Training trees on tails with applications to portfolio choice

Guillaume Coqueret, Tony Guida

Noname manuscript No.
(will be inserted by the editor)

Training trees on tails with applications to portfolio choice

Guillaume Coqueret · Tony Guida

the date of receipt and acceptance should be inserted later

Abstract In this article, we investigate the impact of truncating training data when fitting regression trees. We argue that training times can be curtailed by reducing the training sample without any loss in out-of-sample accuracy as long as the prediction model has been trained on the *tails* of the dependent variable, that is, when 'average' observations have been discarded from the training sample. Filtering instances has an impact on the features that are selected to yield the splits and can help reduce overfitting by favoring predictors with monotonous impacts on the dependent variable. We test this technique in an out-of-sample exercise of portfolio selection which shows its benefits. The implications of our results are decisive for time-consuming tasks such as hyperparameter tuning and validation.

Keywords Decision trees · Filtering training set · Factor investing · Portfolio choice · Feature selection

1 Introduction

Decision trees and their extensions are known to be quite efficient forecasting tools when working on tabular (highly structured) data. Recently, the surge in Machine Learning applications in finance has led to multiple publications that use tree methods in portfolio allocation problems.[1] The underlying rationale behind the choice of trees is that they split firms into homogeneous clusters in terms of performance. Hence, the investor will naturally bet on the clusters with the highest (forecasted) profitability and evidently avoid the underperforming groups. Extensions such as random forests or boosted trees help refine the predictions and improve accuracy and their construction follows the same spirit.

Our article relates to four connected streams of the literature dedicated to decision trees and to machine learning (ML) more generally. The first stream pertains to enhancements that seek to meliorate the design of tree structures. Indeed, the top-down construction of decision trees is known to lead to sub-optimal structures (see, e.g., the references in Norouzi, Collins, Johnson, Fleet, and Kohli (2015) for general trees, or in Loh et al. (2009) and Bertsimas and Dunn (2017) in the case of classification trees). One reason for that is that early splits may overlook important features to the benefit of noisy variables. In turn, this may lead to spurious partitions and inefficient predictions. In addition to the ones mentioned above, most of the contributions that seek to improve tree algorithms do so by refining the splitting decision (see, e.g., Chou (1991), Lopez, Milhaud, Thérond, et al. (2016)). The premise of our paper is different: we stick to classical splitting rules but investigate if working on subsamples can help improve performance. The rationale behind this choice is that instances are probably not equal and some are likely to carry more signal than others. One benefit of this approach is that many advanced models (random forests and boosted trees) aggregate

G. Coqueret (corresponding author)
EMLYON Business School, 23 avenue Guy de Collongue, 69130 Ecully, France
E-mail: coqueret@em-lyon.com

T. Guida
RAM Active Investments, 8 rue du Rhone, 1204, Geneva, Switzerland
E-mail: tgu@ram-ai.com

[1] See, e.g., Ballings, Van den Poel, Hespeels, and Gryp (2015), Patel, Shah, Thakkar, and Kotecha (2015), Moritz and Zimmermann (2016), Krauss, Do, and Huck (2017), Gu, Kelly, and Xiu (2019) and Huck (2019). Some papers use tree-based scenarios, like Köksalan and Şakar (2016) and Yan, Chen, Consigli, Liu, and Jin (2019), but those are very different tools.

simple trees and our method translates to these techniques seamlessly. Consequently, resorting to our simple method does not impose to recode dedicated libraries whereas enforcing alterations of the splitting rules requires a fastidious coding effort.

Our research also relates to a second concomitant stream of the machine learning literature: feature selection. We refer for instance to the survey articles Guyon and Elisseeff (2003) and Chandrashekar and Sahin (2014) and to the monographs Liu and Motoda (2012) and Kuhn and Johnson (2019) for more details on this topic. In line with the '*garbage in, garbage out*' principle, the purpose of feature selection is to feed the machine learning engine a relevant set of predictors to maximize the quality of future forecasts. Usually, this is performed on the whole dataset before the training phase. As we will show, altering the training set by removing instances will naturally impact the way features are selected as splitting variables. Hence, focusing on a particular subset of observations will have an effect similar to raw feature selection and this effect diffuses at each level of the tree.

The third theme is overfitting. Most often, the proposed solution for decision trees is pruning (Esposito, Malerba, and Semeraro (1997), Stahl and Bramer (2012)) because the shortest way to simplify a tree is to cut its branches. We believe that another route is to consider partitions that are not too extreme and that lead to clusters that are relatively balanced. Indeed, early splits that lead to very small nodes are more likely to come from flukes than long lasting patterns. Under some circumstances, our filtering approach overcomes this shortcoming by leading to splits that are less stiff compared to the original ones.

Lastly, the fourth domain is active learning (see, e.g., Settles (2012)). In active learning, the algorithms learn on a fraction of the training sample: they ask for the labels of some particular instances. Our approach is somewhat similar because the full sample is not considered. Nonetheless, the main difference is that, in active learning, it is the learner (algorithm) that chooses the subsample on which it trains. In our extreme learning, it is the user (human) that filters the instances.

In this paper, we analyze the impact of filtering the training set before building a decision tree. More specifically, we remove the instances that correspond to the 'average' values of the dependent variable. For instance, if this variable is a simple return, then we keep only the high and low returns and get rid of the bulk of the distribution. While this may seem suboptimal because it mechanically removes possibly worthy information, we show that the impact on the structure of the tree may in fact be beneficial. In addition, our method is incredibly simple and contrasts with more sophisticated and computationally intense methods of prior dimensionality reduction (as in Ali and Yaman (2013) for example).

A simplistic justification of the filtering can be formulated as follows. Money managers are mostly interested in high returns (which they want) and low returns (which they seek to avoid at all cost). Hence, why bother with the intermediate returns, which, in fact, are susceptible to carry more noise than signal? As we show, the implications are more complex. In a classical regression tree, when a split is performed, the focus is only set on minimizing the total dispersion stemming from the two resulting clusters. Often, this is not optimal for many reasons, of which we list two:

- a very local pattern is identified by the tree that only holds for the training sample (and not for any other reasonable test sample);
- the tree focuses on variables that have a good fit but with erratic (and possibly spurious) impact on the predicted variable.

We argue that in some cases, filtering the training dataset helps overcome these two issues. In all of our tests, the filtering procedure yielded partitions that were better balanced and less focused on localized idiosyncrasies. Moreover, our investigations show that the filter gives more importance to (i.e., selects) the features that have a monotonous impact on the predicted variable. This is an essential property because in finance, researchers prefer when the relationship is monotonous, which is for instance one of the sought properties in factors defined in the asset pricing literature, as reported in Tables IV and V of Fama and French (1992).[2]

The idea of manually filtering occurrences is not new: several articles have already tested this idea successfully (Fu, Du, Guo, Liu, Dong, and Duan (2018) and Guida and Coqueret (2018)). But, to the best of our knowledge, no study explains why filtering could or should work. The present paper is intended to fill this void. Moreover, we present en route some previously undocumented theoretical properties of splits

[2] One reason why monotonous patterns matter is that they are robust, hence an investor can confidently rely on them to build portfolios. However, in practice, monotonicity tests are seldom carried out even though they exist. Romano and Wolf (2013) detail one such procedure for instance. For simplicity, so-called factors are often determined by looking at differences related to extreme values of features. The whole distribution of features is rarely exploited for at least two reasons. First, it is less straightforward and harder to assess. Second, it is likely to reduce the significance of factors. Because of the publication bias towards positive results, academics seek statistical significance and thus ignore tests that are likely to curtail it.

of regression trees.

The main contributions of the paper are threefold.

1. First, we show a local effect of the filter: it shifts extreme splits back to the center. This can be desirable because more moderate splits are less prone to overfitting. Extreme splits can very well stem from idiosyncrasies in the training set that generalize poorly out-of-sample.

2. Second, at the multivariate (global) level, we demonstrate that filtering the training set favors features that display a monotone pattern with respect to the label. We believe that these monotone patterns should be pushed forward because they are more likely to be lasting and robust.

3. Lastly, we confirms our theoretical findings with a portfolio backtest based on a large financial dataset. It shows that our method is *often* more efficient and *always* much faster than the classical approach that consists in feeding predictive algorithms with all the data at hand.

The paper is structured as follows. In Section 2, we briefly expose the core ideas of the paper via a simple example. In Section 3, we investigate the location of optimal splits based on the underlying data generator which specifies the relationship between the label and each feature separately. In Section 4, we go one step further and quantify the gains associated to particular generators. Section 5 is the core of the paper and is dedicated to the impact of filtering on the gains realized by splits. In Section 6, we apply the filtering technique in an out-of-sample portfolio selection exercise. Finally, Section 7 concludes. All proofs are gathered in the appendix.

Lastly, because we believe reproducibility is an important facet of empirical research, we share both the data and code that were used to produce the results of the paper.[3]

2 Preliminary example

Before we formally introduce the theoretical concepts behind the ideas in the paper, it is preferable for pedagogical purposes to start with a simple illustration that will guide the reader. Fundamentally, decision trees are clustering algorithms that seek to group observations in homogeneous bundles where homogeneity is calculated with respect to one variable in particular. In the money management industry, this dependent variable is a performance indicator, like an average return or a risk-adjusted metric, like the Sharpe ratio. Once the tree is built, the clusters encompass assets which have similar types of performance (e.g., low, medium, high). Naturally, the investor will seek to allocate wealth to those with the highest levels of predicted performance. In the context of factor investing, the trees are constructed based on a large set of firm-specific predictors. These features are those that will determine how the clusters are built, i.e., how assets are grouped.

The example below is by construction hand-picked because it reflects the two points we make later on in the paper. The dataset is the one described below in Section 6. We focus on features gathered in the calendar year 2002. The dependent variable in the one year ahead return of the stocks. The asset manager wishes to understand which variables in 2002 impacted this future annual return and builds a first simple regression tree, depicted in the left graph of Figure 1.

The average return in the sample is high (51%, shown in the rounded square) in part because it is driven upwards by a few small firms that achieve high levels of profitability during this period. The first splitting criterion is past short term (one year) volatility and the split is located at the 94% quantile of this variable. Stocks with very high volatility over this period belong to the right bucket, with an average return of 170% while those with more moderate variations are located in the left bucket, with an average return of 44%. The growing of the tree then continues. For the left group, the next split is driven by the market capitalization while for the right group it is performed according to the ratio of free cash flows divided by book value (those are accounting items). The output encompasses 4 groups and each has a different size and a different average return. The leftmost group has the lowest return (39%) but gathers 88% (also shown in a rounded square) of the sample while the rightmost group reaches an impressive 340% return, but over only 1% of the sample.

The right tree is built on a smaller sample which contains the instances with extreme returns (top and bottom 20%). Because of asymmetries in returns, the average value is higher compared to the one in the left tree. There are notably two differences with the previous tree that we want to highlight. The first is obviously that the splitting variables are not the same. A natural question, which we partly answer below,

[3] The material can be accessed here: www.gcoqueret.com/tot.html

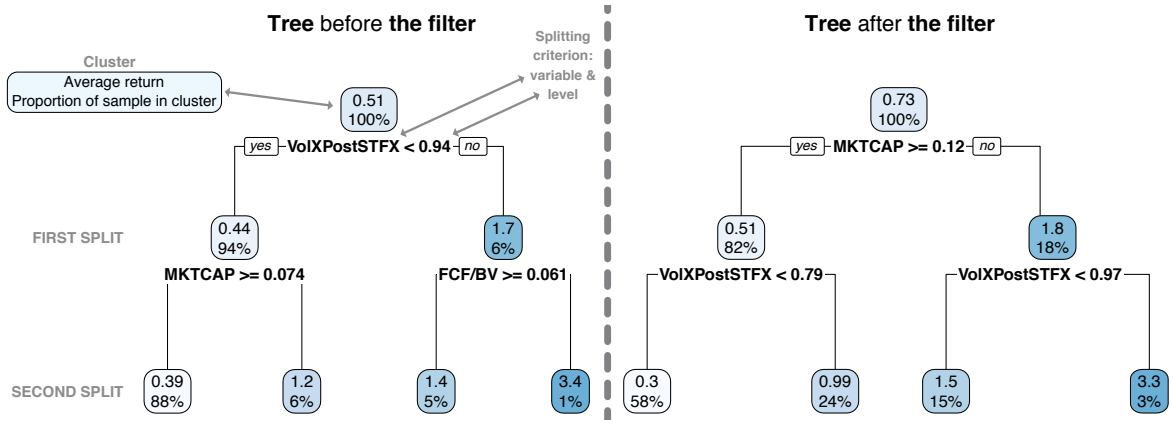

Fig. 1 A first example: trees based on data from year 2002. The dependent variable is the 12 month future return. The filtered dataset retains only the observations with top and bottom 20% values for the dependent variable. The items **VolXPostSTFX**, **MKTCAP** and **FCF/BV** relate to ex-post (i.e., past) short term volatility, market capitalization and free cash flows divided by book values, respectively.

is: what is the difference between the two and is one option better than the other? The second difference is the level of split locations: they are closer to the median (0.5) in the second tree. One direct implication is that the four clusters are much more balanced in terms of sizes: between 3% and 58% of the sample in the clusters after two splits, versus 1% to 88% in the first tree. This underlines one potential risk and drawback associated to trees (and ML methods in general): the tendency to overfit. Indeed, the small cluster in the first tree with 340% average return may very well be attractive to the potential investor. Nevertheless, this group, given its minuscule size, way well be an idiosyncrasy of the training set and never materialize out-of-sample, which is a major risk.[4]

In the remainder of the paper we try to make the case that filters help the decision maker in the two following ways illustrated above:

- by picking more reliable splitting variables;
- and by splitting more to the center of distributions, thereby creating moderate, more robust, splits and reducing overfitting.

The next section lays the ground of our theoretical framework.

3 Location of splits: univariate analysis

Sections 3 and 4 provide details on how regression trees work following their sequential construction, that is, starting with a micro view on where splits are located at the predictor level (Section 3) to a more macro (multivariate) analysis on how features compete to achieve the highest gain (Section 4).

3.1 Theoretical setting

Our framework fits closely to the way regression trees are built at the node level, i.e., looking at feature-level relationships first and then picking the most relevant predictor to perform the split. We thus start by introducing a simple univariate model[5] for the data generating process over individual features:

$$Y_i = g(X_i) + E_i, \tag{1}$$

where the error terms (or perturbations) and the X_i are i.i.d. sequences of random variables with finite variances. The E_i are centered with density f_E and finite variance σ_E^2. The X_i are also assumed to have an absolutely continuous distribution with density f_X and we moreover impose that the corresponding support is a simple interval, bounded or unbounded. Without much loss of generality, we assume that all marginal

[4] In this particular example, allocating to highly volatile stocks resembles a lottery bet and is as such very risky indeed.

[5] The multivariate analysis involving *all* features (and how they compete) will come later in the paper, from Section 5.2 onwards.

cumulative distribution functions (c.d.f.) are strictly increasing. For a given variable X, the corresponding c.d.f. will be denoted with F_X. We write $f(\epsilon, x)$ for the joint (continuous) density of the couple (E, X). We recall the definition of the conditional density $f(\epsilon|x) = f(\epsilon, x)/f_x(x)$ and we finally impose

$$\int_{\mathbb{R}} \epsilon f(\epsilon|x) d\epsilon = 0 \tag{2}$$

for all x in the support of X: not only are the errors centered unconditionally, but also conditionally on X. This assumption is rather commonplace in the modelling literature and will be required to dismiss the correlation term in the computation of the variance of Y. In line with the abundant nomenclature in the field, we will refer to Y as the output, label (even if it is a number) or the dependent variable, and to X as the input, feature, independent/exogenous variable, or predictor.

Equation (1) is not too restrictive per se. First, a univariate representation matches the way trees are traditionally greedily grown: the procedure starts by looking at splits at the individual variable level and then chooses which variable makes the more sense (i.e., augments the fit the most). Our representation implies that we consider splits individually, i.e. that Equation (1) holds for each node and each exogenous variable. Second, the condition we impose on g is simply that it be continuous on the support of the X_i, with $\int_{\mathbb{R}} g(x)f(x)dx < \infty$. This assumption makes sense when both the output Y and the variable X have finite supports, which is the case in practice. Finally, our requirement on errors is a zero conditional and unconditional mean, which is arguably a loose constraint as we do not impose a particular distribution.

In the paper, we will consider two types of forms for g. The first form, which we call the natural form is when $g(x) = \mathbb{E}[Y|X = x]$ and for which (2) is not binding. This specification is sometimes called the regression function (see Section 2.4 in Hastie, Tibshirani, and Friedman (2009)) in the statistical learning literature. The second type of forms encompasses parametric families and will be useful to study outcomes in particular cases.

We follow the standard literature on regression trees, as exposed in Breiman, Friedman, Stone, and Olshen (1984) or in Chapter 9 of Hastie et al. (2009). Given a sample of (Y_i,X_i) of size N, a regression tree seeks the splitting point as $\underset{c}{\operatorname{argmin}}\, V_N(c)$ with

$$V_N(c) = \sum_{X_i<c} \Big(g(X_i) + E_i - m_N^-(c)\Big)^2 + \sum_{X_i>c} \Big(g(X_i) + E_i - m_N^+(c)\Big)^2, \tag{3}$$

where $m_N^-(c) = \frac{1}{\#\{i,X_i<c\}} \sum_{X_i<c}(g(X_i) + E_i)$ and $m_N^+(c) = \frac{1}{\#\{i,X_i>c\}} \sum_{X_i>c}(g(X_i) + E_i)$ are the average values of Y, conditional on X being smaller or larger than c. The cardinal function $\#\{\cdot\}$ counts the number of occurrences of its argument (the size of the set). For tractability purposes, we will work in this paper with the limiting case $N \to \infty$ and consider the 'continuous' version of the splitting criterion:

$$V(c) = \mathbb{E}\left[(g(X) + E - m^-(c))^2 \mathbf{1}_{\{X<c\}}\right] + \mathbb{E}\left[(g(X) + E - m^+(c))^2 \mathbf{1}_{\{X>c\}}\right], \tag{4}$$

with $m^-(c) = \mathbb{E}[g(X)\mathbf{1}_{\{X<c\}}]/\mathbb{P}[X < c]$ and $m^+(c) = \mathbb{E}[g(X)\mathbf{1}_{\{X>c\}}]/\mathbb{P}[X > c]$. Finally, we quantify the gain obtained by the split by computing the difference of dispersions before and after the split:

$$G(c) = \mathbb{V}[Y] - V(c) = \mathbb{E}[(g(X) + E - \mathbb{E}[g(X)])^2] - V(c). \tag{5}$$

We underline that switching to a continuous setting can easily be justified when working on large datasets. The amount of financial data available nowadays is a genuine fertile ground: in the applications of Section 6, the training samples include dozens of thousands of lines. Under these conditions, the continuous versions of splits and gains will be good approximations of their discrete counterparts.

3.2 Properties of splits

The purpose of this subsection is to theoretically characterize the location of the optimal splits obtained by minimising (4). We start with a general result and then impose some assumptions that allow to refine the properties of the splits.

Proposition 1 *A point c^* is an optimal split if it satisfies the following equality*

$$\left(\int_{-\infty}^{c^*} f_X(y)dy\right)^{-1} \int_{-\infty}^{c^*} g(y)f_X(y)dy + \left(\int_{c^*}^{\infty} f_X(y)dy\right)^{-1} \int_{c^*}^{\infty} g(y)f_X(y)dy = 2g(c^*). \tag{6}$$

The proof of the proposition is detailed in Appendix A.1. The most remarkable feature of this result is that the condition does not depend on the error terms (because of the assumption of Equation (2)). The equation in the proposition directly leads to the following corollary.

Corollary 1 *The optimal splits have the following properties.*

- ***Property 1****. If c^* is an optimal split associated to g, then it is also an optimal split associated to $x \mapsto ag(x) + b$ for $a, b \neq 0$.*
- ***Property 2****. Assume that f_X is symmetric around its mean and median, μ. If the function $x \mapsto g(x + \mu) - g(\mu)$ is odd, then μ is the (only) optimal split.*
- ***Special case****. If $g(x) = ax + b$ with $a \neq 0$, the median is the optimal split if and only if it is equal to the mean.*

The first point is immediate. The proof of the second point relies on the fact that, at the median, the inverse integrals in (6) are both equal to 1/2. Moreover, we have, on the l.h.s. of (6),

$$\begin{aligned} & 2\left(\int_{-\infty}^{\mu} g(y)f(y)dy + \int_{\mu}^{\infty} g(y)f(y)dy\right) \\ = & \, 2\left(\int_{-\infty}^{0} (g(y+\mu) - g(\mu) + g(\mu))f(y)dy + \int_{0}^{\infty} (g(y+\mu) - g(\mu) + g(\mu))f(y)dy\right) \\ = & \, 2g(\mu). \end{aligned}$$

The proof of the last point follows the same lines.

In order to go any further, we make strong assumptions both on the distribution of the X_i and on g. As is often done in practice, e.g. for scaling purposes, we transform the points X_i by applying their empirical c.d.f.[6] Hence, their distribution converges to the continuous uniform law over the unit interval. Equation (6) translates into $(c^*)^{-1}\int_0^{c^*} g(y)dy + (1-c^*)^{-1}\int_{c^*}^1 g(y)dy = 2g(c^*)$. After basic manipulations, this yields, for $c^* \in (0,1)$:

$$h(c^*) = (1-2c^*)\int_0^{c^*} g(y)dy + c^*\int_0^1 g(y)dy - 2c^*(1-c^*)g(c^*) = 0 \tag{7}$$

The second type of assumption that we impose relates to the function g. We detail them in Condition 1. While our results hold in a slightly more general context, these restrictions help lighten the proofs in the Appendix.

Condition 1 *The function g satisfies the following set of restrictions:*

1. *g is bounded with bounded continuous derivative;*
2. *$g(0) \neq g(1)$ and $\int_0^1 g(y)dy \in (\min\{g(0), g(1)\}, \max\{g(0), g(1)\})$.*

In practice, it is not unusual to work with bounded output values. The second point imposes that g does not spend too much time below $\min\{g(0), g(1)\}$ or above $\max\{g(0), g(1)\}$. In addition, $g(0) \neq g(1)$ indicates that the average output value differs at least slightly when the input value is very small versus very large. This point will be verified in Section 6.2, though it is essentially technical.

Proposition 2 *Under Condition 1, if the X_i are uniformly distributed on [0,1], then h defined in (7) has at least one root. One of these roots is an optimal split and it is located between the smallest zero (inside (0,1)) and largest zero (inside (0,1)) of h'.*

The proof of the proposition is located in Appendix A.2. In all generality, it is hard to provide upper and lower bounds for the split location because the zeroes of h' are hard to locate analytically.

[6] It is generally admitted that monotonous transformations of predictive variables do not substantially alter the structure of the tree, or, more generally, of the predictions of other ML engines. We resort to this technique for analytical tractability, but we highlight in all transparency that it can have non-negligible effects, as is shown in Galili and Meilijson (2016). Furthermore, we recall that normalisations and scaling procedures are common practice in machine learning: it is known that neural networks perform better when all variables have the same scales: either they are located inside a bounded interval (usually $[-1, 1]$ or $[0, 1]$), or they are standardized to reach unit variance. Finally, it can also be argued that the prior transformation of data points could be directly embedded in the definition of the generator g. In any case, such scaling procedures are now commonplace in recent in asset pricing theory contributions: we refer to Kelly, Pruitt, and Su (2019) and Koijen and Yogo (2019) to name but a few.

3.3 Parametric families

In this section, we investigate the location of splits for simple parametric generators. As we will show in the empirical section of the paper, these forms are actually quite relevant in practice in spite of their apparent simplicity. Moreover, we work under the assumptions of Proposition 2. Critically, we impose a uniform distribution for the X_i, which is done (again) under mild loss of generality.

First, the simplest form for g is linear: $g(x) = ax + b$. This can be a good approximation as long a the relationship between Y and X is monotonous but neither too convex, nor too concave. Condition (7) simplifies to

$$ac^*(2c^* - 1)(c^* - 1) = 0.$$

Clearly Condition 1 holds when $a \neq 0$ and then the splitting point is $c^* = 1/2$ because we exclude the degenerate roots $c^* = 0$ and $c^* = 1$. This is but a special case of Corollary 1.

The next logical choice is the quadratic form. It can be useful when the relationship between X and Y is monotonous up to a certain point, and then switches to the opposite direction. The general case is uselessly tedious and we discuss the elementary form $g(x) = (x - b)^2$.[7] The parameter b controls the degree of asymmetry of the function over [0,1]. Evidently, $b = 1/2$ violates the second point of Condition 1. More precisely, the integral $\int_0^1 (y-b)^2 dy = 1/3 - b + b^2 \in (b^2, (1-b)^2)$ only if $b \notin (1/3, 2/3)$. Plainly, the first point of Condition 1 holds. The function h is a fourth order polynomial, $h(c) = \frac{1}{3}c(1-c)(1-3b+(1+6b)c-4c^2)$, with roots equal to zero, one, and

$$c^{\pm}(b) = \frac{1 + 6b \pm \sqrt{17 - 36b + 36b^2}}{8}. \tag{9}$$

In Figure 2 below, we plot the functions $c^-(b)$ and $c^+(b)$. The two vertical lines delimit the zone that violates the second point of Condition 1. Moreover, the parts of the curves that lie outside [0,1] are shown in dotted lines because a split cannot occur outside the unit interval. Hence, the only remaining points are the ones on the thick coloured lines. They all lie around the middle point, inside the [1/4,3/4] interval and they converge to 1/2 when $|b|$ increases to infinity.

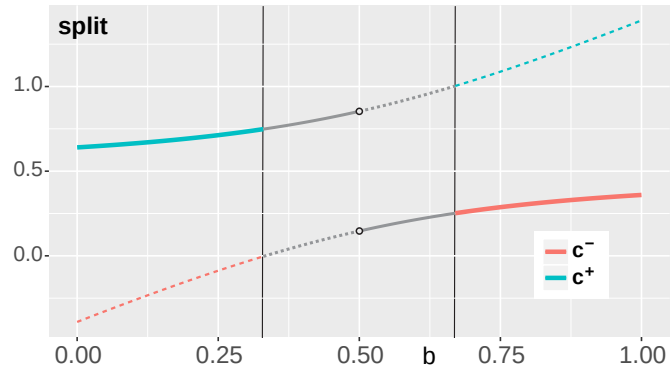

Fig. 2 Split location for quadratic g.

For the sake of completeness, we discuss the splits that occur inside the grey region of Figure 2. Those are the cases for which g is less asymmetric inside the unit interval. The problem in this case is that the two roots lie in the unit interval and both correspond to local minima of the objective function. As it turns out, the *global* minimum on [0,1] is reached by the root closest to 1/2, which is why the other values are shown in dotted grey in Figure 2. The limiting values are obtained for $b = 1/2$ and are equal to split locations of

[7] This is justified by Property 1 of Corollary 1. In the general case, $g(x) = a_2x^2 + a_1x + a_0$ with $a_2 \neq 0$, the function h in (7) is a polynomial of degree four and thus has four roots: 0, 1 and

$$x^{\pm} = \frac{a_2 - 3a_1 \pm \sqrt{17a_2 + 18a_1a_2 + 9a_1}}{8a_2}, \tag{8}$$

where the term inside the square root is positive over the $\mathbb{R}^2$ plane. Notably, a_0 plays not role in the splitting location, which makes sense. For the sake of simplicity, we choose to reduce the study to a one-parameter family.

$(4 - 2\sqrt{2})/8 \approx 0.146$ and $(4 + 2\sqrt{2})/8 \approx 0.854$. These two values correspond to the small black circles in the middle of the graph and highlight the discontinuity of the split's location at the point $b = 1/2$. At this point, it theoretically is impossible to decide which split to choose. In practice, a minuscule difference in the data will sway the decision in one or the other direction. We provide more details and insights about the quadratic form in Appendix B.

Finally, we briefly analyze the exponential case $g(x) = e^{ax}$, $a \in \mathbb{R}$, and the logarithmic form $g(x) = \log(x + b)$, $b > 0$, which both satisfy all points of Condition 1. There are no simple analytical forms for the zeros of h, but the latter can be calculated and plotted. The expressions are the following

$g(x)$	$h(c)$
e^{ax}	$a^{-1}(c - 1 + ce^{a} + (1 - 2c - 2ac + 2ac^{2})e^{ac})$
$\log(x + b)$	$-2c(1 - c) - b(1 - c)\log(b) + c(1 + b)\log(1 + b) + (b - c - 2bc)\log(b + c)$

In Figure 3, we plot these two functions for various values of their respective parameters. We observe monotonous patterns in both specifications: the roots of the functions increase in a for the exponential form and they also increase in b for the logarithmic form. When g is exponential, the roots can span the whole unit interval, as a varies from $-\infty$ to $+\infty$. When g is logarithmic, the roots are bounded below by the solution to $-2(1-x) = \log(x)$, which is $-2^{-1}W(-2/e^{2}) \approx 0.203$, where W is the Lambert (or product-log) function. This was obtained by letting b go to zero in the expression of h. When b increases to infinity, the second order Taylor expansion of $\log(1 + x)$ implies that h converges to $(c - 3c^{2} + 2c^{3})/(2b)$, which has its root at $c = 1/2$.[8] A logarithmic form thus imposes a strong restriction on the split: it must lie within (0.2,0.5).

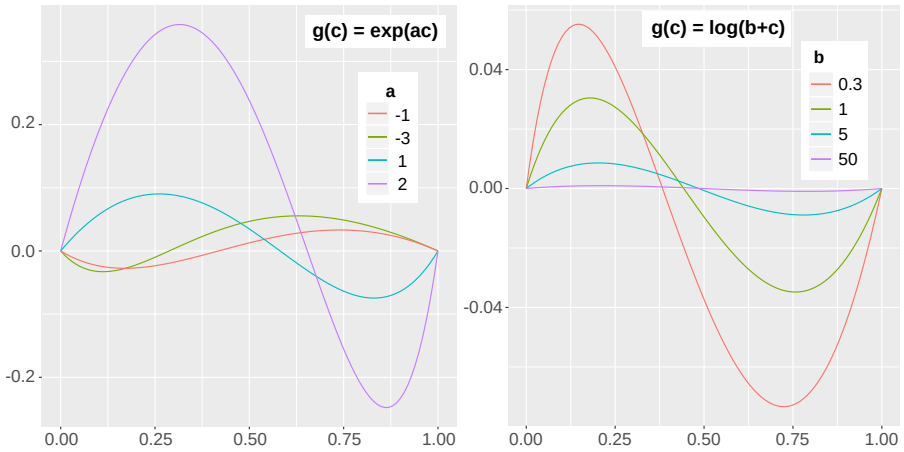

Fig. 3 h **function for exponential (left) and logarithmic (right)** g.

Overall, our results outline that optimal splits often occur in the bulk of the distribution, e.g., within the interquartile range. Counter-examples include exponential forms with large absolute arguments, and, to a lesser extent, quadratic generators when they are very symmetric.

4 Gains

In this section, we start by investigating the gains that are achieved at the optimal split for any given variable. We use notations for the initial split at the root of the tree, but the reasoning applies at any splitting level. As we did in the previous section, we state a general result first and then focus on particular cases.

[8] The full details of the derivation are available upon request.

4.1 General result

We start by providing a general result with no assumption on the generator function g nor on the distribution of the variables.

Proposition 3 *The following statements hold true.*

1. *All other things equal, the optimal gain is independent from σ_E^2.*
2. *At any splitting point c, the gain is equal to*

$$G(c) = \frac{\left(\mathbb{P}[X > c]\mathbb{E}[g(X)\mathbf{1}_{\{X<c\}}] - \mathbb{P}[X < c]\mathbb{E}[g(X)\mathbf{1}_{\{X>c\}}]\right)^2}{\mathbb{P}[X < c]\mathbb{P}[X > c]} \tag{10}$$

$$= \mathbb{P}[X < c]\mathbb{P}[X > c]\left(\mathbb{E}[g(X)|X < c] - \mathbb{E}[g(X)|X > c]\right)^2 \tag{11}$$

The proof of the proposition is located in Appendix A.3. The first point will never be useful in practice because for a given y, altering σ_E^2 implies a change in g as well (and vice-versa). The second point in the proposition shows that the gain is an increasing function of the distance between the conditional means $\mathbb{E}[g(X)|X < c^*]$ and $\mathbb{E}[g(X)|X > c^*]$. Intuitively, this makes sense: the split will be more effective if the average value of $g(X)$ is very different for values of X on both sides of c^*.

4.2 Linear versus quadratic form

In the remainder of this section, we assume that X has uniform distribution on $[0, 1]$, so that $\mathbb{V}[aX] = a^2/12$. We start with the simple case $g(x) = ax + b$. As seen above, the optimal split is located at $c = 1/2$ and the gain is thus

$$G_{\text{linear}}(a) = a^2/12 - \int_0^{1/2}(x - 1/4)^2 dx - \int_{1/2}^1 (x - 3/4)^2 dx = a^2/12 - a^2/48 = a^2/16. \tag{12}$$

The more pronounced the impact of X (i.e., the larger the magnitude of $|a|$), the greater the gain.

We then turn to the quadratic case, namely $g(x) = (x - b)^2$. The computations in this case are much more involved. The initial variance is

$$\int_0^1 ((x - b)^2 - (1/3 - b + b^2))^2 dx + \sigma_E^2 = (15b^2 - 15b + 4)/45 + \sigma_E^2.$$

The expression of the post-split dispersion function V_{quad} is given in Equation (37) in Appendix B and has a long polynomial form. In order to obtain the optimal gain, it is required to apply (i.e., compose) the optimal splitting value obtained in (9). We denote the corresponding function $G_{\text{quad}}(b)$. Its expression is too long and complicated to analyze rigorously; we simply write:

$$G_{\text{quad}}(b) = \begin{cases} (15b^2 - 15b + 4)/45 + \sigma_E^2 - V_{quad}(c^+(b)) & \text{if } \ b \leq 1/2 \\ (15b^2 - 15b + 4)/45 + \sigma_E^2 - V_{quad}(c^-(b)) & \text{if } \ b \geq 1/2 \end{cases}, \tag{13}$$

where $c^{\pm}$ is defined in (9). When $b = 1/2$, $V_{quad}(c^+(b)) = V_{quad}(c^-(b)) = \sigma_E^2 + \frac{11}{2880}$.

4.3 Comparison under iso-variance

Our purpose here is to compare the gains obtained in contests between linear generators and quadratic generators. Following our previous notations, we work with only one parameter for each family: a in the linear case and b in the quadratic case. Contests between two generators of the same family have obvious winners: in the linear case, the largest absolute slope $|a|$ wins. In the quadratic case defined above, the generator with b parameter that is farthest from $1/2$ will also win.

In the contest, we need to consider a constant initial variance: the relationship $y = g(x) + \epsilon$ will always imply $\mathbb{V}[y] = \mathbb{V}[g(x)] + \sigma_E^2$. In any contest between $x^{(1)}$ and $x^{(2)}$, it must hold that $\mathbb{V}[g(x^{(1)})] + \sigma_1^2 = \mathbb{V}[g(x^{(2)})] + \sigma_2^2$ and all specifications should respect this equality. Hence we impose some restrictions which we detail below.

In a linear versus quadratic contest, we oppose (a, σ_E^2) to (b, σ_E^2): the noise terms are the same. The initial condition is $a^2/12 + \sigma_E^2 = (15b^2 - 15b + 4)/45 + \sigma_E^2$, which is equivalent to $a(b) = \sqrt{12(15b^2 - 15b + 4)/45}$. We thus introduce, for a fixed σ_E^2,

$$G_{\text{linear}}(a(b)) = G_{\text{linear}}\left(\sqrt{12(15b^2 - 15b + 4)/45}\right) = \frac{15b^2 - 15b + 4}{60}. \tag{14}$$

Because of the symmetry of $(x - b)^2$ around $b = 1/2$ (we work in [0,1]), we will only consider the lower half $b \in [0, 1/2]$ (associated to c^+ defined in (9)).

In Figure 4, we plot the variation in gain when switching from the first alternative (linear) to the second (quadratic). We call this quantity $\Delta G(b) = G_{\text{linear}}(a(b)) - G_{\text{quad}}(b)$ where the two terms are defined in Equations (13) and (14). The plot shows a tipping point ($b \approx 0.383$) beyond which linear models lead to more gains. When the underlying impact of the quadratic generator is *too* symmetric and very much U-shaped, a linear model will be chosen as the splitting variable. When the quadratic effect is unbalanced (when b is far enough from 1/2 and the form is J-shaped), then it will lead to the highest gain, compared to the equivalent linear generator.

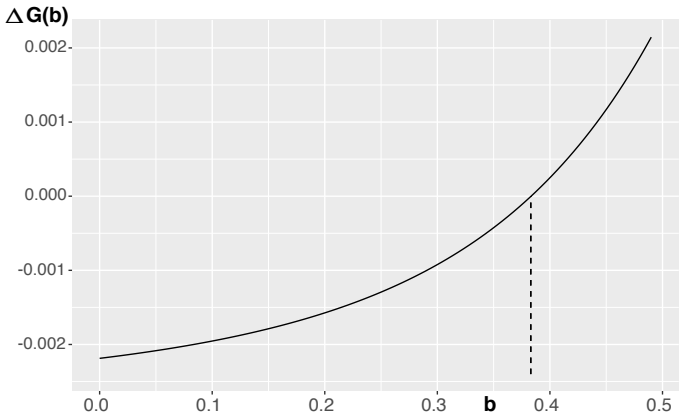

Fig. 4 Gain for stylized models. We plot the function $\Delta G(b)$: the difference in gain in the linear versus quadratic contests when both generators yield the same ex-ante variance. The dotted vertical line materializes the point where linear generators start to outperform quadratic generators.

5 The impact of filtering

We now turn to the core theme of the paper which is the impact of training the tree on a subsample of the data. We assume that the dataset has been filtered such that the Y_i inside the q and $1 - q$ quantiles have been removed (with $q < 1/2$), so that only extreme observations are kept (top q proportion and bottom q proportion).

This section, the core of the paper, is structured as follows:

1. In Section 5.1, we investigate the impact at the univariate scale. Our empirical results show that the filter is useful when the generator experiences large local variations to the left or to the right of the support of the predictor. Intuitively, a split occurs near this zone because the tree detects a change in the relationship between X and Y. Sometimes, this effect can be highly local (very close to 0 or to 1) and the filter shifts the split back to the 'middle', i.e., closer to 0.5.
2. In Section 5.2, we present a simple example that introduces the multivariate analysis: the competition between features.
3. In Section 5.3, our results show one key property which is that the filtering procedure favors the predictors that have the most monotonous impact on Y.
4. Section 5.4 is a robustness check of the above analysis.
5. Lastly, Section 5.5 discusses limitations and counter-examples.

5.1 Impact on split location

We start with a theoretical result when g is affine, i.e., $g(x) = ax + b$. Under some additional symmetry assumptions on f, we demonstrate below that removing a portion of the data that is centered around the median of Y will not affect the location of the split. Beyond this simple case, we cannot provide analytical results, if only because the splitting location is largely unknown even for the unfiltered sample. We will thus work with the stylized examples seen above - along with a few newcomers. In the following proposition, we impose $g(x) = ax + b$. Moreover, we need to define a notion of symmetry for f.

Definition 1 f is symmetric in both variables if $f(z, x) = f(-z, x)$ and if $f(z, \mu_X + x) = f(z, \mu_X - x)$, where μ_X is the median of X.

Since we work with continuous variables, we choose the following notation: instead of conditioning on $X = x$, we will condition on $X \in dx$. Notably, the expectation of Y, conditionally on $Y > F_Y^{-1}(1-q)$ or $Y < F_Y^{-1}(q)$ and conditionally on $X \in dx$, will be denoted with

$$\begin{aligned} \mathcal{E}_q(x) &= \mathbb{E}\left[Y \left| (X \in dx) \cap \left\{ (Y < F_Y^{-1}(q)) \cup (Y > F_Y^{-1}(1-q)) \right\} \right.\right] \\ &= \frac{\int_{y \notin \left(F_Y^{-1}(q), F_Y^{-1}(1-q)\right)} y f(y - ax - b, x) dy}{\int_{y \notin \left(F_Y^{-1}(q), F_Y^{-1}(1-q)\right)} f(y - ax - b, x) dy}. \end{aligned} \tag{15}$$

Proposition 4 *If $g(x) = ax + b$ and if f is symmetric in both variables, then,*

1. *the density of X conditional on $Y \notin (F_Y^{-1}(q), F_Y^{-1}(1-q))$ is symmetric around μ_X,*
2. *the conditional average defined above is such that $x \mapsto \mathcal{E}_q(\mu_X + x) - \mathcal{E}_q(\mu_X)$ is odd.*

The proof is located in Appendix A.4. The direct consequence of the proposition, by Corollary 1, is that truncating the training sample does not change the location of the split when g is affine.

The impact on gain is much harder to handle. For instance, let us assume independence between X and E and a uniform distribution for X. The simple computation of the variance of the truncated Y is not straightforward as it depends on the choice of of the distribution of E. There is no simplification in the computation of the integral. The expressions of total variation post-split are even more involved, which makes any intelligible analysis impossible. Consequently, beyond this theoretical result, we turn to numerical studies in the remainder of the paper.

The next logical step is to investigate the impact of filtering when generators are not linear. We proceed with simulations in which the error terms in (1) are independent from X and have a standard Gaussian distribution. One million data points are sampled and a tree with unique split is then built. We test the three generators defined in Section 3.3: quadratic ($g(x) = (x-b)^2$), exponential ($g(x) = e^{-bx}$) and logarithmic ($g(x) = \log(b + x)$). The results are shown in Figure 5.

First of all, these results confirm the theoretical forms obtained in Section 3.3:

- when the quadratic form has an argument (b) increasingly close to 0.5 (dark blue), the splits are further from 0.5;
- when the argument in the exponential has a large magnitude (light blue), the splits also shift away from the center;
- when the shape is logarithmic, the support of the split is narrow and included in $[0.2, 0.5]$.

The impact of the filter seems to depend on the parameters as well. In some cases, the filter seems to have a marginal (or simply no) impact because the points lie on the bisector of the quadrant. There are some clusters of points that nonetheless lie slightly above the bisector. They correspond to high parameter values for quadratic and exponential generators and to low parameter values for the logarithmic form. The quadratic case is not very interesting because when its parameter is large, it behaves like the linear generator and we see that the splits are close to 0.5. The exponential (convex decreasing) and logarithmic (concave increasing) forms are worth commenting. In both cases, the generator experiences a strong variation to the left of the origin and then progressively stagnates when x reaches one: the derivative of the generator progressively vanishes. This explains why the initial splits occur very much to the left of 0.5.

For the exponential generator, as shown in Section 3.3, the split can be arbitrarily close to zero if the parameter is negative enough. The impact of the filter is clear: it slightly shifts the split back towards 0.5. One graphical explanation for this change is located in Appendix C. We provide an empirical illustration of this effect in Section 6.

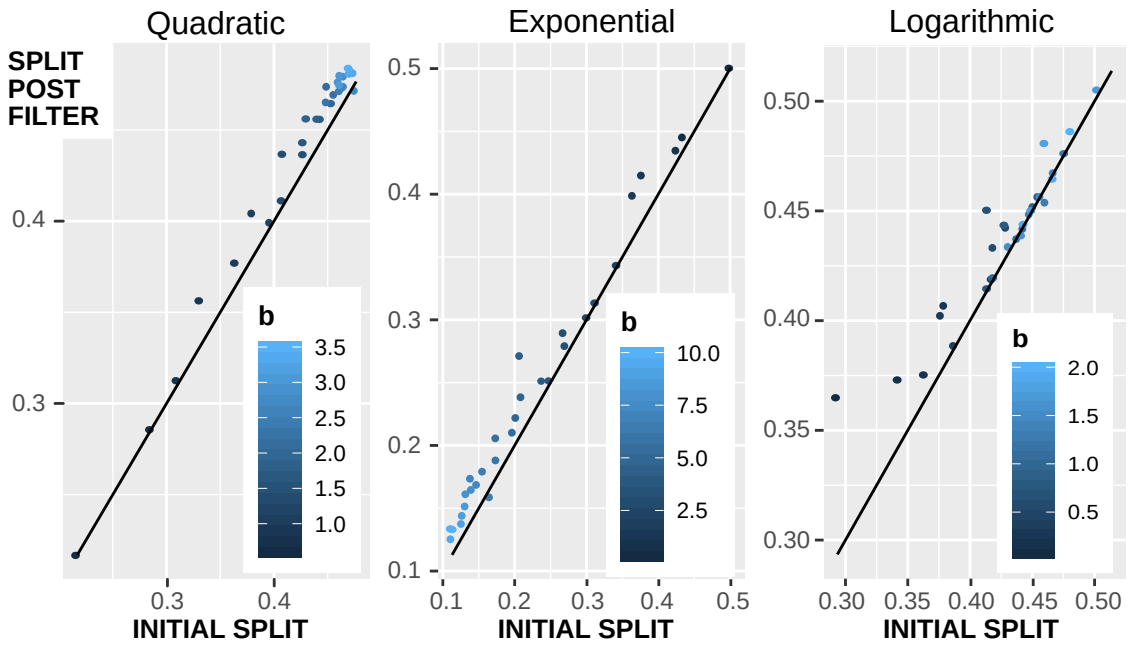

Fig. 5 Impact of filtering on split location. We display the split location after filtering as a function of the split location before the filter. The number of simulations is 10^6 and errors are independent and $\mathcal{N}(0,1)$ distributed. The filtering intensity is $q = 0.2$. The generators are quadratic ($g(x) = (x-b)^2$), exponential ($g(x) = e^{-bx}$) and logarithmic ($g(x) = \log(b+x)$). The black line is the identity function $f(x) = x$.

Overall, when the feature has a close to linear impact on the label, the filter does not change the splitting location. Nevertheless, when the generator is subject to strong local variations, the filter shifts the split closer to the middle of the distribution: it has a taming effect against extreme splitting locations.

5.2 Impact on gains: toy example for the multivariate case

Until now, our analysis focused on the impact of the filter on univariate relationships between individual features and the dependent variable. In the construction of the tree, once the optimal split is found for each feature, the final choice for the split is determined by a competition between all predictors and the one generating the highest gain wins. One natural (and deep) question thus pertains to whether or not the filter alters this competition (and if yes, in what ways).

The motivation of our whole study can be easily understood with a simple example. We consider the following model with two independent variables (for expositional clarity):

$$y = x_1/3 + \sin(10x_2)/4 + x_2/9 - 0.12 + \epsilon/10, \quad \epsilon \sim \mathcal{N}(0,1),$$

which is visually represented with sample points and their smoothed conditional averages in the left graph of Figure 6. If a simple tree is built on data generated by this model, the first split will be decided according to x_2 around a value of 0.65. The total gain divided by the number of samples is equal to 0.0069 for x_1 and 0.0077 for x_2. Now, if we filter the data with a threshold of $q = 0.2$, then the resulting tree is altered and the split is dictated by variable is x_1 around 0.5. The splitting variable is robust to changes in q: switching from $q = 0.05$ to $q = 0.25$ does not alter the outcome.

After the filter, the two clusters of the linear generator are comparable and contribute almost equally to the total dispersion. The two clusters of the drifted sine generator are very imbalanced as is illustrated in the right graph of Figure 6. The cluster on the right ($x_2 > 0.65$) is smaller and relatively homogeneous but the one one the left ($x_2 < 0.65$) is bigger and has a large dispersion in y because of the fluctuations of the generator.

Arguably, from an investor point of view, x_1 seems like the better, more robust, choice. Indeed, even though its effect is somewhat less pronounced, it is monotonous. Moreover, the right cluster of x_2 has a large mean in y, but the aggregate behavior of y for large values of x_2 has an inverted U-shape, which is troubling especially because the largest values of x_2 lead to negative average values of y. As such, this cluster could

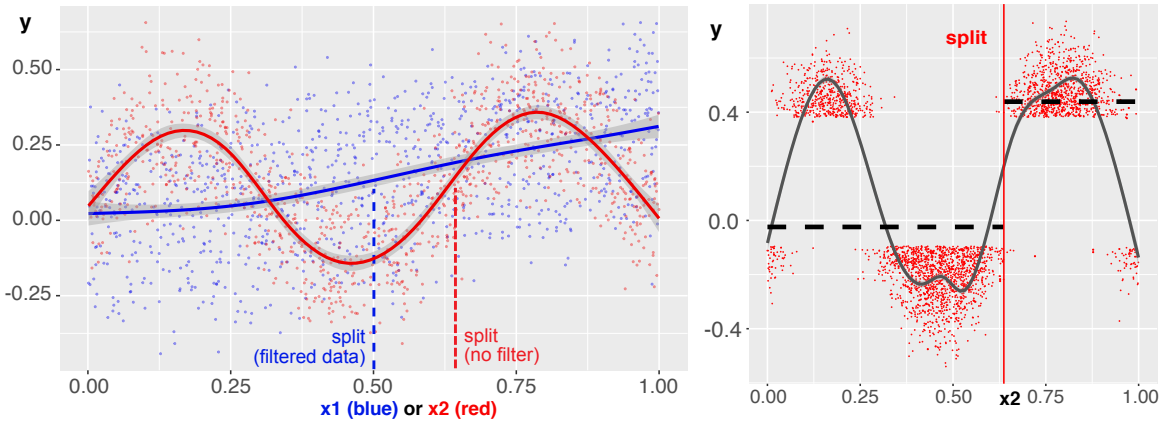

Fig. 6 Toy example: conditional averages and split locations. On the left graph, the conditional average of y is plotted for both variables. On the right graph, it is plotted for x_2 only after the 'average' values of y were removed ($q = 0.15$). The vertical lines mark the splitting point (blue for x_1 and red for x_2) and the dashed horizontal lines display the average value of y within each cluster.

be interpreted as a potential fluke which may fail to generalize out-of-sample. In contrast, it would require a major shift in the data generating process for the impact of x_1 to vanish or reverse.

5.3 Impact on gains: large scale simulation

In continuation of the comparisons of Section 4.3, we devise a simulation protocol to compare gains between generators after the data has been filtered. Crafting a protocol to evaluate gains depending on generators is not straightforward because in addition to generators, gains will also depend on the total variance of y and on the signal-to-noise ratio, which we define as $\sigma_E^{-2}\mathbb{V}[g(X)]$. Hence, across generators, these indicators should be comparable so that they are controlled for. We discuss these issues in more details in the next subsection.

The steps of the numerical study are the following:

1. Generate the data. For simplicity, a total of 10^6 points are simulated via Equation (1) assuming that X and E are independent and that E follows a scaled centered Gaussian distribution while X is uniformly distributed on the unit interval. In Equation (1), the variance of perturbations is equal to $\sigma_E^2 = 0.01$ and the variance of generator will lie in some interval (see Table 1). On average, the variance of the noisy term is four times larger than that of the generator;
2. Compute the gain of one split via (3), which we write g_j for generator j;
3. Perform the filtering: remove all observations such that Y_i lies in the quantile range $(q, 1-q)$;
4. Compute the gain of one split via (3) applied on the filtered data, which we write $g_j^{(q)}$.

We sum up the generators that we use for our study in Table 1. The parameter ranges are chosen so that the gains before (and, in fact, after) filtering are comparable across generators. For each family of g, we discretize the parameter space uniformly along 1,000 values inside the ranges given in Table 1. One exception is the polynomial family, for which the parameters are sampled randomly with the degree fixed arbitrarily to six: we generate the coefficient of each monomial for each degree from one to six. We restrict the study to 1,000 simulations because as we show below the patterns are clear and increasing the size would lead to the exact same conclusion, thereby adding little value.

In the left graph of Figure 7, we plot the generators for the highest, lowest and average value of the parameter range reported in Table 1. This shows the diversity of behaviors in our study. The polynomial family was generated randomly. Given the requirement on gain homogeneity, three families yield very close patterns: linear, logarithmic and exponential. Their impact on y is quasi-linear. One common property of all generators is their smoothness and we discuss this point further in Section 5.5 below.

For the sake of comparison, we plot *actual* generators from the dataset we use in Section 6. These generators are the smoothed conditional averages of the one month ahead return, computed as a function of predictors values.[9] They are depicted in Figure 8. Naturally, the curves are not as regular as those of Figure 7, but the shapes and overall patterns match quite well. There are generators that:

[9] More precisely, the calculation is that of the loess (locally estimated scatterplot smoothing) function.

Model	Generator	Parameter range	Std. dev. range
linear	$g(x) = cx$	$c \in [0.1, 0.3]$	$\hat{\sigma} \in [0.029, 0.087]$
quadratic	$g(x) = (x - c)^2$	$c \in [0.43, 0.59]$	$\hat{\sigma} \in [0.075, 0.091]$
logarithmic	$g(x) = \log(x + c)$	$c \in [7, 14]$	$\hat{\sigma} \in [0.021, 0.083]$
exponential	$g(x) = \exp(-cx)$	$c \in [0.1, 0.32]$	$\hat{\sigma} \in [0.028, 0.079]$
sine	$g(x) = \sin(cx)/c$	$c \in [3, 13]$	$\hat{\sigma} \in [0.050, 0.097]$
polynomial	$g(x) = \sum_j c_j x^j$ with $\sum_j c_j = 0$	random	$\hat{\sigma} \in [0.048, 0.083]$

Table 1 Families of generators. The ranges of standard deviations are computed over the 1,000 simulations. The condition $\sum_j c_j = 0$ for the polynomial generator increases the odds of non-monotonous behavior. The standard deviations and their ranges are computed *after* the simulations.

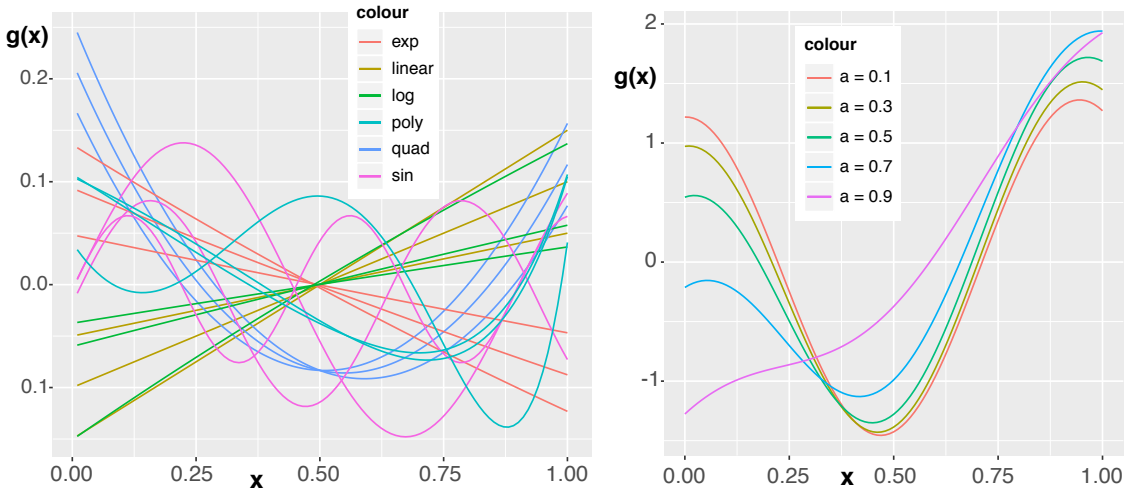

Fig. 7 Sample plots of simulated generators. In the left graph, the generators are those defined in Table 1 and used in Section 5.3, while in the right graphs are those used in Section 5.4.

– are almost constant and oscillate smoothly (CAPEX depreciation and Debt/Equity);
– have a close to linear impact (EBITDA, FCF/Assets);
– increase mildly at first and then have a convex boost to the right (BidAsk);
– oscillate while increasing slowly (Net Debt Yield) or decreasing slowly (Gross Profits);
– decrease and then increase (Buyback yield).

At a first order of magnitude, these behaviors can be replicated by the theoretical forms shown in Figure 7, which validates our initial choices.

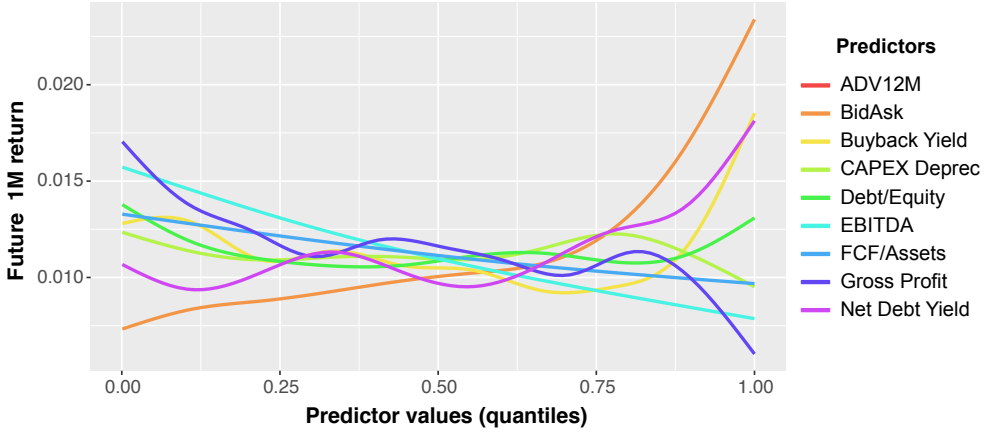

Fig. 8 Sample plots of estimated generators (conditional averages).

In Figure 9, we plot the gains obtained after the filter as a function of the gains before the filter: g_j is shown on the x-axis while g_j^q is represented on the y-axis. This figure indicates that up to a handful of exceptions discussed in Section 5.5, the monotone families (grouped in red) form an upper enveloppe that achieves the highest gain post-filter for a given level of initial gain. This is one key result of the paper and it is not impacted by changes in values of q: the same pattern emerges for $q = 0.1$ or $q = 0.25$, which are the typical values that can be used in practice.

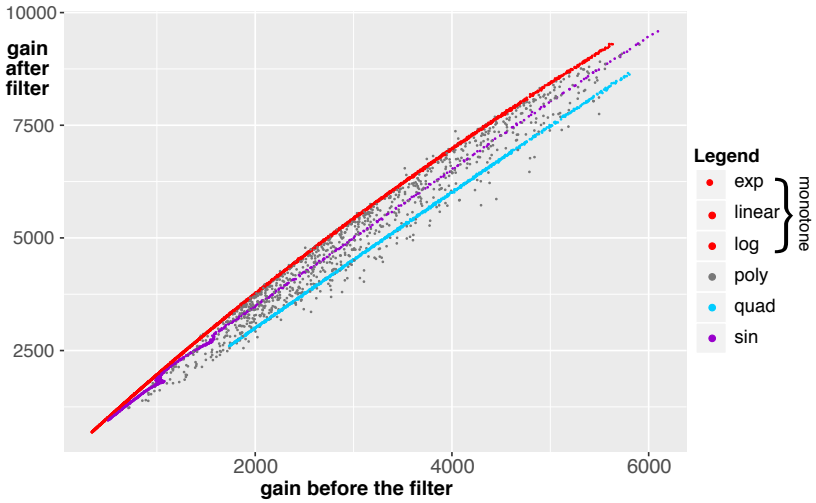

Fig. 9 Gain for stylized models before and after filtering. We plot the gains obtained by different generators. The x-axis is the original gain before the filter while the y-axis is the gain after filtering with $q = 0.15$. Colors mark the difference between the generators.

The implications of this study are summarized in Figure 10. For simplicity, let us consider only two explanatory variables, shown as black and red circles on the scheme. Before filtering, the one represented in black would be chosen as splitting criterion because it has the largest gain (x-axis). After the filter, the red one wins the contest (higher position on the y-axis). The filter has altered the competition between the variables and has favored the variable with monotonous impact on y. In short, given a large number of predictors with same ex-ante (pre-filter) gain, the filter will push the predictor that has the most monotone impact on the label.

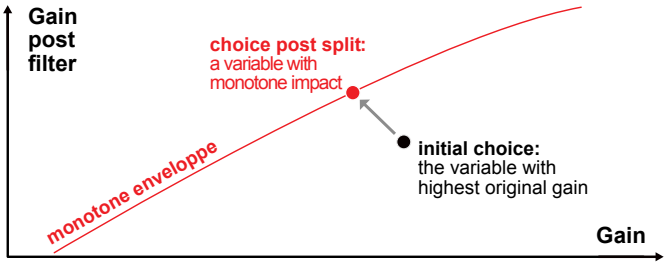

Fig. 10 Alteration of the contest via filtering.

5.4 Decomposing the contribution to gain

One point of criticism for the protocol is that we do not quantify the degree to which the generators are monotonous. This would be useful because it might help us show that the impact of filtering is indeed purely driven by monotonicity. One basic reason for this shortcoming is that there is no unambiguous way of quantifying monotonicity. Nonetheless, it is possible to increase the control over the set of generators.

Suppose we impose $g_\alpha(x) = \alpha g_1(x) + (1-\alpha) g_2(x)$, where g_1 is strictly increasing and g_2 is purely oscillatory. Then we can evaluate the impact of α on the gains realized after the filter.

A second important input is the quantity of noise inside the model, which we measure via σ_E^2. All other things equal, we expect a predictor associated to a smaller amount of noise to have superior explanatory power and hence to be favored as the splitting variable. In order to numerically confirm these intuitions, we consider the following setup:

$$y_i = g_\alpha(x_i) + E_i(\alpha), \quad \mathbb{V}[E_i(\alpha)] = \sigma_\alpha^2,$$
$$g_\alpha(x) = \frac{\alpha x + (1-\alpha)\cos(x/0.15) - m_\alpha}{s_\alpha},$$

where we scale the input of the trigonometric function to avoid unrealistic variations in the generators. Moreover, the generators are shifted by m_α and scaled by s_α to reach zero mean and unit variance. Sample plots of g_α are shown in the right panel of Figure 7 for values of α that are odd multiples of 0.1. By construction, as α increases, the pattern progressively shifts from oscillatory to increasing.

In a first phase, we reproduce the protocol of Section 5.3 when $\sigma_\alpha^2 = 4$ is constant across generators. The resulting gain analysis is provided in the left panel of Figure 11 below. The conclusion is unambiguous: both types of gains are strongly correlated and are driven by the mixing parameter a: when a is large and the generator has a 'more' monotonous shape, the gains are higher. One explanation for this is the signal to noise ratio. The variance implied by the monotonous generators is higher, thus for a fixed σ_α^2, there is more signal per unit of noise. We must thus correct for this bias by adapting the noise parameter for each generator.

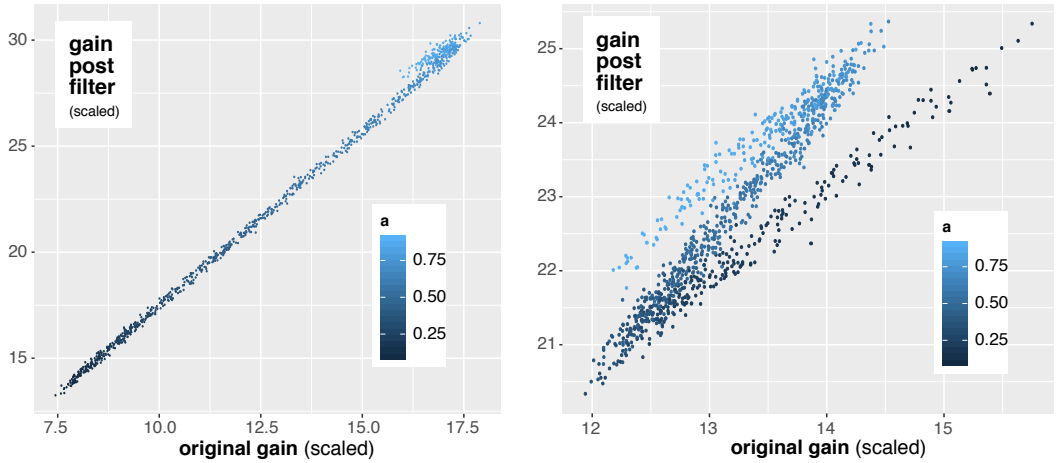

Fig. 11 Monotonicity and noise. In the left graph, the noise is the same in all simulations: $\sigma_\alpha^2 = 4$. In the right graph, the noise increases with monotonicity: $\sigma_\alpha^2 = 6a^{0.6}$. The average noise in both configurations is the same.

Hence, in our second batch of simulations, we set $\sigma_\alpha^2 = 6a^{0.6}$, so that the amount of noise increases with the intensity of the signal. This particular choice is the result of iterations: we tested different parametric forms and this one gave the best results, that is, the smallest interval for original gain (the noise adjustment is good enough that all models have comparable initial gains). The particular Z-shape in the right graph of Figure 11 comes from the initial V-shaped generator but the pattern is clear. For any given initial split, the gain after the filter is higher when a is bigger, that is when the generator is *more* monotonous (the linear, increasing, part has a higher coefficient in the mix). In a nutshell, this study confirms the findings of the preceding subsection and highlights the influence of monotonicity on gains before and after the filtering of the data.

5.5 Discussions on limitations, robustness, outliers and counterexamples

Limitations. By their empirical nature, our results are *qualitative*. While the construction of the quadratic and sine generators impose a non-monotonic behavior, we do not control the shape of the polynomials used

in the simulations. We can only observe that the monotonous generators are those for which the variation in gain is the largest after the filter.

This formal problem is impossible to solve from a theoretical point of view. Any reasonable non-degenerate generator g can be associated with an optimal split before (c_g) and after (c_g^q) some q-filtering has occurred. Both splits lead to gains $G(g, c_g)$ before the filter and $G(g, c_g^q)$ after the filter. Under natural boundedness and smoothness constraints, it is impossible to analytically characterize the function g that maximizes $G(g, c_g^q) - G(g, c_g)$ for a fixed level of initial gain $G(g, c_g)$:

$$g^* = \underset{g \in \mathcal{G}_\alpha}{\operatorname{argmax}} \left\{ G(g, c_g^q) \right\}, \text{ s.t.} \begin{cases} \int_0^1 g(x)^2 dx < \delta_1 & \text{boundedness} \\ |g(x) - g(y)| < \delta_2 |x - y| & \text{smoothness} \end{cases},$$

where $\mathcal{G}_\alpha$ is the set of all generators that yield an initial gain $G(g, c_g) = \alpha$. This task is infeasible for two reasons. First, the set $\mathcal{G}_\alpha$ is largely inaccessible. Second, the above optimization is a nested programme because c_g^q is itself the output of an initial minimization. The usual methods in optimization in function spaces (see Sasane (2016) for instance) do not apply in this case, which justifies the necessity to resort to simulations.

Robustness. Our results are not altered when changing the value of q. When q is in the reasonable range $[0.15, 0.25]$,[10] the same patterns are uncovered. Working with q smaller than 0.1 leads to degenerate cases in which suboptimal generators become competitive because the filter leads to special clusters. Moreover, discarding more than 80% of the original sample seems exaggerated: the loss in signal is too costly.

In Section 5.3, we work with a signal-to-noise ratio of four, on average. This indicator has a strong impact on our results. When it increases, all points progressively converge to the upper enveloppe and the filter loses its discriminatory power.

Outliers. Next, we also give more details on the points that lie above the red enveloppe in Figure 9. We ran simulations and kept the polynomial coefficients to isolate the cases that beat the linear/monotone generators in post-filter gains. As it turns out, the *overperforming* generators have shapes close to those of (shifted) odd monomials of high orders. The typical example is $g(x) = (x - 0.5)^{2k+1}$ for $k \in \mathbb{N}$. We show the case $k = 1$ in the left panel of Figure 12. The *gain in gain* is straightforward to explain for this example. Removing the average values will create two very homogeneous clusters shown in the red rectangles: one to the left (x close to zero with very negative values for y) and one to the right (x close to one with very positive values for y). The gain brought by the split is sizable because the two sets have very different y values, which means that the initial variance is large and strongly reduced following the split. The resulting gain is thus considerable. The fact that such functions would be chosen after a filtering procedure is not a problem at all: the clusters make sense and the relationship is highly monotonous.

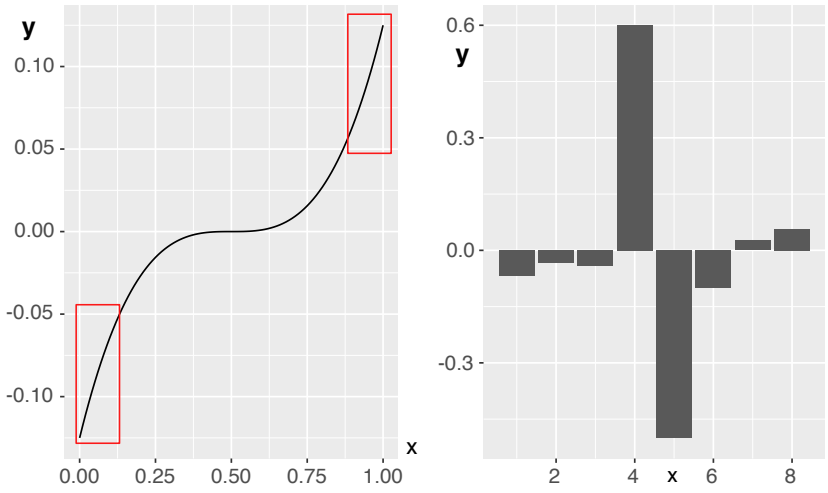

Fig. 12 Counterexamples: generators that surpass monotone/linear forms.

[10] This interval leads to retain between 30% and 50% of the original sample. These figures are those that we use in practice because they keep between one half and one third of the original data, which is reasonable.

Counterexamples. In full disclosure of our study, we finally mention another interesting family of generators: the piecewise constant functions. In unreported results, we also tested functions of this type which we also generated randomly. Just like for polynomial generators, a handful of cases were found to beat the linear/monotone threshold of Figure 9. In the right panel of Figure 12, we provide one example of such counterexample when the unit interval is divided into eight uniform segments. The generator has a constant value on each one of these segments. The explanation of the outstanding gains reached by this configuration is the same as above: removing the average values of y creates two natural clusters which consist essentially of the two big bars in the figure. In this case, the split would be more problematic because there is not clear intuition behind the relationship between x and y. The good news is that such configuration is highly unlikely for continuous variables in any reasonable dataset. As we show in the empirical section below, generators are usually very smooth, provided that the amount of data is sufficient. Under this condition, pathological cases like the one above will not be an issue in practice.

Final comment. Lastly, the changes in splitting variables will ultimately depend on the variables in the first place. If there are few variables, competition will be limited and the filter will have no impact, especially if one variable strongly dominates the others in terms of initial gains. As is shown in Figure 9, the two types of gains are very strongly correlated. The effect of filtering will be pronounced if the initial competition between feature is fierce, which usually requires a substantial number of predictors.

6 Financial application: trees in portfolio selection

In this section, we move forward to illustrations of our results on real data.

6.1 Introductory considerations

Before we provide data and methodological specifications, we discuss two crucial points: the nature of features in the exercise and the type of forecasting tool we will use.

When building a forecasting model based on machine learning methods, it is preferable to choose variables that have some explanatory power over future performance. Luckily, the financial literature has a large body of articles that reference such variables. In the recent years, these variables are often associated to pricing factors. The detection process is stable since the seminal contribution of Fama and French (1992): portfolios are built according to some sorted firm characteristic and average returns determine if the characteristic is impactful.[11] Usually, five or ten portfolios are constituted, based on quintiles or deciles of the characteristic, respectively. If the return of the first portfolio (lowest quintile or decile) is significantly different from that of the last portfolio (highest quintile or decile), then the characteristic is said to be a driver of returns in the cross-section of assets.[12]

Monotonicity is less often mentioned or tested as it is a condition which is both more technical to assess and harder to fulfil (see, e.g., Romano and Wolf (2013)). Nonetheless, it is clear that variables with erratic influence on returns are usually not retained in the portfolio construction process. In fact, researchers and practitioners who work with machine learning tools generally feed well known variables to their algorithms. These variables stem from the empirical asset pricing literature. A large overview of such features is provided in Green, Hand, and Zhang (2013), Harvey et al. (2016) and in the appendix of Linnainmaa and Roberts (2018). As we outline below, our choice of features follows these lines closely.

Once the features have been defined (and collected) comes the choice of the forecasting method. While we have only worked with simple trees up to now, they are not the best choice for portfolio construction. The first reason for this is their raw predictive ability which is outpaced by that of their enhanced versions: boosted trees and random forests. The second reason is purely technical and pertains to the portfolio construction process. The clustering form that emanates from the tree is highly unpractical. Let us imagine a tree with depth of five,[13] so that $2^5 = 32$ leaves are possible if we rule out non-binary trees. Each one of these leaves will have an average performance and any instance to be predicted will end up in one of those clusters. The immediate implication is that there are only 32 possible values for the forecasts. If there are many instances (i.e., stocks or assets), then many of them will share the same forecast, which is unsuitable

[11] For a review on asset pricing anomaly detection, we refer to Goyal (2012).

[12] The notion of 'significant' difference is subject to an ongoing debate among researchers. We refer to Harvey, Liu, and Zhu (2016) and Harvey (2017) for more debate on the subject.

[13] Trees usually have depths between 3 and 6.

when selecting assets based on these performance predictions. Indeed, if we fix the size of the target portfolio to 100,[14] it is unlikely that the combination of the best clusters will lead to exactly 100 assets. Random forests and boosted trees, because they combine many trees allow for unique predictions for each new instance. It it then easier to sort the instances and make homogeneous portfolios with user-specified sizes. For these reasons, we will resort to boosted trees for the allocation exercise.

Boosted trees have emerged as one of the leading machine learning algorithms both in ML competitions and in practical applications. They have been successfully used in Finance and we refer to Krauss et al. (2017), Guida and Coqueret (2018) and Gu et al. (2019) for details on that matter. Boosted trees are an ensemble methods: they aggregate individual learners. The key feature is that each new tree is built in such a way that it optimally improves the fit to the data. A detailed account of their properties is out of the scope of this paper and we refer to Chen and Guestrin (2016) for an exhaustive presentation of the exact algorithm. The construction of boosted trees is more complex compared to simple tree hence Section 6.4 will test if the filter can also deliver performance for more sophisticated learning models.

6.2 Data

Our proprietary dataset consists of 108 characteristics of 1182 firms the stocks of which are listed on US markets, plus one label which is the future one-year return.[15] For a given stock, this label is highly autocorrelated, which is a prerequisite because the features are also very autocorrelated.[16] The list of features is provided in the Appendix, Table 6. A large majority of features are accounting-based and proxy a large scope of documented anomalies: size (Fama and French (1992), Van Dijk (2011), Asness, Frazzini, Israel, Moskowitz, and Pedersen (2018)), value (Fama and French (1992), Pätäri and Leivo (2017)), profitability and investment (Fama and French (2015)) and quality (Asness, Frazzini, and Pedersen (2014)). Some features are price-based (past returns, relative strength index) and reflect momentum-like patterns (Novy-Marx (2012), Asness, Frazzini, Israel, and Moskowitz (2014)) or volatility-based (related to low-risk strategies as in Baker, Bradley, and Taliaferro (2014) and the references therein). While most firms are US-based, some of them are located in Canada or Mexico, hence, some features are quoted in the local currency (FX suffix in the table). The chronological range is December 2002 to May 2016 and the points are sampled at a monthly frequency.

Following the discussion in footnote 6, we engineer all features so that, each month, their value is equal to their level on the empirical cumulative distribution function of the current month. The features are thus quantile scores and, as such, all predictor values lie inside the unit interval and are uniformly distributed for a given month. We recall that normalisations are commonplace, both in portfolio selection (e.g., Brandt, Santa-Clara, and Valkanov (2009) or Ammann, Coqueret, and Schade (2016)) and in the asset pricing literature (Kelly et al. (2019), Koijen and Yogo (2019)).

Given the amount of data at our disposal, providing simple descriptive statistics is impractical. To briefly summarize the data, we provide in Figure 13 two illustrative plots. The first on the left shows the distribution of the label while the second one on the right plots the smoothed (locally estimated scatterplot smoothing (loess)) conditional average of the three most common features used in the literature (size, value and momentum). The first two of them follow the expected pattern: small firms and 'cheap' firms experience, on average, higher returns. Unfortunately, the momentum anomaly is reversed in our sample. One possible reason is that the aftermath of the 2008 financial crisis is a typical example of a momentum crash (Barroso and Santa-Clara (2015), Daniel and Moskowitz (2016)) during which the premium associated to past performance reverses. The end of 2002 also marks a low point for US equities and the following months could also qualify for a momentum crash. Thus, in our sample, the momentum premium is very much mitigated by losses *subsequent* to market collapses.

6.3 Simple trees

Given the data described above, we build two types of regression trees on annual subsets: first the trees are built on the whole dataset and in a second step, they are trained on the filtered data with $q = 0.2$. We

[14] It is common practice in the money management industry to fix the size of the portfolio policies.

[15] The data and the code can be accessed here: www.gcoqueret.com/tot.html

[16] In linear models, the problem of regressing monthly returns on autocorrelated predictors is well documented since the seminal work of Stambaugh (1999). To circumvent this problem, we resort to a dependent variable that behaves like the predictors. We hope to unveil patterns that will be long-lasting, i.e., that will hold out-of-sample.

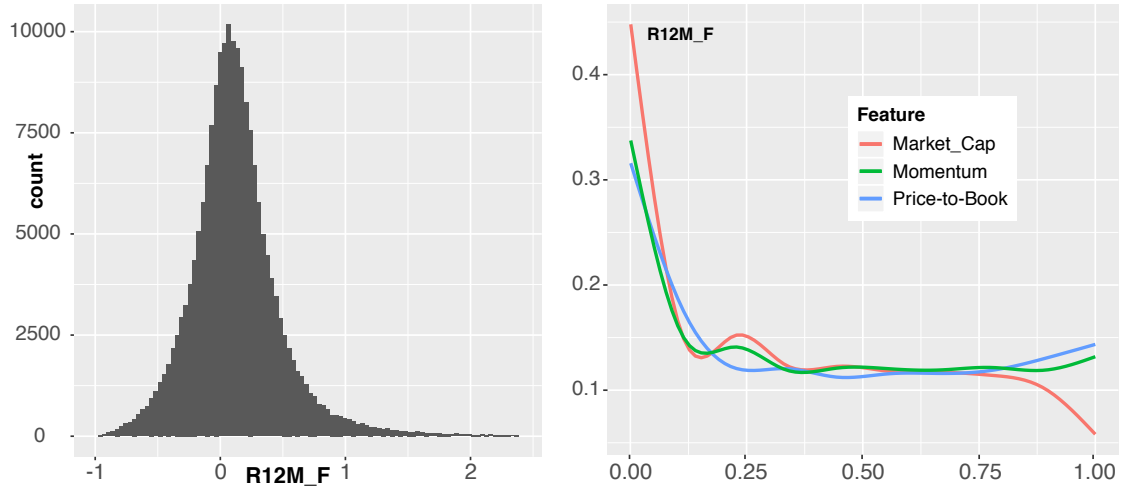

Fig. 13 Descriptive statistics: Histogram of the label (twelve month future return) and shapes of common generators (size, past annual performance and price-to-book ratio).

report the first splitting variable and the split location in Table 2 below. The first line corresponds to the two trees shown in the preliminary example in Figure 1.

Year	Initial splitting var.	Initial split location	Filtered splitting var.	Filtered split location
2002	VolXPostSTFX	0.939	MKTCAP	0.120
2003	MKTCAP	0.079	MKTCAP	0.164
2004	Gross_Profit	0.880	GP Margin	0.880
2005	Net Margin	0.366	Net Margin	0.396
2006	GP Margin	0.947	GP Margin	0.853
2007	ADV12M	0.071	ADV12M	0.073
2008	MKTCAP	0.006	MKTCAP	0.006
2009	R12M_USD	0.041	R12M_USD	0.047
2010	GP/TOA	0.176	GP/TOA	0.176
2011	MKTCAP	0.013	MKTCAP	0.026
2012	VolXPostSTFX	0.998	VolXPostLTFX	0.852
2013	OCF/TOA	0.011	Gross Profit Margin	0.040
2014	GP/TOA	0.495	GP/TOA	0.495
2015	BidAsk	0.928	BidAsk	0.928
2016	VolXPostLTFX	0.993	VolXPostLTFX	0.986

Table 2 First split information for simple trees. We provide the first splitting variable as well as the split location when training trees on annual data.

Out of the 15 years in the sample, four yield different splitting variables. This is an echo to Sections 5.2 and 5.3 in which we show that the filter may indeed change the choice of the splitting feature. Out of the 11 cases in which the feature remains the same, 5 have identical splitting location, which means that the filter has had no impact. In all remaining cases, the location of the split is closer to 0.5 (the middle point) when the tree is trained on the splitting data. This is a strong corroboration of the phenomenon unveiled in Figure 5 where it is shown that the filter produces splits that are closer to the center. In Table 2, the change can be minor (e.g., in 2005, from 0.366 to 0.396) or more pronounced as in 2012, from 0.99 to 0.85. These results confirm some theoretical insights from Section 5.

The generator associated to market capitalization in 2003 is close to that of the aggregate sample (see Figure 13) and is somewhat similar to that of Figure 17 in the Appendix (strongly decreasing close to zero and then stable). This very clearly explains the shift from a split at 7.9% before the filter to one at 16.4% after the filter.

Given model hyperparameters (learning rate, number of trees) **do**:
For dates $t = 1, \ldots, T$ **do**:
1. build 5 year training sample with no look ahead bias
2. perform filter if need be (depending on the case)
3. train model with given set of hyperparameters
4. build return predictions based on current time-t values of features

For portfolios $i = 1, \ldots, 6$ **do**: (*loop on performance ranks*)
a) select assets belonging to return group i (based on prediction quantiles);
b) store average return of the corresponding portfolio (equal weights).

Table 3 Steps of the backtest. In the secondary loop, the user ranks stocks according to their predicted values. In group 1, the 16.67% of stocks with lowest predictions are grouped, while group 6 encompasses the 16.67% of stocks with highest performance. Intermediate groups gather the intermediate levels of predictions.

6.4 Implications for portfolio choice: dynamic backtesting

We then proceed to a classical out-of-sample backtesting exercise. Each month, starting in January 2008 and ending in May 2016, we train the model on the previous five years of data. Given the current characteristics of firms, we proceed to the prediction of future performance and create portfolios based on forecasted this indicator. From the 1,182 stocks we constitute 6 portfolios of exactly 197 assets. This number is enough to ensure sufficient diversification within the portfolios. The stocks are sorted according to the forecast and grouped into the 6 portfolios of homogeneous predicted performance. This way, the first portfolio gathers all stocks with the worst predictions while the sixth portfolio groups those with the best.

The portfolios are then held for one month (after which the portfolio return is computed) and the procedure starts over the following month. Assets are equally-weighted inside the portfolios so that there is no further intermediate step between the prediction and the output which is the average return of the portfolios. We also provide the computation time of the backtesting. The computer we used was a 2018 13" Macbook Pro with a 3.1GHz Intel Core i5 with 16Go LPDDR3 RAM. At each step (each month), the model is trained on approximately 56,000 instances (full sample case) with all 108 predictors. The detailed steps are outline in Table 3.

Lastly, we test different parametrizations for the boosted trees. We only alter the usual hyperparameters (learning rate η and number of trees n) and consider three configurations: deep (100 trees, $\eta = 0.3$), intermediate (50 trees, $\eta = 0.5$) and light (25 trees, $\eta = 0.7$). Our results are collected in Table 4.

config.	q	(6)-(1) (t-stat)	CPU (s)	Portfolio rank / Average return					
				(1)	(2)	(3)	(4)	(5)	(6)
deep	0.1	2.457	782	0.007	0.008	0.008	0.008	0.009	0.015
deep	0.2	2.479	1408	0.006	0.007	0.008	0.008	0.010	0.016
deep	0.3	2.536	2054	0.007	0.006	0.008	0.009	0.009	0.016
deep	0.5	2.451	3400	0.007	0.006	0.008	0.009	0.010	0.015
intermediate	0.1	2.209	396	0.006	0.008	0.007	0.009	0.010	0.013
intermediate	0.2	2.133	726	0.008	0.007	0.007	0.008	0.009	0.015
intermediate	0.3	2.841	1109	0.006	0.007	0.009	0.009	0.009	0.015
intermediate	0.5	2.561	1683	0.007	0.007	0.008	0.009	0.009	0.015
light	0.1	2.105	222	0.007	0.008	0.008	0.009	0.010	0.013
light	0.2	2.209	383	0.007	0.008	0.008	0.007	0.010	0.015
light	0.3	2.590	543	0.007	0.007	0.008	0.009	0.009	0.015
light	0.5	2.859	841	0.007	0.008	0.008	0.008	0.009	0.015

Table 4 Backtesting metrics. We gather the key indicators of our out-of sample study. The allocation takes place between January 2008 and May 2016 and the training is performed on the most recent 5 years of data (rolling window). The configurations for the boosted tree algorithm are deep (100 trees, $\eta = 0.3$), intermediate (50 trees, $\eta = 0.5$) and light (25 trees, $\eta = 0.7$). The parameter q sets the filtering intensity: the proportion of retained occurrences is equal to $2q$; $q = 0.5$ corresponds to no filtering at all. CPU times are given in seconds and relate to the entire backtesting loop. Lastly, we performed a simple t-test on the difference of extreme portfolios (first minus sixth) and we provide the corresponding t-statistic in the third column.

The first conclusion from the table is that the algorithms do their jobs quite well: in all lines, the average returns are monotonous (or close to) with the rank of the portfolio so that the models generalize fairly well out-of-sample. Notably, portfolios 5 and 6 always have the highest average returns, but the monotonicity is less marked for portfolios 1 to 4. The important criterion is the degree to which the allocation is able to produce portfolios with sharply increasing average returns when the portfolio rank increases. As is customary in the asset pricing literature, we compute the difference in returns between portfolio 6 and portfolio 1 and report the associated t-statistic in the third column of Table 4. All values lie solidly above the 1.96 threshold (95% confidence level), so that the models do manage to discriminate between profitable assets and low-performing ones. Logically, the computation times increase with q (more training data requires more time to evaluate the optimal spits) and with the configuration (heavier configurations involve more calculations because more trees are grown).

The second conclusion is the core finding of the paper. The performance of the allocation is not much impacted by the filter. In fact, in some cases, the filter seems to slightly improve the predictions. For instance, in the deep configuration, the t-statistics of the filtered portfolios are all higher than that of the base-case portfolio. We nonetheless acknowledge a minor deterioration of the results for $q = 0.1$ in the less sophisticated configurations. All in all, discarding half or even two thirds of the training sample does not alter the outcome of the portfolio construction process. Nonetheless, this has a sizeable impact on computation times. Switching from $q = 0.5$ (full sample) to $q = 0.2$ (40% of the sample) divides the backtesting durations by a factor 2.5.

For the sake of completeness, we wish to quantify the loss in return that is suffered if the filter is performed the other way round, i.e., by keeping the bulk of returns so that the sample consists of all 'average' observations. Indeed, if there is no loss under the opposite filter, then our arguments vanish.[17] To assess the cost of this loss of information, we calculate the average return of long-short (best minus worst) strategy based on the prediction from the boosted trees. This evaluates the quality of the signal which is derived from the data. We report our results in Figure 14. We test different configurations and compare the filtering methods when retaining 20%, 40% and 60% of the original sample.

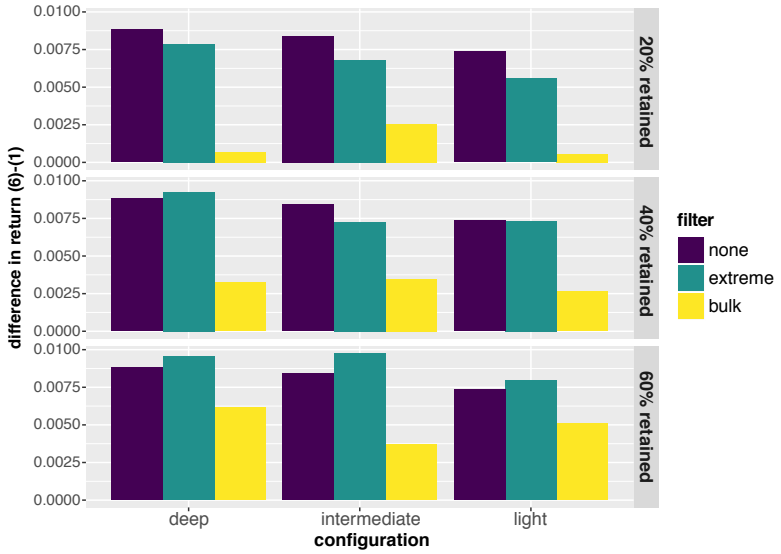

Fig. 14 Comparison of filtering methods. We compute the difference between the average return of best performing portfolio (6) and that of the least performing (1). The values are expressed at the original monthly frequency. The configurations are those defined in the original protocol. Each line of barplots corresponds to one filtering intensity: the first intensity is strong (keeping only 20% of training data) the third one is mild (retaining 60% of the data).

The striking pattern is the difference between the green and yellow rectangles: the models based on extreme observations yield portfolios that are much more profitable. The deficit of bulk-trained models compared to the other two cases is marked and distinctly highlights that the most valuable information is

[17] An important alternative is the *random* filter. Nonetheless, it is much more expensive empirically because the performance is strongly dependent on the random seed. Robust results require at least 100 iterations of each backtest. In (partial) unreported results, we document that the random filter is inferior to the extreme filter, especially if the filter is intense.

not located in the middle of the distribution of returns, but more probably in its tails. When only a small portion of data is retained (20%, top panel), the algorithm trained on *medium* returns fails to deliver any robust out-of-sample discrepancy between the top predictions and worst predictions.

When its intensity is not to strong (60% of the data is kept), the extreme filter improves the returns across all configurations. This is no longer true when the sample is strongly curtailed (20% of the data is retained) and in these cases, there is a clear loss. This underlines the need to choose a reasonable filtering intensity.

The strong patterns observed in Figure 14 are a forceful confirmation that all instances are not endowed with the same informational quality. Focusing on the right observations does not damage forecasting accuracy and reduces training times by a factor 1.5 to 5.

To conclude our analysis, we form long-only portfolios because they are the most widespread (apart from hedge funds, few institutions implement long-short strategies). In the process described in Table 3, the last part is replaced by the construction of only one portfolio, namely the one that retains the 60 stocks which have the highest predictions for future returns. The predictive engine is the *light* one described above, namely the one which aggregates only 25 trees. The portfolio values are initialized at one. Their evolution is plotted in Figure 15 and key performance indicators including risk-adjusted metrics are displayed in Table 5. As a benchmark, we use the equally-weighted (EW) portfolio of all stocks in the sample, which is a yardstick that is known to be hard to beat, as is shown by DeMiguel, Garlappi, and Uppal (2007).

First of all, the results are impressive and speak largely in favor of ML-based approaches, even though the latter do not help avoid major downturns because the equity space is highly correlated in the midst of financial crises. Higher risk (ex: larger drawdowns) do lead to higher rewards (average returns) in the long run. This makes sense: the aim of the models is not to reduce risk but to deliver returns, which they do. In fact, the gain in returns is not offset by the less favorable levels of volatility: the ML-based Sharpe ratios are well above that of the EW portfolio. The second takeaway is that the predictions relying on the filtered training samples do not perform worse than those based on the large datasets. This finding is our last yet forceful argument in favor of our filtering procedure.

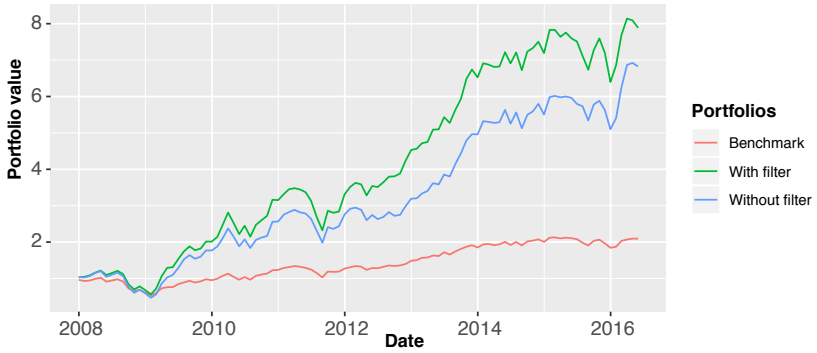

Fig. 15 Long-only portfolios. We plot portfolio values of three alternatives: the equally-weighted benchmark of all stocks, the ML-based allocation based on all instances, and the ML-based allocation based on the filtered instances. The backtest starts in January 2008.

	Equally weighted benchmark	ML-based (no filter)	ML-based (with filter)
Average return	0.0091	0.0239	**0.0250**
Volatility	**0.0603**	0.1016	0.0984
Sharpe ratio	0.1507	0.2352	**0.2542**
Maximum drawdown	**0.4740**	0.6078	0.5366

Table 5 Performance analytics. We gather standard key financial indicators. The first three quantities are expressed at the monthly frequency. The maximum drawdown is the maximum loss suffered by the portfolio, expressed as a percentage of the peak value. The most favorable figures are shown in bold font.

7 Conclusion

In this article, we analyze the impact of removing particular instances from the training sample when fitting regression trees. Our results show that the same level of performance can be achieved with two or three times less data as long as the filter focuses on extreme values of the dependent variable. If we posit that the signal that is extracted from a dataset is proportional to its size, then the filtered samples undoubtedly have a higher signal-to-noise ratio as long as at least one third of the original sample is preserved.

Our attempt to explain this evidence has led us towards two thought-provoking patterns. First, filtering can lead to splits that yield more balanced clusters. Second, filtering is favorable to variables that have a monotonous impact on the label. Both of these conclusions point towards a reduction in overfitting. Indeed, extreme splits that form micro-clusters are more likely to stem from idiosyncrasies from the training set that may vanish in the testing set. Hence taming extreme thresholds is a good safeguard against overfitting. Similarly, promoting features that have a monotonous impact on the dependent variable makes also great sense from an investor's standpoint. Indeed, it seems preferable to base portfolio decisions on variables that have a robust and coherent impact on the future performance rather than on predictors over which future returns fluctuate.

In their monograph on statistical learning, Hastie et al. (2009) give the computational cost of training trees: it is located between $N \log(N) p$ and $N^2 p$ where N is the number of instances and p the number of predictors. Being able to reduce the training size threefold can thus speed up the computations by a factor three to the very least and by a factor nine in the best case. These magnitudes are those we observed in our backtests. This can be very helpful when it comes to computationally greedy (and imperative) tasks such as grid or random search in hyperparameter optimization.

While all of our results revolve around tree-based methods, we believe that similar patterns can be obtained for other algorithms (neural networks in particular). If future research corroborates this conjecture, it will confirm that extreme observations carry much more signal compared to average sample points.

8 Acknowledgements

The authors thank an anonymous referee for his/her precious comments that have improved the clarity of the paper.

Appendices

A Proofs

A.1 Proof of Proposition 1

The expressions of the conditional means are

$$m^-(c) = \left(\int_{-\infty}^{c} f_X(y)dy\right)^{-1} \int_{-\infty}^{c} g(y) f_X(y)dy$$
$$m^+(c) = \left(\int_{c}^{\infty} f_X(y)dy\right)^{-1} \int_{c}^{\infty} g(y) f_X(y)dy$$

Likewise, the total dispersion defined in (4) is equal to

$$V(c) = \pi^-(c) + \pi^+(c), \tag{16}$$

where

$$\pi^-(c) = \int_{-\infty}^{c} \int_{\mathbb{R}} \Big(g(x) + z - m^-(c)\Big)^2 f(z,x)dxdz \tag{17}$$
$$\pi^+(c) = \int_{c}^{\infty} \int_{\mathbb{R}} \Big(g(x) + z - m^+(c)\Big)^2 f(z,x)dxdz \tag{18}$$

Obviously, we work on the asymptotic expressions for analytical tractability. In the sequel, we will drop the N scaling factor in (16) for notational simplicity. We re-arrange π^- and π^+ via the conditional density:

$$\begin{aligned}
\pi^-(c) &= \int_{-\infty}^{c} \int_{\mathbb{R}} \Big(g(x) + z - m^-(c)\Big)^2 f(z|x) f_X(x)dxdz \\
&= \int_{-\infty}^{c} \int_{\mathbb{R}} \Big[(g(x) - m^-(c))^2 + z^2 - 2z(g(x) - m^-(c))\Big] f(z|x) f_X(x)dxdz \\
&= \int_{-\infty}^{c} \int_{\mathbb{R}} \Big[(g(x) - m^-(c))^2 + z^2\Big] f(z,x)dxdz \\
\pi^+(c) &= \int_{c}^{\infty} \int_{\mathbb{R}} \Big[(g(x) - m^+(c))^2 + z^2\Big] f(z,x)dxdz
\end{aligned}$$

where in the third line we have used (2). By the Leibniz integral rule for differentiation, we have

$$\frac{\partial}{\partial c}\pi^-(c) = \int_{\mathbb{R}} \Big[(g(c) - m^-(c))^2 + z^2\Big] f(z,c)dz - 2\int_{-\infty}^{c} \int_{\mathbb{R}} \left(\frac{\partial}{\partial c} m^-(c)\right) (g(x) - m^-(c)) f(z,x)dxdz$$
$$\frac{\partial}{\partial c}\pi^+(c) = -\int_{\mathbb{R}} \Big[(g(c) - m^+(c))^2 + z^2\Big] f(z,c)dz - 2\int_{c}^{\infty} \int_{\mathbb{R}} \left(\frac{\partial}{\partial c} m^+(c)\right) (g(x) - m^+(c)) f(z,x)dxdz$$

with

$$\begin{aligned}
\frac{\partial}{\partial c} m^-(c) &= \left(\int_{-\infty}^{c} f_X(y)dy\right)^{-1} g(c) f_X(c) - f_X(c) \int_{-\infty}^{c} g(y) f_X(y)dy \left(\int_{-\infty}^{c} f_X(y)dy\right)^{-2} \\
&= \left(\int_{-\infty}^{c} f_X(y)dy\right)^{-1} f_X(c) \Big(g(c) - m^-(c)\Big), \qquad (19) \\
\frac{\partial}{\partial c} m^+(c) &= -\left(\int_{c}^{\infty} f_X(y)dy\right)^{-1} f_X(c) \Big(g(c) - m^+(c)\Big). \qquad (20)
\end{aligned}$$

Hence, after multiple simplifications the first order derivative satisfies

$$\begin{aligned}\frac{\partial}{\partial c}V(c) =&\frac{\partial}{\partial c}\pi^-(c) + \frac{\partial}{\partial c}\pi^+(c)\\ =&\int_{\mathbb{R}}\left[m^-(c)^2 - m^+(c)^2 - 2g(c)(m^-(c) - m^+(c))\right] f(z,c)dz\\ &- 2\left(\int_{-\infty}^{c} f_X(y)dy\right)^{-1} f_X(c)\left(g(c) - m^-(c)\right) \times \int_{-\infty}^{c}(g(x) - m^-(c))f_X(x)dx \qquad (21)\\ &+ 2\left(\int_{c}^{\infty} f_X(y)dy\right)^{-1} f_X(c)\left(g(c) - m^+(c)\right)] \times \int_{c}^{\infty}(g(x) - m^+(c))f_X(x)dx \qquad (22)\\ & \qquad (23)\end{aligned}$$

In (21) and (22), the integrals on the right are equal to zero, which leads to

$$\begin{aligned}\frac{\partial}{\partial c}V(c) =&f_X(c)\left[m^-(c)^2 - m^+(c)^2 - 2g(c)(m^-(c) - m^+(c))\right]\\ &f_X(c)\left[m^-(c) + m^+(c)^2 - 2g(c)\right](m^-(c) - m^+(c)), \qquad (24)\end{aligned}$$

and

$$\frac{\partial}{\partial c}V(c) = 0 \quad \Leftrightarrow \quad \begin{cases} m^-(c) + m^+(c) = 2g(c) \\ \quad \text{and / or} \\ m^-(c) - m^+(c) = 0. \end{cases} \qquad (25)$$

In the equivalence above, we have relied on $f_X(c) > 0$. Indeed, support of f_X is an interval and on this interval it is assumed that $f_X > 0$. The spit must occur at some point inside the support of f_X (i.e., the input data), hence $f_X(c) > 0$. Indeed, a split cannot by definition be located at a point to the left or to the right of *all* of the points X_i.

We discuss the second order condition below. The second order derivative is obtained by differentiating (24):

$$\begin{aligned}\frac{\partial^2}{\partial c^2}V(c) =&f_X'(c)\left[m^-(c) - m^+(c)\right]\left[m^-(c) + m^+(c) - 2g(c)\right] \qquad (26)\\ &+ 2f_X(c)\left(m^-(c)\frac{\partial}{\partial c}m^-(c) - m^+(c)\frac{\partial}{\partial c}m^+(c)\right) \qquad (27)\\ &- 2f_X(c)(g'(c)\left(m^-(c) - m^+(c)\right)) \qquad (28)\\ &- 2f_X(c)g(c)\left(\frac{\partial}{\partial c}m^-(c) - \frac{\partial}{\partial c}m^+(c)\right), \qquad (29)\end{aligned}$$

where we write f' for the derivative of f. When the first order condition is met, we have the three cases from Equation (25):

– **Case** $m^-(c) + m^+(c) = 2g(c)$ and $m^-(c) - m^+(c) \neq 0$: we replace $g(c)$ in (26) and (29)

$$\begin{aligned}\frac{\partial^2}{\partial c^2}V(c) =& - 2f_X(c)g'(c)\left(m^-(c) - m^+(c)\right) \qquad (30)\\ &+ f_X(c)[m^-(c) - m^+(c)]\left(\frac{\partial}{\partial c}m^-(c) + \frac{\partial}{\partial c}m^+(c)\right),\end{aligned}$$

which implies that $(m^-(c) - m^+(c))\left(\frac{\partial}{\partial c}m^-(c) + \frac{\partial}{\partial c}m^+(c) - 2g'(c)\right) \geq 0$ is required to achieve minimisation. Plugging $g(c) = (m^-(c) + m^+(c))/2$ in (19) and (20) further simplifies this condition to

$$\begin{aligned}&(m^-(c) - m^+(c))\left[f_X(c)\frac{m^+(c) - m^-(c)}{2}\left(\left(\int_{-\infty}^{c} f_X(y)dy\right)^{-1} + \left(\int_{c}^{\infty} f_X(y)dy\right)^{-1}\right) - 2g'(c)\right] \geq 0\\ \Leftrightarrow \quad &-f_X(c)\frac{(m^+(c) - m^-(c))^2}{2} - 2(m^-(c) - m^+(c))\left(\int_{-\infty}^{c} f_X(y)dy\right)\left(\int_{c}^{\infty} f_X(y)dy\right)g'(c) \geq 0\end{aligned}$$

In all generality it seems impossible to go further than this inequality but we discuss a special case in the subsequent proof.

– **Case** $m^-(c) - m^+(c) = 0$ and $2g(c) \neq (m^-(c) + m^+(c))$: lines (26) and (28) vanish so

$$\begin{aligned}\frac{\partial^2}{\partial c^2}V(c) &= 2f_X(c)\left(m^-(c)\frac{\partial}{\partial c}m^-(c) - m^+(c)\frac{\partial}{\partial c}m^+(c)\right) - 2f_X(c)g(c)\left(\frac{\partial}{\partial c}m^-(c) - \frac{\partial}{\partial c}m^+(c)\right)\\ &= 2f_X(c)(m^-(c) - g(c))\frac{\partial}{\partial c}m^-(c) - 2f_X(c)(m^+(c) - g(c))\frac{\partial}{\partial c}m^+(c)\\ &= -2[f_X(c)(m^-(c) - g(c))]^2\left(\int_{-\infty}^{c} f_X(y)dy\right)^{-1} - 2[f_X(c)(m^+(c) - g(c))]^2\left(\int_{c}^{\infty} f_X(y)dy\right)^{-1}\end{aligned}$$

where we have used expressions (19) and (20). This last line is always negative, thereby implying that the split is *maximising* the dispersion function V. It can thus be ruled out.

– **Case** $m^-(c) - m^+(c) = 0$ and $2g(c) = (m^-(c) + m^+(c))$: by (30), this leads to $\frac{\partial^2}{\partial c^2}V(c) = 0$, which is inconclusive. This case is clearly degenerate as it implies $g(c) = m^-(c) = -m^+(c)$ by adding or subtracting the two conditions. According to the second condition, all three terms must be equal to zero. We rule this case out.

A.2 Proof of Proposition 2

First, when $f_X(x) = 1_{\{x\in[0,1]\}}$, we have $m^-(c) - m^+(c) = \frac{\int_0^c g(y)dy - c\int_0^1 g(y)dy}{c(1-c)}$ and $m^-(c) + m^+(c) - 2g(c) = \frac{h(c)}{c(1-c)}$, where h is defined in (7). Hence, by (24),

$$\frac{\partial}{\partial c}V(c) = \frac{\int_0^c g(y)dy - c\int_0^1 g(y)dy}{c(1-c)} \times \frac{h(c)}{c(1-c)}. \tag{31}$$

In addition, g is bounded on $[0,1]$, thus after some simplifications,

$$\lim_{c\to 0^+}\frac{\partial}{\partial c}V(c) = -\left(g(0) - \int_0^1 g(y)dy\right)^2 \leq 0 \quad \text{and} \quad \lim_{c\to 1^-}\frac{\partial}{\partial c}V(c) = \left(g(1) - \int_0^1 g(y)dy\right)^2 \geq 0.$$

This implies that V starts by decreasing to the right of zero, and ends by increasing to the left of one. Because $h(0) = h(1) = 0$, it holds that $V(0) = V(1) = 0$. Hence, if there exists at least one root to $\frac{\partial}{\partial c}V(c) = 0$ in $(0,1)$, then one of them minimizes V over this interval.

Next, we list some properties of h defined in (7) because the solution to $\frac{\partial}{\partial c}V(c) = 0$ with $\frac{\partial^2}{\partial c^2}V(c) \geq 0$ reduces to that of $h(c) = 0$.[18] Plainly,

$$h'(c) = -2\int_0^c g(y)dy + \int_0^1 g(y)dy - (1-2c)g(c) - 2c(1-c)g'(c), \tag{32}$$

$$h''(c) = -3(1-2c)g'(c) - 2c(1-c)g''(c), \tag{33}$$

so that $h'(0) = \int_0^1 g(y)dy - g(0)$ and $h'(1) = g(1) - \int_0^1 g(y)dy$, as well as $h''(0) = -3g'(0)$ and $h''(1) = 3g'(1)$. Under Condition 1, both $h'(0)$ and $h'(1)$ have the same sign. Hence, both near 0 and 1, h has the same monotonous behaviour (increasing or decreasing). Since $h(0) = h(1) = 0$, this requires that h' switches signs at least twice. Hence the solution to (7) does exist and is located between the smallest root and largest root of $h'(x) = 0$ contained in $(0,1)$.

A.3 Proof of Proposition 3

We start with the first statement. First of all, from Proposition 1, we have that the optimal split is not affected by the noise structure. Second, the dispersion term $V(c)$ in 4 can be re-written as

$$V(c) = \mathbb{E}\left[(g(X) - m^-(c))^2\mathbf{1}_{\{X<c\}}\right] + \mathbb{E}\left[(g(X) - m^+(c))^2\mathbf{1}_{\{X>c\}}\right] + \sigma_E^2,$$

[18] We saw in the second case in the previous proof that if the first term of (31) is zero, then V is maximized.

where the terms $m^{\pm}(c)$ depend on X only. The numerator in (5) is

$$\mathbb{V}[Y] = \mathbb{E}[(g(X) + E - \mathbb{E}[g(X])^2] = \mathbb{E}[g(X)^2] - \mathbb{E}[g(X)]^2 + \sigma_E^2, \tag{34}$$

so that any gain has form

$$G = \mathbb{E}[g(X)^2] - \mathbb{E}[g(X)]^2 - \mathbb{E}\left[(g(X) - m^-(c))^2 \mathbf{1}_{\{X<c\}}\right] + \mathbb{E}\left[(g(X) - m^+(c))^2 \mathbf{1}_{\{X>c\}}\right],$$

which does not depend on σ_E^2.

We then switch to the second statement. We recall $m^- = \mathbb{E}[g(X)\mathbf{1}_{\{X<c\}}]/\mathbb{P}[X < c]$ and $m^+ = \mathbb{E}[g(X)\mathbf{1}_{\{X>c\}}]/\mathbb{P}[X > c]$. We drop the dependence in c for notational convenience. The second term in (5) is equal to

$$\begin{aligned}
V(c) &= \mathbb{E}\left[\left(g(X) - m^-\right)^2 \mathbf{1}_{\{X<c\}}\right] + \mathbb{E}\left[\left(g(X) - m^+\right)^2 \mathbf{1}_{\{X>c\}}\right] + \sigma_E^2 \\
&= \mathbb{E}[g(X)^2] + (m^-)^2\mathbb{P}[X < c] + (m^+)^2\mathbb{P}[X > c] - 2m^-\mathbb{E}[g(X)\mathbf{1}_{\{X<c\}}] - 2m^+\mathbb{E}[g(X)\mathbf{1}_{\{X>c\}}] + \sigma_E^2 \\
&= \mathbb{E}[g(X)^2] - (m^-)^2\mathbb{P}[X < c] - (m^+)^2\mathbb{P}[X > c] + \sigma_E^2
\end{aligned}$$

We now write without loss of generality $c = \mathbb{P}[X < c]$ and $1 - c = \mathbb{P}[X > c]$, as if X was uniformly distributed. This is just meant to simplify the notations. We have

$$\begin{aligned}
V(c) &= \mathbb{E}[g(X)^2] - \mathbb{E}[g(X)]^2 + \mathbb{E}[g(X)]^2 - \frac{\mathbb{E}[g(X)\mathbf{1}_{\{X<c\}}]^2}{c} - \frac{\mathbb{E}[g(X)\mathbf{1}_{\{X>c\}}]^2}{1-c} + \sigma_E^2 \\
&= \mathbb{E}[g(X)^2] - \mathbb{E}[g(X)]^2 \\
&\quad + \frac{c(1-c)\mathbb{E}[g(X)(\mathbf{1}_{\{X<c\}} + \mathbf{1}_{\{X>c\}})]^2 - (1-c)\mathbb{E}[g(X)\mathbf{1}_{\{X<c\}}]^2 - c\mathbb{E}[g(X)\mathbf{1}_{\{X>c\}}]^2}{c(1-c)} + \sigma_E^2 \\
&= \mathbb{E}[g(X)^2] - \mathbb{E}[g(X)]^2 - \frac{\left((1-c)\mathbb{E}[g(X)\mathbf{1}_{\{X<c\}}] - c\mathbb{E}[g(X)\mathbf{1}_{\{X>c\}}]\right)^2}{c(1-c)} + \sigma_E^2 \\
&= \mathbb{V}[Y] - c(1-c)(\mathbb{E}[g(X)\mathbf{1}_{\{X<c\}}] - \mathbb{E}[g(X)\mathbf{1}_{\{X>c\}}])^2
\end{aligned}$$

where, in the last lines, several simplifications occur. The final expression of the gain follows (combining the above with (34)). The fourth line is a simple application of the definition of the conditional expectation (e.g., $\mathbb{E}[g(X)\mathbf{1}_{\{X<c\}}] = \mathbb{E}[g(X)|X < c]P[X < c]$).

A.4 Proof of Proposition 4

We have

$$P[X \leq x, Y \leq y] = \int_{v\in[0,x]} \int_{z\in[-\infty, y-ax-b]} f(z, v)dvdz,$$

so that a double differentiation in x and y leads to

$$P[X \in dx, Y \in dy] = f(y - ax - b, x)dxdy.$$

We then recall the inverse c.d.f. function of Y, $F_Y^{-1}(q)$, which satisfies $P[y \leq F_Y^{-1}(q)] = q$. Conditionally on $Y > F_Y^{-1}(1-q)$ or $Y < F_Y^{-1}(q)$, the density of X is thus defined as

$$\begin{aligned}
P_q(x) &= P\left[X \in dx | \{(Y > F_Y^{-1}(1-q)) \cup (Y < F_Y^{-1}(q))\}\right] \qquad (35) \\
&= \frac{\int_{(y>F_Y^{-1}(1-q))\cup(y<F_Y^{-1}(q))} f(y - ax - b, x)dy}{\int_{\mathbb{R}} \int_{(y>F_Y^{-1}(1-q))\cup(y<F_Y^{-1}(q))} f(y - ax - b, x)dxdy} dx \\
&= \frac{\int_{(y>F_Y^{-1}(1-q)-ax-b)\cup(y<F_Y^{-1}(q)-ax-b)} f(y, x)dy}{\int_{\mathbb{R}} \int_{(y>F_Y^{-1}(1-q)-ax-b)\cup(y<F_Y^{-1}(q)-ax-b)} f(y, x)dxdy} dx, \qquad (36)
\end{aligned}$$

where the third line was obtained by simple substitution. If μ_X is the median of X, we omit the scaling denominator and compute

$$\begin{aligned}
P_q(\mu_X + x) - P_q(\mu_X - x) \propto & \int_{(y>F_Y^{-1}(1-q)-a(\mu_X+x)-b)\cup(y<F_Y^{-1}(q)-a(\mu_X+x)-b)} f(y, \mu_X + x)dy \\
& - \int_{(y>F_Y^{-1}(1-q)-a(\mu_X-x)-b)\cup(y<F_Y^{-1}(q)-a(\mu_X-x)-b)} f(y, \mu_X - x)dy \\
= & \int_{(y>F_Y^{-1}(1-q)-a(\mu_X+x)-b)\cup(y<F_Y^{-1}(q)-a(\mu_X+x)+b)} f(y, \mu_X + x)dy \\
& - \int_{(y>F_Y^{-1}(1-q)-a(\mu_X-x)-b)\cup(y<F_Y^{-1}(q)-a(\mu_X-x)-b)} f(y, \mu_X + x)dy \\
= & \int_{F_Y^{-1}(1-q)-a(\mu_X+x)-b}^{F_Y^{-1}(1-q)-a(\mu_X-x)-b} f(y, \mu_X + x)dy \\
& - \int_{F_Y^{-1}(q)-a(\mu_X+x)-b}^{F_Y^{-1}(q)-a(\mu_X-x)-b} f(y, \mu_X + x)dy.
\end{aligned}$$

where in the second expression, we have used the symmetry $f(z, \mu_X + x) = f(z, \mu_X - x)$. To prove the first point of the proposition, it suffices to show that the two terms are equal. By the symmetry of f, it suffices to show that the integration ranges of each integral are symmetric around zero, i.e., that the middle points of each range are equidistant from zero. These middle points are located at $F_Y^{-1}(1-q) - a\mu_X - b$ and $F_Y^{-1}(q) - a\mu_X - b$. If we assume $\mu_Y = a\mu_X + b$, the sum of these two values is equal to

$$F_Y^{-1}(1-q) + F_Y^{-1}(q) - 2(a\mu_X + b) = F_Y^{-1}(1-q) + F_Y^{-1}(q) - 2\mu_Y = 0,$$

because $F_Y^{-1}(1-q)$ and $F_Y^{-1}(q)$ are located symmetrically around μ_Y. To complete the proof of the first point of the proposition, it suffices to show that indeed the medians satisfy $\mu_Y = a\mu_X + b$.

$$\begin{aligned}
P[Y \leq a\mu_X + b] - P[Y \geq a\mu_X + b] &= \int_{-\infty}^{a\mu_X+b} \int_{\mathbb{R}} f(y - ax - b, x)dxdy - \int_{a\mu_X+b}^{\infty} \int_{\mathbb{R}} f(y - ax - b, x)dxdy \\
&= \int_{\mathbb{R}} \left(\int_{-\infty}^{a(\mu_X - x)} f(y, x)dy \right) dx - \int_{\mathbb{R}} \left(\int_{a(\mu_X - x)}^{\infty} f(y, x)dy \right) dx \\
&= \int_{\mathbb{R}} \left(\int_{-\infty}^{a(\mu_X - x)} f(y, x)dy \right) dx - \int_{\mathbb{R}} \left(\int_{-\infty}^{a(x - \mu_X)} f(y, x)dy \right) dx \\
&= \int_{\mathbb{R}} \left(\int_{-\infty}^{\mu_X - y/a} f(y, x)dx \right) dy - \int_{\mathbb{R}} \left(\int_{\mu_X + y/a}^{\infty} f(y, x)dx \right) dy \\
&= 0,
\end{aligned}$$

where in the third equality we have used the symmetry in the first variable and in the last one, the symmetry in the second variable.

For the second point of the proposition, we must resort to a simplified reasoning to lighten the notations. The main function we are interested in is $f(y - a(\mu_X + x) - b, \mu_X + x)$ and its integration range is $y \notin \left(F_Y^{-1}(q), F_Y^{-1}(1-q)\right)$. If we substitute $z = -y + 2(a\mu_X + b) = -y + 2\mu_X$, the function, by symmetry in the first variable, becomes $f(z - a(\mu_X - x) - b, \mu_X + x)$ and the integration range $z \notin \left(2\mu_Y - F_Y^{-1}(q), 2\mu_Y - F_Y^{-1}(1-q)\right)$ since $F_Y^{-1}(q)$ and $F_Y^{-1}(1-q)$ are equidistant from μ, a simplification occurs and the interval is simply inverted: $z \notin \left(F_Y^{-1}(1-q), F_Y^{-1}(q)\right)$. This inversion is then compensated by the inverse sign between dy and dz. We are now ready to provide the full details. For notational convenience, we introduce the interval $S_q = \left(F_Y^{-1}(1-q), F_Y^{-1}(q)\right)$.

$$
\begin{aligned}
\mathcal{E}_q(\mu_X + x) - \mathcal{E}_q(\mu_X) &= \frac{\int_{y \notin S_q} y f(y - a(\mu_X + x) - b, \mu_X + x) dy}{\int_{y \notin S_q} f(y - a(\mu_X + x) - b, \mu_X + x) dy} - \frac{\int_{y \notin S_q} y f(y - a\mu_X - b, \mu_X) dy}{\int_{y \notin S_q} f(y - a\mu_X - b, \mu_X) dy} \\
&= \frac{\int_{y \notin S_q} (2\mu_Y - z) f(z - a(\mu_X - x) - b, \mu_X + x) dz}{\int_{y \notin S_q} f(y - a(\mu_X + x) - b, \mu_X + x) dy} - \frac{\int_{y \notin S_q} (2\mu_Y - z) f(z - a\mu_X - b, \mu_X) dz}{\int_{y \notin S_q} f(y - a\mu_X - b, \mu_X) dy} \\
&= -\frac{\int_{y \notin S_q} z f(z - a(\mu_X - x) - b, \mu_X - x) dz}{\int_{y \notin S_q} f(y - a(\mu_X - x) - b, \mu_X - x) dy} + \frac{\int_{y \notin S_q} z f(z - a\mu_X - b, \mu_X) dz}{\int_{y \notin S_q} f(y - a\mu_X - b, \mu_X) dy} \\
&= -\mathcal{E}_q(\mu_X - x) + \mathcal{E}_q(\mu_X),
\end{aligned}
$$

where in the third equality, we have discarded the $2\mu_Y$ terms which cancel out and have used two symmetry properties: that of the denominator which was proven above and that of the second variable in f.

B Details on the quadratic form

We provide further insights on the case when $g(x) = (x-b)^2$. Notably, we plot the functions that help shed light on the main issues. Essentially, our results are confirmed when looking at the objective function V defined in (16). Up to a factor that does not depend on c (the quadratic term in z^2 in π^- and π^+ defined in (17) and (18)), it can be evaluated as

$$
V_{\text{quad}}(c) = \sigma_E^2 + \frac{4 - 15b + 15b^2 - 5c + 30bc - 45b^2c - 5c^2 + 45b^2c^2 + 5c^3 - 30bc^3 + 5c^4}{45}, \tag{37}
$$

and we plot the second part for four values of b in Figure 16 below.

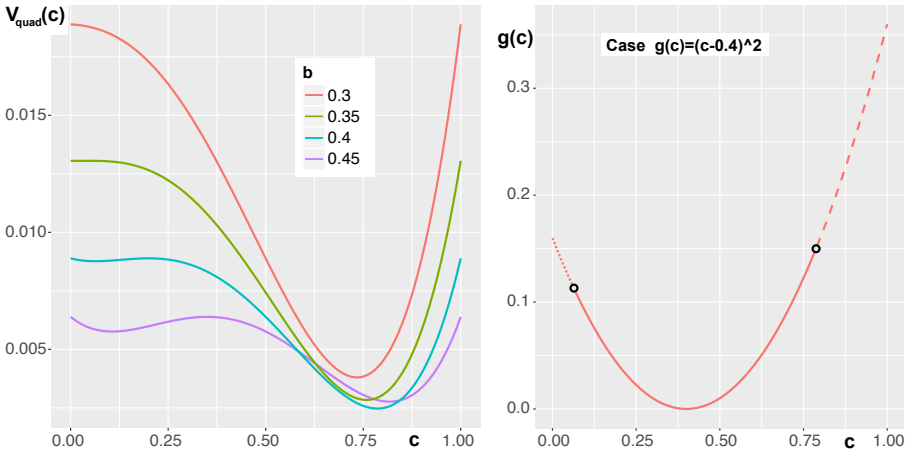

Fig. 16 Illustrations: On the left graph, we plot $V_{\text{quad}}(c)$ defined in Equation (37) with $\sigma_E^2 = 0$. On the right graph, we show the sample function $g(x) = (x - 0.4)^2$, along with the optimal splits.

In all cases, the objective function immediately starts to decrease at zero. The main difference is that for $b \in (1/3, 2/3)$, the decrease is followed by a small bump. We observe it for $b = 0.4$ and $b = 0.45$. The cause of the bump is not visually obvious but can be summarized as follows. Around 0.25, the decrease in π^+ is slow in the region around $c = 0.20$ to $c = 0.25$ and is more than compensated by the increase in π^-, hence the sum of the two increases slightly. However, past $c = 0.5$, the decrease in π^+ is sharp while the increase in π^- is slower. Hence, the true minimum is located between 0.5 and 1 (0.786 in this case).

Lastly, we highlight that the bump implies a local maxima. It is where the function h has a zero. This was treated in the proof of Proposition 1, in the case when $m^-(c) - m^+(c) = 0$.

C Impact of filter on exponential generator

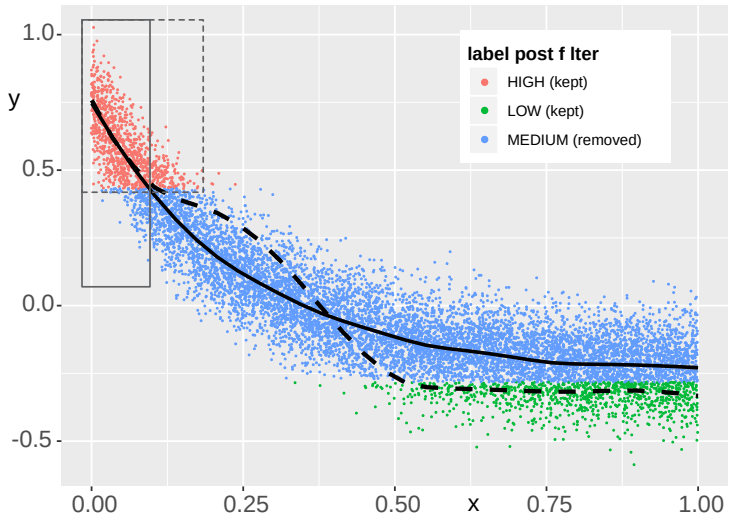

Fig. 17 Clustering post filter: the case of exponential generators. We plot the data points and show the filter effect ($q = 0.15$) with colors. The initial generator is the full black line while the one post-filter is in dotted lines. The rectangles show the homogeneous clusters: the dotted one (post-filter) goes further to the right and the split is closer to the median.

D Features

(EBITDA - Capex) / Interest Expense	Intangible Assets/Revenue
Accounts Receivable, 1 Yr. Growth %	Long-Term Debt / Total Capital
ADV12M	MKTCAP
ADV20	MoM12MFX
ADV60	MoM6MFX
Asset Turnover	Net Avail. For Common Margin %
BidAsk	Net Debt
BuyBackYld	Net Debt / EBITDA
CAPEX_depreciation	Net Income, 1 Yr. Growth %
Cash Per Share - (Ratio)	Net Margin
Common Equity, 1 Yr. Growth %	NetDbtYld
Current Ratio	NI/OA
DebtEquity	NI/TOA
Diluted EPS Before Extra, 1 Yr. Growth %	Normalized Net Income, 1 Yr. Growth %
Earnings From Cont. Operations, 1 Yr. Growth %	OCF Margin
EBIAT Ratio	OCF/BV
EBIT / Interest Expense	OCF/CE
EBIT Margin %	OCF/NOA
EBIT/BV	OCF/OA
EBIT/CE	OCF/TA
EBIT/NOA	OCF/TBV
EBIT/OA	OCF/TOA
EBIT/TA	Op Margin
EBIT/TBV	Op_Prt_Margin
EBIT/TOA	Payables/Receivables
EBITA, 1 Yr. Growth %	PB
EBITDA	PE
EBITDA / Interest Expense	Prof_on_Assets
EBITDA Margin %	Quick Ratio
EV_EBITDA	R12M_USD
FCF Margin	R1M_USD
FCF_on_Assets	R3M_USD
FCF/BV	R6M_USD
FCF/CE	Recurring Earnings/Total Assets (%)
FCF/NOA	Return on Capital (%)
FCF/OA	ROA
FCF/TA	ROC
FCF/TBV	ROCE
FCF/TOA	ROE
FCFYld	RONOA
FFO to Gross Profit (x)	ROTE
Fixed Assets Turnover (Average Fixed Assets)	RSI_12M_FX
GP Margin	RSI_3M_FX
GP/BV	Share_Turn
GP/CE	Tangible Book Value, 1 Yr. Growth %
GP/NOA	TEV less Market Cap
GP/OA	Total Capital
GP/TA	Total Debt
GP/TBV	Total Debt to Capital (%)
GP/TOA	Total Debt/Revenue
Gross Profit Margin %	Total Liabilities / Total Assets
Gross Profit, 1 Yr. Growth %	VolXPostLTFX
Gross_Profit	VolXPostSTFX
Income From Continuing Operations Margin %	Working Capital, 1 Yr. Growth %

Table 6 List of the 108 features. Most abbreviations are standard in accounting.

References

Ali, Ö. G. and K. Yaman (2013). Selecting rows and columns for training support vector regression models with large retail datasets. *European Journal of Operational Research 226*(3), 471–480.

Ammann, M., G. Coqueret, and J.-P. Schade (2016). Characteristics-based portfolio choice with leverage constraints. *Journal of Banking & Finance 70*, 23–37.

Asness, C., A. Frazzini, R. Israel, T. J. Moskowitz, and L. H. Pedersen (2018). Size matters, if you control your junk. *Journal of Financial Economics 129*(3), 479–509.

Asness, C. S., A. Frazzini, R. Israel, and T. J. Moskowitz (2014). Fact, fiction and momentum investing. *Journal of Portfolio Management 40*(5), 75–92.

Asness, C. S., A. Frazzini, and L. H. Pedersen (2014). Quality minus junk. *Review of Accounting Studies 24*(1), 1–79.

Baker, M., B. Bradley, and R. Taliaferro (2014). The low-risk anomaly: A decomposition into micro and macro effects. *Financial Analysts Journal 70*(2), 43–58.

Ballings, M., D. Van den Poel, N. Hespeels, and R. Gryp (2015). Evaluating multiple classifiers for stock price direction prediction. *Expert Systems with Applications 42*(20), 7046–7056.

Barroso, P. and P. Santa-Clara (2015). Momentum has its moments. *Journal of Financial Economics 116*(1), 111–120.

Bertsimas, D. and J. Dunn (2017). Optimal classification trees. *Machine Learning 106*(7), 1039–1082.

Brandt, M. W., P. Santa-Clara, and R. Valkanov (2009). Parametric portfolio policies: Exploiting characteristics in the cross-section of equity returns. *Review of Financial Studies 22*(9), 3411–3447.

Breiman, L., J. Friedman, C. J. Stone, and R. Olshen (1984). *Classification And Regression Trees*. Chapman and Hall.

Chandrashekar, G. and F. Sahin (2014). A survey on feature selection methods. *Computers & Electrical Engineering 40*(1), 16–28.

Chen, T. and C. Guestrin (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. ACM.

Chou, P. A. (1991). Optimal partitioning for classification and regression trees. *IEEE Transactions on Pattern Analysis & Machine Intelligence 4*, 340–354.

Daniel, K. and T. J. Moskowitz (2016). Momentum crashes. *Journal of Financial Economics 122*(2), 221–247.

DeMiguel, V., L. Garlappi, and R. Uppal (2007). Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy? *Review of Financial Studies 22*(5), 1915–1953.

Esposito, F., D. Malerba, and G. Semeraro (1997). A comparative analysis of methods for pruning decision trees. *IEEE Transactions on Pattern Analysis & Machine Intelligence 19*(5), 476–491.

Fama, E. F. and K. R. French (1992). The cross-section of expected stock returns. *Journal of Finance 47*(2), 427–465.

Fama, E. F. and K. R. French (2015). A five-factor asset pricing model. *Journal of Financial Economics 116*(1), 1–22.

Fu, X., J. Du, Y. Guo, M. Liu, T. Dong, and X. Duan (2018). A machine learning framework for stock selection. *arXiv preprint 1806.01743*.

Galili, T. and I. Meilijson (2016). Splitting matters: how monotone transformation of predictor variables may improve the predictions of decision tree models. *arXiv preprint 1611.04561*.

Goyal, A. (2012). Empirical cross-sectional asset pricing: a survey. *Financial Markets and Portfolio Management 26*(1), 3–38.

Green, J., J. R. Hand, and X. F. Zhang (2013). The supraview of return predictive signals. *Review of Accounting Studies 18*(3), 692–730.

Gu, S., B. T. Kelly, and D. Xiu (2019). Empirical asset pricing via machine learning. *Review of Financial Studies Forthcoming*(XX), XXX–XXX.

Guida, T. and G. Coqueret (2018). Ensemble learning applied to quant equity: gradient boosting in a multifactor framework. In *Big Data and Machine Learning in Quantitative Investment*, pp. 129–148. Wiley.

Guyon, I. and A. Elisseeff (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research 3*(Mar), 1157–1182.

Harvey, C. R. (2017). Presidential address: the scientific outlook in financial economics. *Journal of Finance 72*(4), 1399–1440.

Harvey, C. R., Y. Liu, and H. Zhu (2016). ... and the cross-section of expected returns. *Review of Financial Studies 29*(1), 5–68.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The Elements of Statistical Learning*. Springer.

Huck, N. (2019). Large data sets and machine learning: applications to statistical arbitrage. *European Journal of Operational Research 278*(1), 330–342.

Kelly, B. T., S. Pruitt, and Y. Su (2019). Characteristics are covariances: A unified model of risk and return. *Journal of Financial Economics 134*(3), 501–524.

Koijen, R. S. and M. Yogo (2019). A demand system approach to asset pricing. *Journal of Political Economy 127*(4), 1475–1515.

Köksalan, M. and C. T. Şakar (2016). An interactive approach to stochastic programming-based portfolio optimization. *Annals of Operations Research 245*(1-2), 47–66.

Krauss, C., X. A. Do, and N. Huck (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European Journal of Operational Research 259*(2), 689–702.

Kuhn, M. and K. Johnson (2019). *Feature Engineering and Selection: A Practical Approach for Predictive Models*. Chapman & Hall/CRC.

Linnainmaa, J. T. and M. R. Roberts (2018). The history of the cross-section of stock returns. *Review of Financial Studies 31*(7), 2606–2649.

Liu, H. and H. Motoda (2012). *Feature selection for knowledge discovery and data mining*, Volume 454. Springer Science & Business Media.

Loh, W.-Y. et al. (2009). Improving the precision of classification trees. *The Annals of Applied Statistics 3*(4), 1710–1737.

Lopez, O., X. Milhaud, P.-E. Thérond, et al. (2016). Tree-based censored regression with applications in insurance. *Electronic Journal of Statistics 10*(2), 2685–2716.

Moritz, B. and T. Zimmermann (2016). Tree-based conditional portfolio sorts: The relation between past and future stock returns. *SSRN Working Paper 2740751*.

Norouzi, M., M. Collins, M. A. Johnson, D. J. Fleet, and P. Kohli (2015). Efficient non-greedy optimization of decision trees. In *Advances in Neural Information Processing Systems*, pp. 1729–1737.

Novy-Marx, R. (2012). Is momentum really momentum? *Journal of Financial Economics 103*(3), 429–453.

Pätäri, E. and T. Leivo (2017). A closer look at value premium: Literature review and synthesis. *Journal of Economic Surveys 31*(1), 79–168.

Patel, J., S. Shah, P. Thakkar, and K. Kotecha (2015). Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert Systems with Applications 42*(1), 259–268.

Romano, J. P. and M. Wolf (2013). Testing for monotonicity in expected asset returns. *Journal of Empirical Finance 23*, 93–116.

Sasane, A. (2016). *Optimization in Function Spaces*. Courier Dover Publications.

Settles, B. (2012). Active learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning 6*(1), 1–114.

Stahl, F. and M. Bramer (2012). Jmax-pruning: A facility for the information theoretic pruning of modular classification rules. *Knowledge-Based Systems 29*, 12–19.

Stambaugh, R. F. (1999). Predictive regressions. *Journal of Financial Economics 54*(3), 375–421.

Van Dijk, M. A. (2011). Is size dead? A review of the size effect in equity returns. *Journal of Banking & Finance 35*(12), 3263–3274.

Yan, Z., Z. Chen, G. Consigli, J. Liu, and M. Jin (2019). A copula-based scenario tree generation algorithm for multiperiod portfolio selection problems. *Annals of Operations Research*, 1–33.