applications* **162** (2020) 113783.
28. M. G. Tsionas, Efficiency estimation using probabilistic regression trees with an application to Chilean manufacturing industries, *International Journal of Production Economics* **249** (2022) 108492.
29. D. Valero-Carreras, J. Aparicio and N. M. Guerrero, Support vector frontiers: A new approach for estimating production functions through support vector machines, *Omega* **104** (2021) 102490.
30. N. M. Guerrero, J. Aparicio and D. Valero-Carreras, Combining data envelopment analysis and machine learning, *Mathematics* **10**(6) (2022) 909.
31. O. B. Olesen and J. Ruggiero, An improved Afriat–Diewert–Parkan nonparametric production function estimator, *European Journal of Operational Research* **264**(3) (2018) 1172–1188.
32. O. B. Olesen and J. Ruggiero, The hinging hyperplanes: An alternative nonparametric representation of a production function, *European Journal of Operational Research* **296**(1) (2022) 254–266.
33. K. Kerstens and I. Van de Woestyne, Cost functions are nonconvex in the outputs when the technology is nonconvex: Convexification is not harmless, *Annals of Operations Research* **305**(1) (2021) 81–106.
34. F. Ang, K. Kerstens and J. Sadeghi, Energy productivity and greenhouse gas emission intensity in Dutch dairy farms: A Hicks–Moorsteen by-production approach under non-convexity and convexity with equivalence results, *Journal of Agricultural Economics* (2022) 1–18.
35. K. Kerstens, J. Sadeghi, M. Toloo and I. Van de Woestyne, Procedures for ranking technical and cost efficient units: With a focus on nonconvexity, *European Journal of Operational Research* **300**(1) (2022) 269–281.
36. G. Kou, Ö. Olgu Akdeniz, H. Dinçer and S. Yüksel, Fintech investments in European banks: A hybrid IT2 fuzzy multidimensional decision-making approach, *Financial Innovation* **7**(1) (2021) 1–28.

37. G. Kou, S. Yüksel and H. Dinçer, Inventive problem-solving map of innovative carbon emission strategies for solar energy-based transportation investment projects, *Applied Energy* **311** (2022) 118680.
38. X. Chao, Y. Dong, G. Kou and Y. Peng, How to determine the consensus threshold in group decision making: A method based on efficiency benchmark using benefit and cost insight, *Annals of Operations Research* **316**(1) (2022) 143–177.
39. Y. Li, G. Kou, G. Li and Y. Peng, Consensus reaching process in large-scale group decision making based on bounded confidence and social network, *European Journal of Operational Research* **303**(2) (2022) 790–802.
40. A. Panwar, M. Olfati, M. Pant and V. Snasel, A Review on the 40 Years of Existence of Data Envelopment Analysis Models: Historic Development and Current Trends, *Archives of Computational Methods in Engineering: State of the Art Reviews* **29**(7) (2022) 5397–5426.
41. J. T. Pastor, J. L. Ruiz and I. Sirvent, An enhanced DEA Russell graph efficiency measure, *European Journal of Operational Research* **115**(3) (1999) 596–607.
42. J. T. Pastor, C. A. Lovell and J. Aparicio, Families of linear efficiency programs based on Debreu's loss function, *Journal of Productivity Analysis* **38**(2) (2012) 109–120.
43. M. Esteve, J. J. Rodríguez-Sala, J. J. López-Espín and J. Aparicio, Heuristic and Backtracking Algorithms for Improving the Performance of Efficiency Analysis Trees, *IEEE Access* **9** (2021) 17421–17428.
44. M. Kuhn and K. Johnson, *Applied Predictive Modeling* (Springer, New York, 2013).
45. S. Perelman and D. Santín, How to generate regularly behaved production data? A Monte Carlo experimentation on DEA scale efficiency measurement, *European Journal of Operational Research* **199**(1) (2009) 303–310.
46. D. Gstach, Bounded vs. unbounded noise in efficiency estimation: Performance of alternative estimators, in *Data Envelopment Analysis in the Service Sector* (Deutscher Universitätsverlag, Wiesbaden, 1999), pp. 103–119.
47. J. Schaefer and M. Clermont, Stochastic non-smooth envelopment of data for multidimensional output, *Journal of Productivity Analysis* **50**(3) (2018) 139–154.
48. J. C. Juo, T. T. Fu, M. M. Yu and Y. H. Lin, Profit-oriented productivity change, *Omega* **57** (2015) 176–187.
49. L. Simar and V. Zelenyuk, On testing equality of distributions of technical efficiency scores, *Econometric Reviews* **25**(4) (2006) 497–522.
50. V. Charles, J. Aparicio and J. Zhu, The curse of dimensionality of decision-making units: A simple approach to increase the discriminatory power of data envelopment analysis, *European Journal of Operational Research* **279**(3) (2019) 929–940.
51. T. Chen and C. Guestrin, XGBoost, in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (San Francisco, USA, 2016), pp. 785–794.
52. A. Daouia, H. Noh and B. U. Park, Data envelope fitting with constrained polynomial splines. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **78**(1) (2016) 3–30.

Apéndice C

`boostingDEA`: A boosting approach to Data Envelopment Analysis in R

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

boostingDEA: A boosting approach to Data Envelopment Analysis in R

Maria D. Guillen [a], Juan Aparicio [a,b,*], Victor J. España [a]

[a] Center of Operations Research (CIO), Miguel Hernandez University of Elche (UMH), 03202 Elche (Alicante), Spain
[b] Valencian Graduate School and Research Network of Artificial Intelligence (valgrAI), 46022 Valencia, Spain

A R T I C L E I N F O

Keywords:
R
Data Envelopment Analysis
Free Disposal Hull
Boosting

A B S T R A C T

`boostingDEA` is a new package for R that includes functions to estimate production frontiers and make ideal output predictions in the Data Envelopment Analysis (DEA) context. The package implements both standard models from DEA and Free Disposal Hull (FDH) and, for the first time, incorporates boosting techniques. Boosting is a method used in machine learning that attempts to overcome the overfitting issue, typically sustained in standard methods, by training multiple models sequentially to improve the accuracy of the overall system. Moreover, the package includes code for estimating several technical efficiency measures using different models such as the input and output-oriented radial measures, the input and output-oriented Russell measures, the Directional Distance Function (DDF), the Weighted Additive Measure (WAM) and the Slacks-Based Measure (SBM).

Code metadata

Current code version	0.1.0
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-23-00466
Permanent link to Reproducible Capsule	NA
Legal Code License	AGPL (≥ 3)
Code versioning system used	Git
Software code languages, tools, and services used	R (≥ 3.5.0)
Compilation requirements, operating environments & dependencies	R (≥ 3.5.0)
If available Link to developer documentation/manual	https://cran.r-project.org/web/packages/boostingDEA/boostingDEA.pdf https://cran.r-project.org/web/packages/boostingDEA/vignettes/boostingDEA.html
Support email for questions	maria.guilleng@umh.es

1. Motivation and significance

Efficiency is a crucial aspect of any production process, as it determines how effectively resources are used to generate output. In the context of organizations, technical efficiency refers to the ability to produce maximum output from a given set of inputs. Measuring technical efficiency provides insights into the performance of firms and can help identify areas where productivity improvements can be made [1,2]. In this context, both parametric (e.g., Stochastic Frontier Analysis [3]) and non-parametric methodologies (e.g., Data Envelopment Analysis [4]) have been developed. However, non-parametric approaches are more appealing than their parametric counterpart due to their flexibility, the mild conditions required and their natural treatment of multi-input multi-output production contexts [5].

One widely-used non-parametric method for measuring technical efficiency is Data Envelopment Analysis (DEA). DEA [4] determines the relative efficiency of a set of decision-making units (DMUs) by taking into account multiple inputs and outputs and comparing the performance of a given DMU to the other DMUs in the set. To do so, DEA relies on the construction of a technology in the input–output space that satisfies free disposability (i.e., it is always possible to do worse), envelopmentness (i.e., including all observed DMUs), convexity and minimal extrapolation (i.e., the smallest set) [6]. On the other hand, another renowned non-parametric approach is Free Disposal Hull (FDH) [7], which is based upon the construction of

* Corresponding author at: Center of Operations Research (CIO), Miguel Hernandez University of Elche (UMH), 03202 Elche (Alicante), Spain.
E-mail address: j.aparicio@umh.es (Juan Aparicio).

https://doi.org/10.1016/j.softx.2023.101549
Received 20 July 2023; Received in revised form 14 September 2023; Accepted 2 October 2023
Available online 10 October 2023

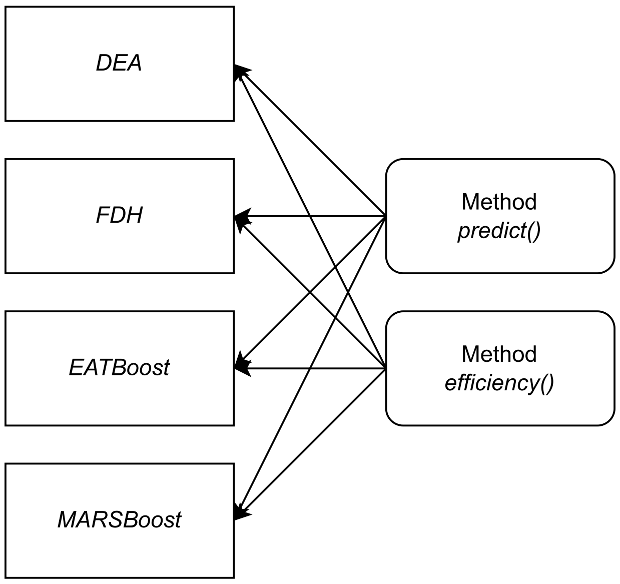

Fig. 1. Software architecture of `boostingDEA`.

a technology that satisfies only free disposability, envelopmentness and minimal extrapolation. In fact, FDH may be considered the "skeleton" of DEA, as the convex hull of the frontier estimated by FDH is identical to the DEA frontier [8].

To make a fair comparison between DMUs, a key aspect is knowing the maximum output a DMU can ideally produce given a certain amount of inputs. However, DEA and FDH tend to overfitting due to the minimal extrapolation principle [9], since this axiom ensures that the estimator of the production function is as close to the data set of DMUs as possible. To overcome this issue, a topic of growing interest is the adaptation of Machine Learning techniques to the efficiency analysis context. In this sense, one major standard machine learning technique is gradient boosting. The basic idea of gradient boosting [10] is to iteratively train a series of weak models, where each model tries to correct the errors made by the previous models. In each iteration, the model is trained on the residuals, which are the differences between the true values and the predicted values of the previous model. The "gradient" in gradient boosting refers to the use of the gradient of the loss function with respect to the model's output to adjust the weights of the data points. This helps the model to focus on the data points that are most difficult to predict. The main advantage of gradient boosting is its ability to produce highly accurate models, especially in case of noisy data or when relationships between the variables are complex [11].

In this paper, we introduce a new R package called boostingDEA that, besides implementing the most popular DEA and FDH models, includes two different boosting algorithms for estimating production frontiers: an adaptation of the Gradient Tree Boosting known as EATBoosting [12,13] and the adaptation of the LS-Boosting algorithm [10] using adapted Multivariate Adaptive Regression Splines (MARS) models [14] as base learners (from now on referred as MARSBoosting [15]). EATBoosting shares similarities with FDH since graphically both generate a step function, while MARSBoosting resembles DEA. However, both algorithms overcome the overfitting problems that characterize standard techniques. Furthermore, in this package, we show how to calculate different technical efficiency measures defined in the literature using different models. In particular, the input and output-oriented radial measures [6], the input and output-oriented Russell measures [16], the Directional Distance Function (DDF) [17], the Weighted Additive Measure (WAM) [18] and the Enhanced Russell Graph measure (ERG) [19], also known as the Slacks-Based Measure (SBM) [20], are included.

2. Software description

In the following subsections, we provide details of the `boostingDEA` architecture and outline the software functionality which include creating the efficiency models, making output predictions and measuring efficiency using them.

2.1. Software architecture

`boostingDEA` implements functions to create four different efficiency models separated in 4 entities: `DEA`, `FDH`, `EATBoost`, and `MARSBoost`. Once created, these models can then be used to make predictions, using the function `predict()`, and to calculate their related efficiency measures, using the function `efficiency()`. This is explained visually in Fig. 1.

2.2. Software functionalities

In the data envelopment analysis context, the productivity and economic performance of a set $\aleph$ of $i = 1, \ldots, n$ observed DMUs is measured. Each of each these DMUs consumes a vector of $j = 1, \ldots, m$ positive inputs $\mathbf{x} \in \mathbb{R}^m_+$ to produce a vector of $r = 1, \ldots, s$ positive outputs $\mathbf{y} \in \mathbb{R}^s_+$. In the package, DMUs' data are managed as `matrix` and/or `data.frame` and their inputs/outputs indexes as `vector`. The set that includes all the technical feasible combinations of $(\mathbf{x}, \mathbf{y})$ is known as the production possibility set or technology, formally defined as $\psi = \{(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{m+s}_+ : \mathbf{x}$ can produce $\mathbf{y}\}$ [21]. Technical efficiency is then defined as the distance from a point belonging to ψ to the production frontier, where the production frontier $\partial(\psi)$ is the "upper" border of the technology. There are various ways of estimating the true frontier of ψ and the true efficiency scores, hence the different models 2.2.1 and their respective predictions 2.2.2. Furthermore, since there are several ways of calculating distances, different efficiency measures are available in the efficiency literature, as explained in 2.2.3.

2.2.1. Creating the models

In the context of DEA, the production technology is calculated under assumptions of free disposability, convexity, envelopmentness and minimal extrapolation. For this case, [6] provide an estimate of the variable returns to scale technology ψ as:

$$\begin{aligned}\psi_{DEA} =&\{(\mathbf{x},\mathbf{y}) \in \mathbb{R}_{+}^{m+s} : y^{(r)} \leq \sum_{i=1}^{n} \lambda_i y_i^{(r)}, r = 1, \dots, s,\\ &x^{(j)} \geq \sum_{i=1}^{n} \lambda_i x_i^{(j)}, j = 1, \dots, m,\\ &\sum_{i=1}^{n} \lambda_i = 1, \lambda_i \geq 0, i = 1, \dots, N\}\end{aligned} \tag{1}$$

This can be computed in the package using the `DEA(data,x,y)` function. This function only requires the `data` containing the units for the analysis and the indexes of the input and output variables (`x` and `y` respectively).

Similarly, FDH also estimates production frontiers, but it is based on just three axioms: free disposability, envelopmentness and minimal extrapolation [7]. In this case, an estimate of the technology is computed as:

$$\begin{aligned}\psi_{FDH} =&\{(\mathbf{x},\mathbf{y}) \in \mathbb{R}_{+}^{m+s} : y^{(r)} \leq \sum_{i=1}^{n} \lambda_i y_i^{(r)}, r = 1, \dots, s,\\ &x^{(j)} \geq \sum_{i=1}^{n} \lambda_i x_i^{(j)}, j = 1, \dots, m,\\ &\sum_{i=1}^{n} \lambda_i = 1, \lambda_i \in \{0,1\}, i = 1, \dots, N\}\end{aligned} \tag{2}$$

Likewise, the FDH model can be computed in R using the `FDH(data,x,y)` function.

As an alternative to the standard FDH technique, the EATBoosting algorithm [12,13] was introduced. This algorithm is an adaptation of the machine learning Gradient Tree Boosting algorithm [10] to estimate production frontiers while satisfying the usual axioms from production theory. In the EATBoosting algorithm, many weak tree-like EAT [9] models are combined in a forward stagewise strategy (where a new predictive model is added in sequence) to generate one strong learner. The final prediction fulfills free disposability and envelopmentness and generates a step-wise surface as an estimator, just as FDH does, but avoids the typical overfitting problem linked to the latter.

The EATBoosting model can be computed in R using the function `EATBoost (data, x, y, num.iterations, num.leaves, learning.rate)`. This function requires the `data` containing the DMUs for the analysis, the indexes of the input and output variables (`x` and `y` respectively) and a set of hyperparameters that control the performance of the model. Those parameters are:

- `num.iterations`, which refers to the number of iterations that the algorithm will perform (i.e., the number of trees in the model).
- `num.leaves`, which limits the number of final leaves of each tree at every iteration.
- `learning.rate`, which applies a weighting factor for the corrections by new trees when added to the model.

MARSBoosting is an adaptation of the Least Squared Boosting (LS-Boosting) algorithm by [10] to estimate production functions, based on the idea of using boosting to ensemble adapted Multivariate Adaptive Regression Splines (MARS) models [14] as base learners. In particular, the adapted LS-Boosting constructs additive regression models by sequentially fitting the adapted MARS models to current pseudo-residuals by least squares at each iteration. MARS algorithm fits piece-wise linear regressions using discrete regression splines at various intervals of the predictor variable space. As a result, the final estimator satisfies envelopmentness, monotonicity and concavity, which allows the approach to be compared to DEA. The final prediction obtained does not have a continuous first derivative, so sharp trend changes can occur. As an alternative, an smoothing procedure can also be applied.

In R, the MARSBoosting model is implemented using the function `MARSBoost(data, x, y, num.iterations, num.terms, learning.rate)`. This function requires the `data`, the indexes of the input (`x`) and output variables (`y`). As hyperparameters, it requires:

- `num.iterations`: the number of iterations the algorithm will perform (i.e., the number of MARS models in the final model).
- `num.terms`: the number of splines created by the MARS algorithm .
- `learning.rate`: the shrinkage factor.

Besides, one of the key aspects of both the EATBoosting and MARSBoosting algorithms is the hyperparameter tuning. To try to find the best value for the hyperparameters, we can resort to a grid of parameter values which can be tested through training and test samples in a user-specified proportion.

In the case of the EATBoosting algorithm, this grid search can be done through the function `bestEATBoost(training, test, x, y, num.iterations, learning.rate, num.leaves, verbose)`, while in the case of the MARSBoosting algorithm this can be done through the function `bestMARSBoost(training, test, x, y, num.iterations, learning.rate, num.terms, verbose)`. These functions receive a vector instead of a single value for each hyperparameter as arguments and evaluate each possible combination in the grid through Mean Squared Error (MSE). Furthermore, through the `verbose` argument, we can specify if we want to "see" the training progress.

2.2.2. Making predictions

In the efficiency context, it is common to measure the performance of one DMU against another DMU [22]. However, to make a fair comparison, an important issue is knowing the maximum output that a DMU can ideally produce given a certain amount of inputs. These estimations can

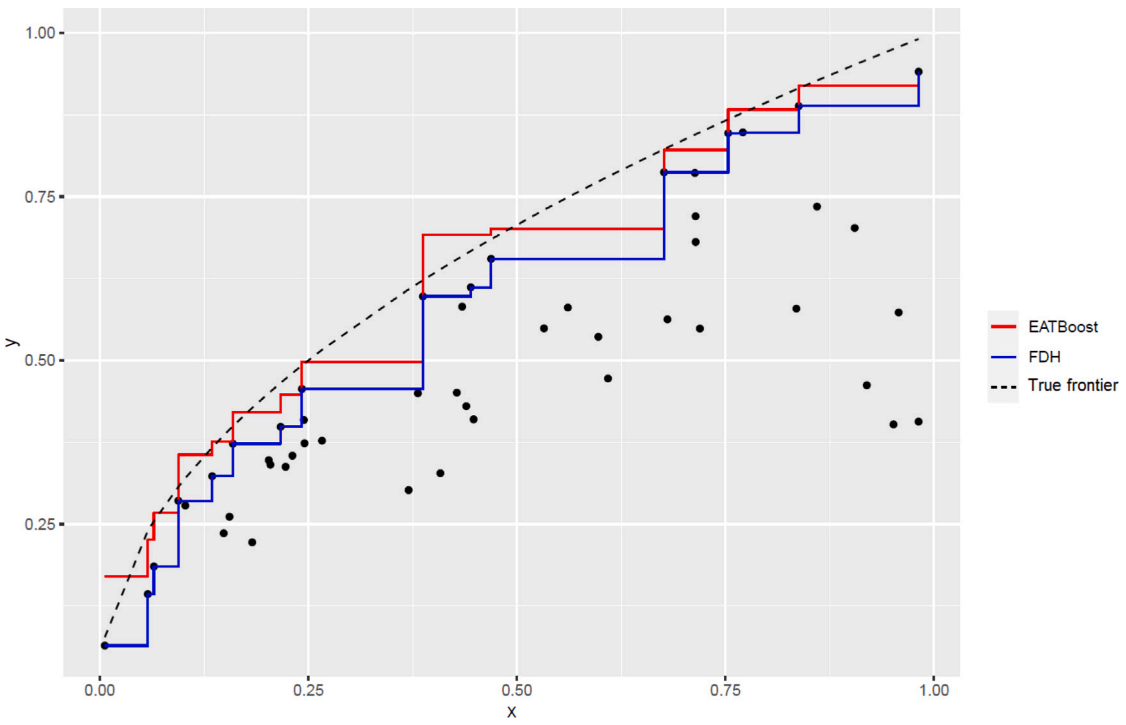

Fig. 2. Example comparing the FDH and the EATBoositng predictions in a single-input single-output scenario with $y = x^{0.5}e^{-u}$, where $x \sim Uni[0,1]$ and $u \sim |N(0,0.4)|$.

be computed in R using `predict(object, newdata, x)` where `object` refers to the previously described `DEA`, `FDH`, `EATBoost`, and `MARSBoost` models. On the other hand, `newdata` refers to a `data.frame` of DMUs and `x`, to the set of input variables indexes. The `data.frame` used as the `newdata` parameter can be the same one as that used by the original DMUs to create the model or a new one, in case of different DMUs, but in the same economic context.

Furthermore, in the case of the MARSBoosting algorithm, both the models with or without the smoothing procedure can be used to compute the predictions. This is specified by an additional parameter `class`. If `class` equals 1 or is unspecified, the model without the smoothing procedure is used, while if it equals 2, the smooth model is applied. Finally, it should be noted that predictions can only be made in single-output scenarios for the `DEA`, `FDH`, and `MARSBoost` models [15], whereas `EATBoost` models can be used in both single-output contexts and multi-output scenarios [12]. Regarding DEA and FDH, these techniques were not originally defined for providing output predictions. Nevertheless, the current version of our package is able to provide these predictions in the single-output case. The multi-output framework will be implemented in future versions of the package. In the case of MARSBoosting, as far as we know, the development of the algorithm is only focused on the single-output scenario (see [15]). Additional developments will be incorporated into the package in future releases when a multi-output version of MARSBoosting is available. In contrast, EATBoosting was originally introduced to provide output predictions for both the single-output scenario and the multi-output production context (see [13]).

To further illustrate the prediction ability of the different algorithms, Fig. 2 compares FDH with EATBoosting and Fig. 3, DEA with MARSBoosting (with and without the smoothing procedure). In both cases, we can see how the prediction made by the boosting algorithms is closer to the true frontier than standard techniques.

2.2.3. Measuring efficiency

In the technical efficiency context, efficiency scores are used to measure the relative efficiency of different DMUs. In the `boostingDEA` package, these scores can be calculated thanks to the function `efficiency(model, measure, data, x, y, . . . )`. For this function, it is necessary to specify the model we want to calculate the score for (i.e., `DEA`, `FDH`, `EATBoost` and `MARSBoost`) and the mathematical programming model which is used to calculate the score, that is, the measure. This is specified using the `model` and the `measure` parameters. If no measure is specified, the radial output measure is calculated. Moreover, the given dataset (`data`) we want to calculate the efficiency scores for and its corresponding input index(es) (`x`) and output index(es) (`y`) must be specified. For best results, it is suggested that the data set with the DMUs whose efficiency is to be calculated match the data set used to estimate the frontier. However, it is also possible to calculate efficiency scores for new data (although infeasibilities can occur if the new unobserved DMUs are outside the computed technologies, resulting in $-\infty$ or $+\infty$ scores depending on if the corresponding optimization program is associated with maximizing or minimizing the objective function, respectively). Additionally, depending on the measure used, extra parameters are required.

The implemented measures are:

- `rad.out`: Output-oriented radial measure [6], which determines the efficiency score for a DMU k with $(\mathbf{x}_k, \mathbf{y}_k) \in R_+^{m+s}$ by equiproportionally increasing all its outputs while maintaining inputs constant.
- `rad.in`: Input-oriented radial measure [6], which determines the efficiency score for an evaluated point $(\mathbf{x}_k, \mathbf{y}_k)$ by equiproportionally decreasing all its inputs while maintaining outputs constant
- `Russell.out`: Output-oriented Russell measure [16], which consider that there might be slack in some but not all of outputs after the output-oriented radial efficiency is achieved.

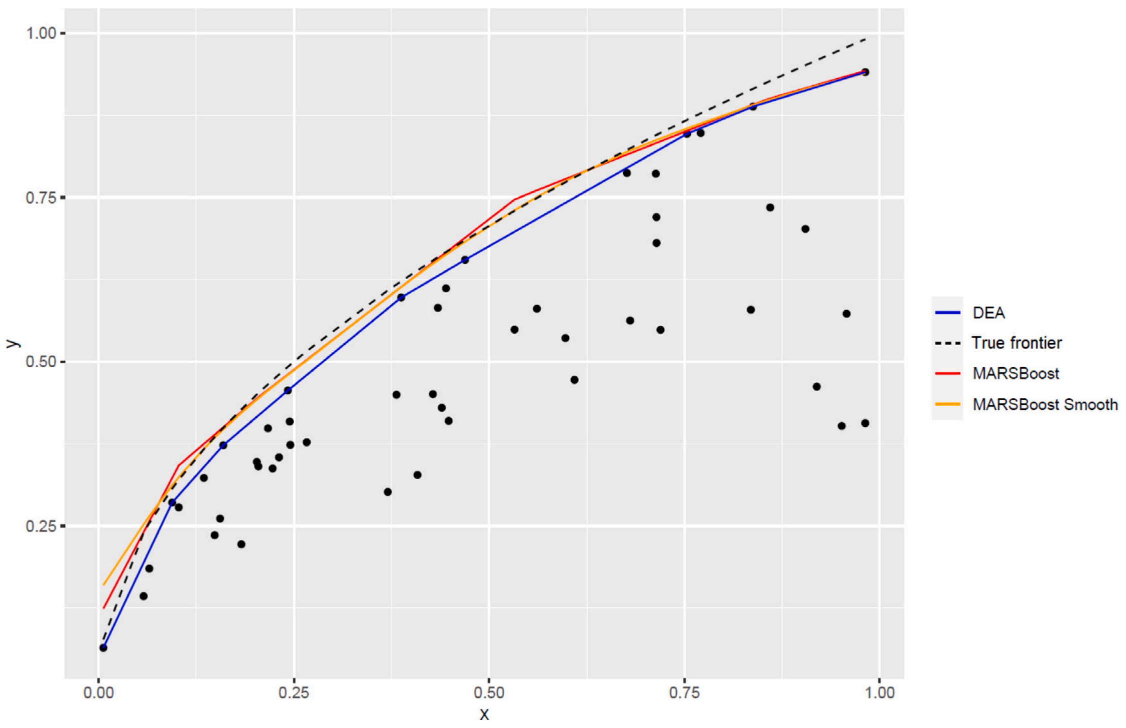

Fig. 3. Example comparing the DEA and the MARSBoosting predictions in a single-input single-output scenario with $y = x^{0.5}e^{-u}$, where $x \sim Uni[0,1]$ and $u \sim |N(0,0.4)|$.

- `Russell.in`: Input-oriented Russell measure, [16], which consider that there might be slack in some but not all of outputs after the input-oriented radial efficiency is achieved.
- `DDF`: Directional Distance Function [17], which projects a DMU $\left(x_k, y_k\right)$ in a preassigned direction $\mathbf{g} = \left(-\mathbf{g}_x, \mathbf{g}_y\right) \neq 0_{m+s}$, with $\mathbf{g}_x \in R_+^m$ and $\mathbf{g}_y \in R_+^s$. The direction vector is specified using the parameter `direction.vector`. Several directional vectors used in the literature are included:

  - The unit vector: $(-\mathbf{g}_x, \mathbf{g}_y) = (\mathbf{1}, \mathbf{1})$.
  - The values of the evaluated observation: $(-\mathbf{g}_x, \mathbf{g}_y) = \left(\mathbf{x}_k, \mathbf{y}_k\right)$.
  - The input and output variables' means: $(-\mathbf{g}_x, \mathbf{g}_y) = (\bar{\mathbf{x}}, \bar{\mathbf{y}})$.
  - A user-specified vector

- `WAM`: Weighted Additive Models [18], which consider that not all units should be of equal importance and introduce some weights $\mathbf{w}^- = (w_1^-, \dots, w_m^+) \in \mathbb{R}_+^m$ and $\mathbf{w}^+ = (w_1^+, \dots, w_s^+) \in \mathbb{R}_+^s$ representing the relative importance of unit inputs and unit outputs, respectively. The package can compute a set of models known as General Efficiency Measures (GEMs) using the parameter `weights`. Those are:

  - The measure of inefficiency proportions (MIP) [23], which uses the weights:

  $$(\mathbf{w}^-, \mathbf{w}^+) = (\frac{1}{\mathbf{x}_k}, \frac{1}{\mathbf{y}_k})$$

  - The range adjusted measure (RAM) [24], which uses the weights:

  $$(\mathbf{w}^-, \mathbf{w}^+) = (\frac{1}{(m+s)R^-}, \frac{1}{(m+s)R^+})$$

  where R^- and R^+ are the inputs and outputs variables' ranges.
  - The bounded adjusted measure (BAM) [25], which uses the weights:

  $$(\mathbf{w}^-, \mathbf{w}^+) = (\frac{1}{(m+s)(\mathbf{x}_k - \underline{\mathbf{x}})}, \frac{1}{(m+s)(\bar{\mathbf{y}} - \mathbf{y}_k)})$$

  where $\underline{\mathbf{x}}$ and $\bar{\mathbf{y}}$ are the minimum and maximum observed values of inputs and outputs, respectively.
  - The normalized weighted additive DEA model [18], which uses the weights:

  $$(\mathbf{w}^-, \mathbf{w}^+) = (\frac{1}{\sigma^-}, \frac{1}{\sigma^+})$$

  where σ^- and σ^+ are the standard deviations of inputs and outputs, respectively.
  - A user-specified vector of weights.

- `ERG`: The Enhanced Russell Graph measure [19], which deals directly with both the input excesses and the output shortfalls of the DMU concerned.

Calculating various measures for EATBoosting models can be highly time-consuming [13]. Therefore, a heuristic approach to compute these scores is provided. The parameter `heuristic` (defaulted to `FALSE`) specifies whether this approach is used. Additionally, when using the MARSBoosting algorithm as the model, only the output-oriented radial measure is currently calculable [15]. Other efficiency measures have not been yet defined in the case of this algorithm. When these measures were defined using MARSBoosting, the package will be updated accordingly.

3. Illustrative example

To demonstrate all the functions available in this package, we utilize a single data set that is included in the package itself. The data set comprises data on banks that were operational in Taiwan in 2010 (originally sourced from [26]). The data set `banks` contains 31 banks with 6 variables. Those variables are:

- `Financial.funds`: deposits and borrowed funds (in millions of TWD).
- `Labor`: number of employees.
- `Physical.capital`: net amount of fixed assets (in millions of TWD).
- `Financial.investments`: financial assets, securities, and equity investments (in millions of TWD).
- `Loans`: loans and discounts (in millions of TWD).
- `Revenue`: interests from financial investments and loans.

The first three variables are considered as inputs for the models, while the following two are considered as outputs. Moreover, the variable `Revenue` is interpreted as a combination of the other two output variables, and can be used as the target variable for a single-output scenario.

```
# Load the database
R> data("banks")
# Save the input and output indexes
R> x <- 1:3
R> y <- 4:5
# Save the matrix of inputs and outputs
R> input <- banks[,x]
R> output <- banks[,y]
# Create training ans test sets
R> N <- nrow(banks)
R> selected <- sample(1:N, N * 0.8)    # Training indexes
R> training <- banks[selected, ]       # Training set
R> test <- banks[- selected, ]         # Test set
```

Here it is an example of how to create all the different models:

```
# Creates a DEA model with:
# 3 inputs: Financial.funds, Labor and Physical.capital
# 1 output: Revenue
R> DEA_model <- DEA(data = banks, x = 1:3, y = 6)
```

```
# Creates a FDH model with:
# 3 inputs: Financial.funds, Labor and Physical.capital
# 1 output: Revenue
R> FDH_model <- FDH(data = banks, x = 1:3, y = 6)
```

```
# Creates an EATBoosting model with:
# 3 inputs:  Financial.funds, Labor and Physical.capital
# 2 outputs: Financial.investments, Loans
# Search of best hyperparameters for an EATBoosting model
R> grid_EATBoost <- bestEATBoost(
  training = training, test = test,
  x = 1:3, y = 4:5,
  num.iterations = c(5, 6, 7),
  learning.rate = c(0.4, 0.5, 0.6),
  num.leaves = c(6, 7, 8),
  verbose = FALSE
)
# Best EATBoosting model
R> EATBoost_best <- EATBoost(
  data = banks, x = 1:3, y = 4:5,
  num.iterations = grid_EATBoost[1, "num.iterations"],
  learning.rate = grid_EATBoost[1, "learning.rate"],
  num.leaves = grid_EATBoost[1, "num.leaves"]
)
```

```
# Creates a MARSBoosting model with:
# 3 inputs: Financial.funds, Labor and Physical.capital
# 1 output: Revenue
# Search of best hyperparameters for a MARSBoosting model
R> grid_MARSBoost <- bestMARSBoost(
  training = training, test = test,
  x = 1:3, y = 6,
  num.iterations = c(5,6,7),
  learning.rate = c(0.4, 0.5, 0.6),
  num.terms = c(6, 8, 10),
  verbose = FALSE
)
# Best MARSBoosting model
R> MARSBoost_best <- MARSBoost(
  data = banks, x = 1:3, y = 6,
  num.iterations = grid_MARSBoost[1, "num.iterations"],
  learning.rate = grid_MARSBoost[1, "learning.rate"],
  num.terms = grid_MARSBoost[1, "num.terms"]
)
```

Next, we can see how to make output predictions.

```
# DEA prediction
R> predict(object = DEA_model, newdata = banks,
           x = 1:3, y = 6)
   Revenue_pred
1      1056.000
2     41007.000
3     23610.840
4      4267.654
5     31506.000
6     35510.000}
```

```
# MARSBoosting prediction (with the smoothing procedure)
R> predict(object = MARSBoost_best, newdata = banks,
           x = 1:3, class = 2)
   Revenue_pred
1      1121.456
2     43058.293
3     25857.201
4      5156.947
5     32379.240
6     35519.006
```

Finally, here is an example of how to calculate some efficiency measures:

```
R> efficiency(model = EATBoost_best, heuristic = FALSE,
              measure = "Russell.in", data = banks,
              x = 1:3, y = 4:5)
                          EATBoost.Russell.in
Export-Import Bank                  1.0000000
Bank of Taiwan                      0.3440001
Taipei Fubon Bank                   0.3068248
Bank of Kaohsiung                   0.4822518
Land Bank                           0.3525174
Cooperative Bank                    0.2897980
```

```
R> efficiency(model = EATBoost_best, heuristic = FALSE,
              measure = "DDF", direction.vector = "mean",
              data = banks, x = 1:3, y = 4:5)
                          EATBoost.DDF
Export-Import Bank        0.0000000000
Bank of Taiwan            0.0000000000
Taipei Fubon Bank         0.2346833493
Bank of Kaohsiung         0.0000000000
Land Bank                 0.0000000000
Cooperative Bank          0.1363531594
```

4. Impact and conclusions

Although there are currently several `R` packages available for estimating technical efficiency, including `deaR` [27] for the estimation of technical efficiency using both classic and fuzzy DEA moles; `Benchmarking` [28] to calculate DEA under different technology assumptions; `productivity` [29] providing various indexes related to DEA; or `FEAR` [30] allowing computing non-parametric efficiency estimates, making inference, and testing hypotheses in frontier models; the `boostingDEA` package differs by offering an alternative based on machine learning to estimate production frontiers. This machine learning approach relies on boosting to overcome the overfitting problems that characterize DEA and FDH (see e.g. [9]). Specifically, `boostingDEA` implements an adaptation of the Gradient Tree Boosting algorithm as well as an adaptation of the LS-Boosting algorithm using MARS models as base learners. Besides, it is shown how to tune these models to determine the best combination of hyperparemeters.

The package provides functions to make ideal output estimation given a certain amount of inputs and functions to calculate different efficiency measures such as the input and output-oriented radial measures, the input and output-oriented Russell measures, the Directional Distance Function (DDF), the Weighted Additive Measure (WAM) and the Enhanced Russell Graph measure (ERG). Furthermore, to illustrate all the models in the package, a data set consisting of banks operating in Taiwan is also provided. Finally, the code is open-source and freely available on a repository (https://github.com/itsmeryguillen/boostingDEA) for users to modify, extend, and adapt according to their needs.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: The authors report financial support was provided by the Spanish Ministry of Science and Innovation and by the Valencian Community (Spain).

Data availability

Data will be made available on request.

Acknowledgments

The authors thank the financial support from the Spanish Ministry of Science and Innovation and the State Research Agency under grants PID2019-105952GB-I00 and PID2022-136383NB-I00/ AEI /10.13039/501100011033. Additionally, Juan Aparicio, Maria D. Guillen and Victor J. España thank the grants PROMETEO/2021/063, CIACIF/2021/345 and ACIF/2021/135 respectively, funded by the Valencian Community (Spain).

References

[1] Pastor JT, Aparicio J, Zofío JL. Benchmarking economic efficiency. Internat Ser Oper Res Management Sci 2022.
[2] O'Donnell CJ. Productivity and efficiency analysis. Springer; 2018.
[3] Aigner DJ, Chu S-f. On estimating the industry production function. Am Econ Rev 1968;58(4):826–39.
[4] Charnes A, Cooper WW, Rhodes E. Measuring the efficiency of decision making units. European J Oper Res 1978;2(6):429–44.
[5] Orea L, Zofío JL. Common methodological choices in nonparametric and parametric analyses of firms' performance. Palgrave Handb Econ Perform Anal 2019;419–84.
[6] Banker RD, Charnes A, Cooper WW. Some models for estimating technical and scale inefficiencies in data envelopment analysis. Manag Sci 1984;30(9):1078–92.
[7] Deprins D, Simar L, Tulkens H. Measuring labor-efficiency in post offices, no. 571. LIDAM Reprints CORE, Université catholique de Louvain, Center for Operations Research and Econometrics (CORE); 1984.
[8] Daraio C, Simar L. Introducing environmental variables in nonparametric frontier models: a probabilistic approach. J Prod Anal 2005;24(1):93–121.
[9] Esteve M, Aparicio J, Rabasa A, Rodriguez-Sala JJ. Efficiency analysis trees: A new methodology for estimating production frontiers through decision trees. Expert Syst Appl 2020;162:113783.
[10] Friedman JH. Greedy function approximation: a gradient boosting machine. Ann Stat 2001;1189–232.
[11] Hastie T, Tibshirani R, Friedman JH, Friedman JH. The elements of statistical learning: data mining, inference, and prediction, vol. 2. Springer; 2009.
[12] Guillen MD, Aparicio J, Esteve M. Gradient tree boosting and the estimation of production frontiers. Expert Syst Appl 2023;214:119134.
[13] Guillen MD, Aparicio J, Esteve M. Performance evaluation of decision making units through boosting methods in the context of free disposal hull: some exact and heuristic algorithms. Int J Inf Technol Decis Mak 2022.
[14] Friedman JH. Multivariate adaptive regression splines. Ann Stat 1991;19(1):1–67.
[15] España VJ, Aparicio J, Barber X, Esteve M. Estimating production functions through additive models based on regression splines. Eur J Oper Res 2024;312(2):684–99.
[16] Färe R, Lovell CK. Measuring the technical efficiency of production. J Econom Theory 1978;19(1):150–62.
[17] Chambers RG, Chung Y, Färe R. Profit, directional distance functions, and Nerlovian efficiency. J Optim Theory Appl 1998;98:351–64.
[18] Lovell CK, Pastor JT. Units invariant and translation invariant DEA models. Oper Res Lett 1995;18(3):147–51.
[19] Pastor JT, Ruiz JL, Sirvent I. An enhanced DEA Russell graph efficiency measure. European J Oper Res 1999;115(3):596–607.
[20] Tone K. A slacks-based measure of efficiency in data envelopment analysis. European J Oper Res 2001;130(3):498–509.
[21] Färe R, Primont D. Multi-output production and duality: Theory and applications. Springer science & business media; 1995.
[22] Bogetoft P, Otto L. Benchmarking with Dea, Sfa, and R, vol. 157. Springer Science & Business Media; 2010.
[23] Charnes A, Cooper WW, Wei Q. A semi-infinite multicriteria programming approach to data envelopment analysis with infinitely many decision-making units. Tech. rep., Texas univ. at Austin center for cybernetic studies; 1987.
[24] Cooper WW, Park KS, Pastor JT. RAM: a range adjusted measure of inefficiency for use with additive models, and relations to other models and measures in DEA. J Prod Anal 1999;11:5–42.
[25] Cooper WW, Pastor JT, Borras F, Aparicio J, Pastor D. BAM: a bounded adjusted measure of efficiency for use with bounded additive models. J Prod Anal 2011;35:85–94.
[26] Juo J-C, Fu T-T, Yu M-M, Lin Y-H. Profit-oriented productivity change. Omega 2015;57:176–87.
[27] Coll-Serrano V, Bolos V, Suarez RB. deaR: Conventional and fuzzy data envelopment analysis. 2022, R package version 1.3.1.
[28] Bogetoft P, Otto L. Benchmarking with DEA and SFA. 2022, R package version 0.31.
[29] Dakpo KH, Desjeux Y, Latruffe L. productivity: Indices of productivity and profitability using data envelopment analysis (DEA). 2018, R package version 1.1.0.
[30] Wilson PW. FEAR: A software package for frontier efficiency analysis with R. Socio-Econ Plan Sci 2008;42(4):247–54.