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

ARTICLE INFO

Keywords:
Data envelopment analysis
Free Disposal Hull
Boosting
Technical efficiency

ABSTRACT

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 $\boldsymbol{x}_j = \left(x_{1j}, \ldots, x_{mj}\right) \in R_+^m$ 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 $\widehat{f}_{FDH}(\boldsymbol{x}) = \max_{j: x_i \geqslant x_{ij}, \forall i} \left\{y_{1j}\right\}$..

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 $f_0(\boldsymbol{x}) = \underset{\gamma}{\operatorname{argmin}} \sum_{j=1}^{n} L(y_{1j}, \gamma)$, 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 $\gamma_q = \underset{\gamma}{\operatorname{argmin}} \sum_{j=1}^{n} L\left(y_{1j}, f_{q-1}(\boldsymbol{x}_j) + \gamma h_q(\boldsymbol{x}_j)\right)$ (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 $I(\boldsymbol{x} \in R_{hq}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \in R_{hq} \\ 0 & \text{if } \boldsymbol{x} \notin R_{hq} \end{cases}$. 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 $f_q(\boldsymbol{x}) = f_{q-1}(\boldsymbol{x}) + \sum_{h=1}^{J_q} \gamma_{hq} I(\boldsymbol{x} \in R_{hq})$ and $\gamma_{hq} = \underset{\gamma}{\operatorname{argmin}} \sum_{x_j \in R_{hq}} L\left(y_{1j}, f_{q-1}(x_j) + \gamma\right)$. 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 $f_0(\boldsymbol{x}) = \underset{\gamma}{\operatorname{argmin}} \sum_{j=1}^{n} L(y_{1j}, \gamma)$

1. For $q = 1$ to$Q$:
   a. For $j = 1, 2, ..., n$ calculate $e_{jq} = -\left[\frac{\partial L(y_{1j}, f(x_j))}{\partial f(x_j)}\right]_{f=f_{q-1}}$
   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 $\gamma_{hq} = \underset{\gamma}{\operatorname{argmin}} \sum_{x_j \in R_{hq}} L\left(y_{1j}, f_{q-1}(x_j) + \gamma\right)$
   d. Update $f_q(\boldsymbol{x}) = f_{q-1}(\boldsymbol{x}) + \sum_{h=1}^{J_q} \gamma_{hq} I(\boldsymbol{x} \in R_{hq})$

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

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 $f_q(\boldsymbol{x}) = f_{q-1}(\boldsymbol{x}) + \nu \sum_{h=1}^{J_q} \gamma_{hq} I(\boldsymbol{x} \in R_{hq})$. 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 $\aleph = \left\{\left(\boldsymbol{x}_j, \boldsymbol{y}_j\right)\right\}_{j=1,...,n}$, 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 $\left(x_i^*, s_i^*\right)$ that minimizes the expression $R(t_L) + R(t_R) = \frac{1}{n}\sum_{(x_j, y_j) \in t_L} \sum_{r=1}^{s} \left(y_{rj} - y_r(t_L)\right)^2 + \frac{1}{n}\sum_{(x_j, y_j) \in t_R} \sum_{r=1}^{s} \left(y_{rj} - y_r(t_R)\right)^2$, 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 $R_t = \left\{\boldsymbol{x} \in R_+^m : a_{it} \leqslant x_i < b_{it}, i = 1, ..., m\right\}$. 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

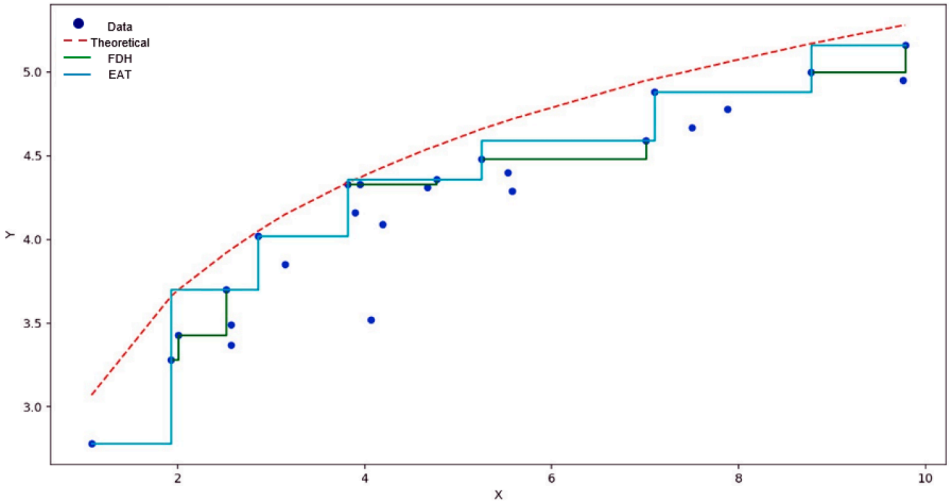

Fig. 1. Graphical illustration of the frontier estimates provided by FDH and EAT (Esteve et al, 2021).

(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., $y_r(t_R) = y_r(t^*)$, $r = 1, ..., s$, and the estimation of the left child node is defined as follows:

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

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 $t' \in \widetilde{T}(k|t^* \to t_L, t_R)$, $r = 1, ..., s$.

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 $r = 1, ..., s$, with $I(\cdot)$ being the indication function. From this estimator, the technology estimated from the EAT algorithm would be as follows:

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

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:

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

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 $\frac{1}{2}\left(y_{1j} - \widehat{f}(\boldsymbol{x}_j)\right)^2, \forall j = 1, ..., n$ (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 $j = 1, ..., n$, 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 $f_0(\boldsymbol{x}) = \max_{j=1,...,n}\left\{y_{1j}\right\}$. Now, according to Friedman (2001), 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 (steps 2.a and 2.b). The components of the gradient $\boldsymbol{e}_q$ are now $e_{jq} = -\left[\frac{\partial L(y_{1j}, f(\boldsymbol{x}_j))}{\partial f(\boldsymbol{x}_j)}\right]_{f=f_{q-1}} = y_{1j} - f_{q-1}(\boldsymbol{x}_j)$ 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 $\aleph_q = \left\{ (\boldsymbol{x}_j, e_{jq}) \right\}_{j=1,...,n}$, 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 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 γ_{hq} that minimizes the loss function over the region R_{hq} is the estimation of the output given by CART for the final node associated with 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 $f_0(\boldsymbol{x}) = \max_{j=1,...,n}\left\{y_{1j}\right\}$

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

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 ν.

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

Proof. Given $q = 1, ..., Q$, 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 $\nu > 0$, $h_q^{\nu}(\boldsymbol{x}) = \nu \sum_{h=1}^{J_q} \gamma_{hq} I(\boldsymbol{x} \in R_{hq})$ is also a monotone non-decreasing function. Then, $f_q(\boldsymbol{x}) = f_{q-1}(\boldsymbol{x}) + \nu \sum_{h=1}^{J_q} \gamma_{hq} I(\boldsymbol{x} \in R_{hq})$ 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, $f_0(\boldsymbol{x}) = \max_{j=1,...,n}\left\{y_{1j}\right\}$, 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 $\nu \in (0, 1]$. Then, $\widehat{f}(\boldsymbol{x}_j) \geqslant y_{1j}$, for all $j = 1, ..., n$.*

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

$\{(x_j, e_{jq})\}_{j=1,\dots,n}$ satisfies that $h_q(x_j) = \sum_{h=1}^{J_q} \gamma_{hq} I(x_j \in R_{hq}) \geqslant e_{jq}$, for all $j = 1,\dots,n$. 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 $j = 1,\dots, n$. Finally, by induction, it is enough to prove that $f_0(x_j) \geqslant y_{1j}$, for all $j = 1, \dots, n$. This last statement is trivial because, by definition, $f_0(x_j) = \max_{j=1,\dots,n}\{y_{1j}\}$

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 $f_0(x_j) = \max_{j=1,\dots,n}\{y_{1j}\}$. As a result, $e_{1j} = y_{1j} - f_0(x_j) = y_{1j} - \max_{j=1,\dots,n}\{y_{1j}\}$. Additionally, if the training data are transformed in the way $y'_{1j} = y_{1j} + k$, $\forall j = 1, \dots, n$, 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 $\max_{j=1,\dots,n}\{y_{1j}\}$ is a constant. Therefore, if $J_1 = \left|\widetilde{T}(\aleph)\right|$, then $\gamma_{h1} = d_1^{T(\aleph')}(R_{h1}) = d_1^{T(\aleph)}(R_{h1}) - \max_{j=1,\dots,n}\{y_{1j}\}$, for all $h = 1, 2, \dots, J_1$. In this way, we have $\widehat{f}(x) = f_1(x) = f_0(x) + \sum_{h=1}^{J_1} \gamma_{h1} I(x \in R_{h1}) = \max_{j=1,\dots,n}\{y_{1j}\} + \sum_{h=1}^{J_1}\left(d_1^{T(\aleph)}(R_{h1}) - \max_{j=1,\dots,n}\{y_{1j}\}\right) I(x \in R_{h1}) = \max_{j=1,\dots,n}\{y_{1j}\} - \max_{j=1,\dots,n}\{y_{1j}\} + \sum_{h=1}^{J_1} d_1^{T(\aleph)}(R_{h1}) I(x \in R_{h1}) = \sum_{h=1}^{J_1} d_1^{T(\aleph)}(R_{h1}) I(x \in R_{h1}) = d_1^{T(\aleph)}(x)$. ■.

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 $e_{rjq} = -\left[\frac{\partial L(y_j, f(x_j))}{\partial f_r(x_j)}\right]_{f_r(x_j) = f_{rq-1}(x_j)} = y_{rj} - f_r(x_j)$ $r = 1, \dots, s$ 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: $\widehat{f}(x) = \left(\widehat{f}_1(x), \dots, \widehat{f}_s(x)\right)$. The multi-output version of the EATBoost algorithm is summarized in Algorithm 3.

Algorithm 3: EATBoost (multi-output version).

Set $f_0(x) = \left(\max_{j=1,\dots,n}\{y_{1j}\}, \dots, \max_{j=1,\dots,n}\{y_{sj}\}\right)$

1. For $q = 1$ to Q:
   a. For $j = 1, 2, \dots, n$ calculate $e_{jq} = y_j - f_{q-1}(x)$
   b. Fit a deep EAT to the targets e_{jq} given terminal regions R_{hq}, $h = 1, 2, \dots, J_q$
   c. For $h = 1, 2, \dots, J_q$ 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 $r = 1, \dots, s$.*

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 $j = 1, \dots, n$, and for all $r = 1, \dots, s$.*

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 $j = 1, \dots, n$;
(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, ν); 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 $\widehat{f}(\boldsymbol{x}|Q, J, \nu)$. Then, the 'quality' of this estimator is assessed by calculating the difference between the observed output y_j and the estimated output $\widehat{f}(\boldsymbol{x}_j|Q, J, \nu)$, 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, ν) is checked. Following the literature (see Hastie et al., 2009), $4 \leqslant J \leqslant 10$ and $0 < \nu \leqslant 0.1$. In this way, a final combination (Q^*, J^*, ν^*) 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^*, ν^*), obtaining the estimation function $\widehat{f}(\boldsymbol{x}|Q^*, J^*, \nu^*)$, 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 $\widehat{f}(\boldsymbol{x})/y$..

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

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

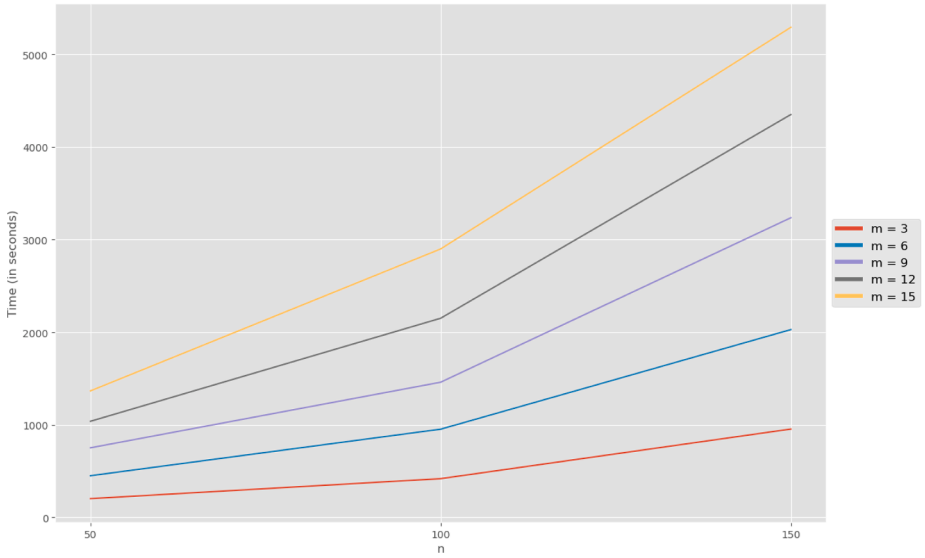

Fig. 2. Computational time for EATBoost.

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

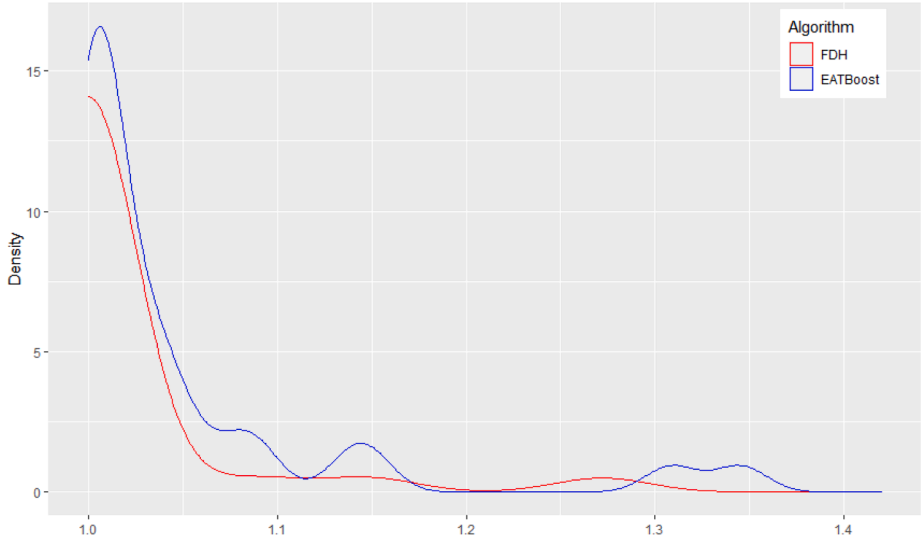

Fig. 3. Efficiency score distributions.

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.