Review of Accounting Studies (2020) 25:1135–1155
https://doi.org/10.1007/s11142-020-09554-9

INVITED SUBMISSION

Machine learning improves accounting: discussion, implementation and research opportunities

Jeremy Bertomeu[1]

Published online: 13 August 2020

Abstract

Machine learning has been growing in importance in empirical accounting research. In this opinion piece, I review the unique challenges of going beyond prediction and leveraging these tools into generalizable conceptual insights. Taking as springboard "Machine learning improves accounting estimates" presented at the 2019 Conference of the Review of Accounting Studies, I propose a conceptual framework with various testable implications. I also develop implementation considerations panels with accounting data, such as colinearities between accounting numbers or suitable choices of validation and test samples to mitigate between-sample correlations. Lastly, I offer a personal viewpoint toward embracing the many low-hanging opportunities to bring the methodology into major unanswered accounting questions.

Keywords Machine learning · Accounting · Estimates · Modelling

JEL Classification C4 · C5 · G3 · M2 · M4

In their new book, *The End of Accounting and the Path Forward For Inventors and Managers*, Baruch Lev and Feng Gu provocatively argue that accounting has not kept pace with secular changes in economic structures. With the decline in the importance of controlling physical means of production, the value of businesses is increasingly driven by intangibles assets - a knowledge economy in which know-hows, customers, brands, and networks explain investor value. Their analysis further takes stock of the growing disconnect between markets where antiquated procedures focus on minutia of historical events of no interest to investors, bury relevant information into aggregated reports, and are often contaminated by managerial judgment.

Baruch Lev is a co-author of this year's must-read paper at the Review of Accounting Studies 2019 conference (Ding et al. 2019) and that we should see similar themes

✉ Jeremy Bertomeu
bjeremy@wustl.edu

[1] Olin School of Business, Washington University, St. Louis, MO, USA

comes as no surprise. This begs the question: Can machine learning help provide better high-quality forward-looking information? Indeed, interests in the accounting community have been growing to integrate machine learning as a set of reporting tools to predict, diagnose, and improve reporting quality, with various new studies showing the quality of machine learning to predict errors and irregularities (Perols 2011; Perols et al. 2017; Bertomeu et al. 2019; Bao et al. 2019), measuring information content (Li 2010; Barth et al. 2019), analyzing financial statements (Binz et al. 2020) or improving audit procedures (Gerakos et al. 2016; Sun 2019), among many others.

In this essay, my objective is twofold. First, leveraging on the insights from Ding et al. (2019), I will describe a new research paradigm that is slowly emerging from the application of machine learning to accounting research. I will argue that, while the tools of machine learning are designed to optimize prediction, as René Thom puts it, "Prédire n'est pas expliquer," i.e., to predict is not to explain, and our role as social scientists is to draw new theoretical insight from a better understanding of complex data. To do this, I will develop a simple conceptual framework to explain the performance of machine learning, show how perspectives of machine learning can inform accounting theory, and point the curious reader to newly available tools and methods to interpret the models.

My second objective is more practical. The use of machine learning is new in accounting, and, with it, come new challenges in fitting tools that were initially not designed from the type of panel or time-series data typically obtained in accounting. I will discuss some of the challenges in applying these tools in accounting datasets and develop common approaches adopted in the existing literature. Then I will illustrate (by example) how to implement a simple machine learning algorithm to eliminate any barrier to entry for researchers interested in bringing machine learning into their own research. Supporting Python code will be distributed on the website of the *Review of Accounting Studies*.

1 Beyond Prediction

1.1 A Conceptual Framework

Machine learning can be thought as an algorithm which outputs an estimator $g^{ML}(h_t)$ of a particular quantity of interest r_t, given an information set h_t observable by an outside user (e.g., regulator, investor, etc.). Suppose that the machine learning algorithm aims to efficiently estimate the mean of this quantity given all known information:[1]

$$g^{ML}(h_t^p) = \mathbb{E}(r_t|h_t^p). \tag{1}$$

[1] This would be desirable if, for example, a decision-maker bears a quadratic loss $\mathbb{E}((g - r_t)^2|h^t)$ when making a decision based on g. This representation is a normalization to the extent that we can always define r_t as the quantity whose first moment is of interest to a decision-maker: if the decision-maker has a loss function $\mathbb{E}(L(g, r_t)|h_t)$ with an optimum given by the first-order condition $\mathbb{E}(L_1(g^{ML}(h^t), r_t)|h_t) = 0$, we can redefine the (implied) quantity of interest as $r_t' \equiv L_1(g^{ML}(h^t), r_t) + g^{ML}(h^t)$, which satisfies (1).

Management, on the other hand, applies a different estimation procedure, which could be driven by accounting procedures or their own judgment and incentives, and makes an estimate $g^m(h_t^m)$ to maximize

$$g^m(h_t^m) \in \arg\max_g \mathbb{E}_m(V(g, r_t)|h_t^m), \tag{2}$$

where $V(.)$ is the objective of the manager, $\mathbb{E}_m$ is a subjective expectation, and h_t^m is the manager's information set when making an estimate.

To set ideas, consider the reporting problem examined by Ding et al. but that one can easily apply to any accounting estimation problem. Table 1 zooms in on the insurance report issued by Lloyds for 2002–2007. Lloyds estimated that, in 2002, that \$7.463 (millions) would be paid off on claims occurring in 2002, of which \$5.354 was settled while a remaining \$2.109 were accrued as liabilities. An incremental \$6.884-\$5,354=\$1.530 was paid off on these claims in 2003. At this point, the majority of the claims would be considered settled or unlikely to trigger payment, so the company reduces its assessment of total repayments down to \$7.270, reducing its remaining liability to \$7.270-\$6.884m=\$.386. Ultimately, within a six-year time frame, only \$7.111 was paid, and management over-estimated payments by $g^m(h_t^m) - r_t$=\$7.463-\$7.111=\$0.352, a hair below 5% of actual payments.

Ding et al. consider different algorithms $g^{ML}(h_t)$ to improve these estimates using information h_t that could be used as an input to make these estimates, e.g., premiums charged, settlements and payments made as well as a number of company and macroeconomic characteristics. Obviously, we would expect machine learning to do worse because an outside user does not have the field expertise of the insurer to make quality estimates. Shockingly, Ding et al. discover a very different pattern in the data. Not only is g^m far from a sufficient statistic to predict settlements but, in four out of five business segments, g^{ML} has smaller out-of-sample errors than g^m even if we do not include management estimates. When included into the algorithm, management estimates had little incremental predictive power and are not always the most important predictor of settlements.

These results decisively support the theory that users would be better off making their own objective estimates over management estimates, lining up with the integrated reporting framework in *The End of Accounting*, which would empower users with the inputs of managerial estimates. Yet what it does not say is what may be driving such higher, and predictable management errors in the form $|r_t - g^{ML}(h_t^p)| <$

Table 1 Incurred and Paid Losses for 2002 Claims at Lloyds

Claim year	Estimate Year	Estimated	Settled
2002	2002	\$7.463	\$5.354
	2003	\$7.270	\$6.884
	2004	\$7.064	\$6.987
	2005	\$7.178	\$7.045
	2006	\$7.169	\$7.060
	2007	\$7.135	\$7.111

$|r_t - g^m(h_t^m)|$. Below, I decompose errors in three broad economic explanations, whose implications for users and regulators may be quite different.

First, h_t^m may be coarser than h_t^p because the management estimate ignores some of the public information used in the algorithm. My (admittedly subjective) conjecture is that this is less the result of ignorance or availability of information than difficulties in incorporating information that does not fit well into a formal model or a set of accounting procedures. For example, it is well-documented that even sophisticated financial experts do not fully incorporate all macro news into their estimates (Hugon et al. 2016). Further, accounting procedures used to generate estimates are often anchored on historical realized settlements rather than incorporating into a statistical model all variables known to correlate to the variable of interest. I will refer to this explanation as the lost information hypothesis (A).

Second, the manager may be using an incorrect statistical model, given a particular information set, such that the subjective expectation $\mathbb{E}_m(.) \neq \mathbb{E}(.)$. Consider a manager with a miscalibrated prior anticipating a high conversation of claims into settlements; the manager would accrue $g^m(h^m) = \mathbb{E}_m(r_t|h_t^p) > \mathbb{E}(r_t|h_t^p)$. This type of miscalibration creates patterns detectable dynamically because, as uncertainty realizes, the estimate will predictably drift toward the true value, over-estimating (on average) settlement amounts over the entire path. Naturally, many other errors could cause the statistical model to be misspecified, such as relying too much on recent experience, putting weight on signals irrelevant for the decision at hand or missing important interactions between variables. In accounting, these errors can occur as a result of behavioral biases but, of greater relevance to accounting regulators, also the procedural rule books that govern how estimates are made and audited. For later use, I refer to this collective set of explanations as the bad model hypothesis (B).

Third, managers face their own objectives that need not be fully aligned with investors. If the manager's objective function is not set to minimize a squared error $V(g, r_t) = -(g - r_t)^2$, the management estimate g^m solving (2) need not be $\mathbb{E}_m(r_t|h_t^m)$ and may exhibit greater estimation errors than $g^{ML}(h^t)$, even if the information sets $h_t^m = h_t$ are identical and the manager uses a correct statistical model $\mathbb{E} = \mathbb{E}_m$. To illustrate this further in a different context, Gu and Wu (2003) show that a financial expert forecasting to minimize absolute errors would issue estimate according to the median rather than the mean. When choosing accruals, many complex considerations take place, such as the effect of accruals on income statement accounts used to assess valuation and performance or the avoidance of under-reported liabilities that could later cause regulatory intervention or lawsuits (Dechow and Skinner 2000; Dechow et al. 2010). I will refer to these explanations as the user misalignment hypothesis (C).

1.2 Causes of Management Errors

How do we distinguish between lost information (A), bad model (B) and user misalignment (C) without detailed institutional of the process through which the estimates were made? As it turns out, machine learning needs not end with improved prediction but can give us pathways to open the black box of what led to bad estimates.

Consider the lost information hypothesis (A). This hypothesis can be tested in a fairly straightforward manner by comparing the variables that enter the construction g^{ML} versus those variables that enter the construction of g^m. While the model that led to g^m is unobservable, one may decompose g^m by predicting the management estimate as a function of various variables important in g^{ML}, i.e., using machine learning to construct:

$$g^{ML,m}(h_t) = \mathbb{E}(g^m(h_t^m)|h_t). \tag{3}$$

The existence of variables important in g^{ML} but not $g^{ML,m}(h_t)$ would suggest that these variables have been omitted in the manager's estimation model.

Certain features of g^{ML} can also suggest lost information. Comparing models with versus without management estimates, variables that are not used in the management estimate are likely to retain their importance or become relatively more important, while variables being considered by management should lose most of their importance after incorporating management estimates. Finally, given prior research, certain types of information are less likely to be part of the construction of management estimates because, institutionally, procedures do not make adjustments for these variables. With this logic at hand, let us apply it to the insurance environment of Ding et al.. The ranking of variables by importance appears, by and large, unchanged by the inclusion of managerial estimates, suggesting that no important variables seem to be fully omitted from the management model. Further, among all variables used by Ding et al., the two most likely variables not to take part in management procedures to estimate payments are GDP growth and inflation. GDP growth would require a model of how changes in the aggregate economy translate into claim severity, and inflation has remained at unprecedented historically low and stable levels. Overall, the evidence seems to rule out lost information as a significant cause of forecast errors in this environment.

The bad model hypothesis (B) is more difficult to assess because we do not directly observe the model used by management, and, even if we could survey management, it would be likely based on mental processes that map inputs into predictions that are neither quantifiable nor machine-readable. To address this difficult problem, let us take a seemingly unrelated parenthesis to delve into interpretation of machine learning models.

Unfortunately, there is a price to pay in exchange for better predictions, in the form of models that do not have closed-form, and thus how a prediction is made can be obscure. Random forests and gradient boosted regression trees, two well-regarded algorithms featured in Ding et al., are examples of ensemble learning methods that combine the prediction of a large number of trees (around 100 trees in Ding et al. but up thousands of trees in applications with larger data sets). There are, however, approaches available to researchers to decode what lies behind a prediction and Ding et al. shall serve us as an example generalizable to any setting. If $g^{ML}(.)$ is sufficiently effective to have filtered a sufficient portion of the noise, using simpler models to analyze the predictions $g^{ML}(.)$ with a simpler (closed-form) model

$$g^{ML,s}(h_t) = \mathbb{E}(g^{ML}(h_t)|h_t) \tag{4}$$

will help clarify how the model makes its prediction. One example of this approach would be to run a regression with suitable (but limited) interaction terms using $g^{ML}(h_t)$ as a dependent variable.

Computer scientists have also contributed additional tools to conduct this type of analysis more systematically. InTrees (Deng 2019), a short-hand for "interpreting tree ensembles," provides a toolbox for generating simple actionable decision rules as a function of a subset of variables. When applied to the Ding et al. setting, one of the decision rules generated by InTrees involves a management estimate below \$124.54, assets below \$385.154, current settlements below \$1.270 for a predicted cumulative payment of \$1.330. InTrees can provide a large number of such rules, partitioning the sample into homogenous groups whose characteristics inform the researcher about how predictions.

Another approach, illustrated in Fig. 1 is to run a weak learner (i.e., a tree with few branches) on a model with management estimates to identify which interactions seem to add explanatory power to management estimates.[2] Along this weak learner, the management estimate is the first variable that is used to predict settlements, with a cutoff at \$1.25 separating high and low expected settlements. For high expected settlements, the weak learner further subdivides the sample as a function of (solely) management estimates, perhaps consistent with more effort in producing estimates for larger magnitudes. For moderate management estimates, however, current settlements become incrementally informative.

Now, what do we learn about the bad model hypothesis (B) from a better understanding of the inner workings of $g^{ML}(h_t)$? The subgroup identified by InTrees suggests that among observations with low managerial estimates, lower current settlements may serve to further narrow down the predicted cumulative payment (Indeed, InTrees favors using current settlements over a finer cut on managerial estimates.) The weak learner in Fig. 1 points to a similar insight, implying, for managerial estimates that are below \$1.250 but higher than \$0.27, actual settlements serve to make better estimates. Both analyses point to specific regions of managerial estimates in which the manager may be using a bad model.

Last but not least, an economic perspective on accounting naturally points to the many incentives mechanisms that create a misalignment between the demand for information by users, and preparers' choice to supply information with contracting, financing, and regulatory consequences. Without further tests, this misalignment would be perceived as part of the bad model hypothesis, and hence additional theory is required to disentangle estimation errors unrelated to personal motive, from biased estimates that serve a particular reporting objective (C).

Identifying (C) requires theory about the causes of the misalignment that would drive managerial discretion. Unfortunately, for this particular purpose, the private nature of most companies examined by Ding et al. makes this question difficult to answer, as we do not observe well the financial stains borne by the company, the compensation contracts, or stock market incentives. Yet, even in this context, variables that would seem plausibly irrelevant to a prediction exercise, given the more

[2] I thank Ting Sun for sharing this analysis for purposes of discussion.

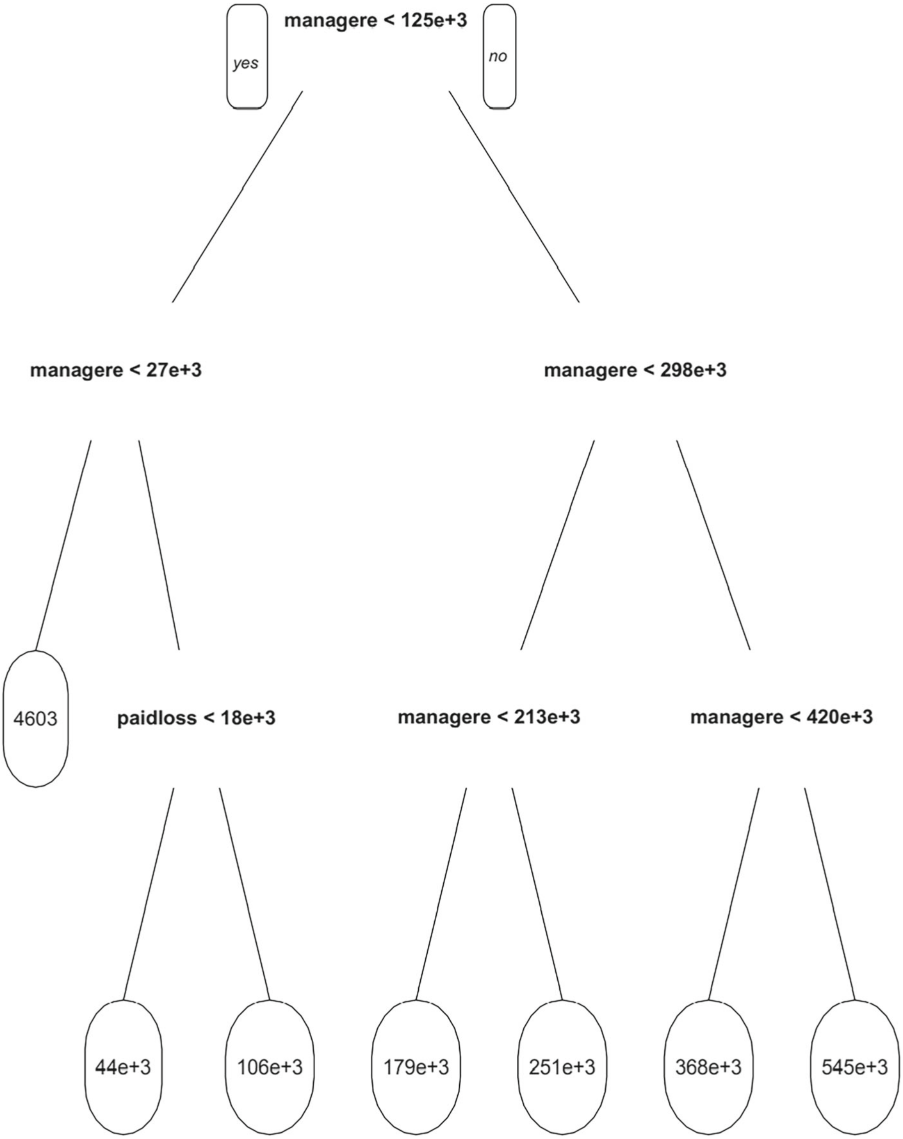

Fig. 1 Predicting cumulative settlements (a weak learner)

disaggregated variables, but may capture a reporting incentive may provide evidence of (C). If an insurance company features more stress on its capital structure due to increasing assets and declining liabilities, pressures to make more aggressive reports would increase. Ding et al., however, show that aggregate accounting variables do not rank high in terms of importance.

To generalize this approach, machine learning provides a method to test more systematically for biases correlated to incentive variables, by regressing management errors $g^m(h_m^t) - g^{ML}(h^t)$ as a function of incentive determinants X_t:

$$g^m(h_m^t) - g^{ML}(h^t) = \beta' X_t + \epsilon_t. \tag{5}$$

This is inherently a traditional research design that compares reports to a “normal” predicted level $g^{ML}(h^t)$. Machine learning can dramatically improve over the quality of this approach by offering a rich, nonlinear model for this expectation. Interestingly, when running this approach, Ding et al. find that errors are correlated to many variables likely to capture incentives: tax shields, violation of regulated ratios and small profits all appear to be correlated to management errors. To summarize, the model suggests that errors would be at least partly driven by user misalignment.

2 Steps into machine learning

2.1 Fact and Fictions

Most researchers in accounting would be familiar with classical statistics, and, while there are many texts that cover machine learning, it is helpful for us to ease the transition to machine learning by building a bridge from statistics to machine learning. Let us begin with a few misconceptions about machine learning.

Machine learning is about prediction, while statistics is about interpretation Accurate prediction is an active subfield of statistics, and research in this area offers many tools commonly used for macro data or high-frequency series, see, e.g., Elliott and Timmermann (2013). Even when not explicitly modelling prediction, the word “interpretation” in econometrics refers to identifying the true causal mechanisms that can predict outcomes out-of-sample. For counter-factual experiments that are not yet observable, because a regulator has not yet passed a regulation or the researchers aim to predict and avoid, certain bad outcomes before they occur, the assumptions behind the causal model provide the means to accurate prediction (Bertomeu et al. 2016).

Machine learning does not require functional forms, while statistical models require user-supplied functional forms Many models used in machine learning use functional forms, e.g., linear regressions are a conventional machine learning algorithm. Vice-versa, many statistical models aim to identify smooth relationships between dependent and independent variables that do not require a parametric knowledge of distributions or functional forms (Pagan and Ullah 1999), and applied researchers commonly augment models with interaction and polynomial forms to capture nonlinearities.

Machine learning does not require a mathematical formalization, while statisticians specify the data-generating process Statistics has a long tradition of theoretical work examining the theoretical properties of estimators. However, this is primarily a consequence of the maturity of the area, which has given researchers time to develop a complete theoretical background for most estimation procedures. But whether the researcher explicitly specifies a data-generating process, the scientific method would assume the existence of an objective data-generating process, which implies the existence of (objective) mathematical properties for both standard estimators and machine learning estimators. We do not know yet the asymptotic properties of most common algorithms in machine learning, nor do we have mathematical guidance to theoretically choose between algorithms or quantify the quality of predictions, but this is the object of ongoing research (Wager and Athey 2018).

Machine learning must balance learning versus overfitting, while statistical models do not need to consider overfitting Many statistical models can be prone to overfitting and have explicit parameters that would overfit the data if left unchecked. Any applied researcher, for example, would be familiar with how firm fixed effects can absorb the economic variation of interest in short panels and how including too many irrelevant variables or interactions can mute estimates and increase standard errors on important estimates. In nonparametric kernel estimation, the choice of bandwidth guides whether the model uses very few observations to fit an expectation or a density, potentially overfitting a sample.

Having noted these similarities, even if we view machine learning as a subbranch of nonparametric statistics, there are a few special areas of emphasis that make it a unique tool for applied researchers. I review these here in terms of the general philosophy of machine learning.

Most statistical estimators follow simple algorithms that can be written down in a few textbook equations. Machine algorithms are much more complex, and their implementation can take the form of many steps –ensemble methods worsen this issue by combining multiple algorithms. Computer scientists argue that this approach, which might appear as a series of adhoc fixes compared to standard statistical methods, yields models in which the sum-of-the-parts is superior to individual pieces. The proof of concept is not in terms of proving mathematical results but in terms of actual performance on classic decision problems, from classifying faces and handwritten text, to medical diagnosis or predicting search results. This is a philosophical point of dissent with classical statistics: when evaluating the performance of an approach, computer scientists would tend to focus on practical decision problems in a sample, which includes assessing the performance on a subset of the sample, while, perhaps more ambitiously, statisticians and econometricians would assess performance from generalizable mathematical properties of the sample.

Both approaches have their limitation, and it is neither my expertise nor my purpose to settle debates over mathematics versus simulation. As to using out-of-sample data to evaluate a model, the idea is not new in statistics and parallels bootstrap methods, which use subsampling to characterize properties of a model (Horowitz 2001). Using these methods is usually conceptually easier for models whose properties are not known to the researcher because it simply involves replicating the steps of an estimation on a subsample. Perhaps this democratizes complex econometric problems by sidestepping the need to analytically derive properties of estimators, and using (instead) the out-of-sample data to examine performance or precision of estimates. Furthermore, even in models whose asymptotics are mathematically known, these methods tend to yield more accurate estimates because either the data may not fully satisfy the theoretical requirements or asymptotics are not the best approximation for a finite sample.

However, failing to use any mathematical guidance also comes with its own limitations, some intuitively apparent and other less so. Most machine learning features a large set of hyperparameters that guide the degree of overfitting. It is not computationally feasible to search over the entire space of such parameters, so researchers would use very coarse grids unlikely to select an optimal model or rely on recommendations that were obtained in a different sample. A less visible limitation is that there is often no true clean out-of-sample whose errors are uncorrelated to a subsample. In panel datasets, we would expect both correlations within firms and within periods, and even if such were the only source of correlations, selecting multiple subsamples with separate firms and periods would remove most of the sample.[3] In addition, without any structure, there is no observed out-of-sample that will allow the researcher to assess the accuracy of a model after agents learn from the model and adjust their expectations (Chemla and Hennessy 2019).

In short, the many algorithms used in machine learning have leveraged from growth in computing, which provides a (partial) solution to not knowing the mathematical structure of an estimator. Even considered as algorithms, many standard nonparametric methods suffer from a curse of dimensionality, where adding too many variable or interactions would become impractical in building the model or require too many observations. Unlike standard models, the algorithms will deliver results, admittedly whose quality will be a function of the data, that have appeared robust to many data-generating processes and therefore can be essential tools for practice.

2.2 A brief tutorial

In this section, I will illustrate first steps to a machine learning exercise in a manner that would seem natural and straightforward to any researcher with basic experience in empirical research. Supporting code in Python is provided in the Appendix for random forests. (The code can be adjusted with minimal changes to other common algorithms.) In my experience, over various accounting datasets, random forests consistently rank among the top algorithms while involving a lower computational burden and a less finicky tuning process.

For purposes of better linking to textbooks for readers interested in delving deeper, I will use here the terminology used by computer scientists. To avoid confusion, I map the most commonly used concepts in Table 2 below.

Consider a dataset $(y_t, X_t)_{t \in T}$, where y_t is a response to be predicted and X_t is a vector of features. We are interested in building a model $g^{ML}(.; \theta)$, where θ is

[3] To illustrate, suppose that we (minimally) wish to separate a sample into three subsamples, a training sample to fit the model, a validation sample to select the hyperparamaters of the model, and a test sample to assess performance. The original firms are firms observed over a full time-series. To select these subsamples in a manner that did not imply any time or firm-level correlations, we would have to first divide periods in subsamples 1, 2 and 3 and then subdivide the firms in groups a, b, and c, dividing the entire sample as $1a$, $2b$ and $3c$ but dropping all other subgroups to avoid correlations. Assuming each group is equally sized, this would imply a data loss of $2/3$.

Table 2 Comparative terminology

Statistics	Machine learning
estimator	algorithm, model
observation	example
independent variable, regressor	feature
dependent variable	response, target

a set of (hyper) parameters for this model and that assigns a prediction $g^{ML}(X;\theta)$ for any X.

Step 1: *Partitioning the Sample.* The first step is to partition the sample into three subsamples T_{train}, T_{val} and T_{test}, where T_{train} will (initially) serve to build $g^{ML}(.;\theta)$ for any given θ, T_{val} will serve to assess the best parameters θ^* and T_{test} will be used to evaluate the performance of the estimator. In an *ideal* implementation, the researcher should choose these subsamples such that the errors terms in

$$y_t = g^{ML}(X_t, \theta^*) + \epsilon_t \tag{6}$$

are independent.

In panel data commonly used in accounting research, unfortunately, creating subsamples with independent error terms is infeasible: at the minimum, observations in the panel are likely to feature correlations within-firm or within-time. The greater this correlation, unfortunately, the more the model is likely to fit noise specific to the firm or time and incorrectly report illusory high performance in validation or test samples because it inadvertently fits the correlated errors in these samples.

A partial solution to this problem is to partition the sample by time, with earlier examples assigned to T_{train}, later examples assigned to T_{val} and the most recent examples assigned to T_{test} (Bertomeu et al. 2019; Bao et al. 2019). A researcher using this method in an accounting panel should note that this an imperfect solution that likely favors algorithms more prone to overfitting: it ignores within-firm or within-industry error correlations, and, even between periods, noise terms in a time-series are rarely independent. The main advantage of this method, on the other hand, is to reduce the computational burden by using a single validation sample T_{val} and ensuring (by construction) that future information is not used when predicting over a particular period. A common choice is to set aside 10% of the sample in T_{test} and assign an additional 20% of the remaining examples as validation T_{val}, which ensures that the model is fitted with at least 72% of all examples.

A second approach, widely adopted in computer science and used by Ding et al., is to use k-fold cross-validation. Examples that were not assigned to T_{test} are partitioned into k subgroups, and the performance of the model is measured over k iterations, varying which subgroup is assigned to T_{val}, with overall performance being measured as the average over all assignments. Cross-validation avoids a performance assessment capturing contextual events of the validation sample. For example, when selecting T_{val}, according to the chronological method used earlier and with (often) few periods in the test and validation samples, performance may reflect the particular macro context of these periods rather than generalizable performance in *future* periods in which this model may be used. In addition, k-fold cross-validation can be combined to sampling by firm or by industry, to mitigate the effect of within-firm or within-industry correlations.

While some studies make recommendations about which of these methods is suitable, I am not aware of any theoretical or simulation study providing us with a comparative assessment of each approach. It is nevertheless worth emphasizing that this problem will provide an unfair advantage to methods likely to overfit these errors. Given a large enough dataset and solely for purposes of comparing algorithms, a supplementary test may be to construct T_{train}, T_{val} and T_{test} by requiring that a period or a firm be included only in one group. Unfortunately, this will usually require dropping a majority of examples. Consider a balanced panel $(y_{it}, X_{i,t})$ with N firms and T periods and suppose we partition the sample by allocating a_1 (resp., a_2) firms and b_1 (resp., b_2) periods into the training (validation) sample, which involves dropping all examples with firm or time that already appear in a different group; see Fig. 2.

A good choice of $(a_i, b_i)_{i=1,2}$ is to maximize the fraction of the sample that can be used, by optimizing:

$$\Sigma \equiv a_1 b_1 + a_2 b_2 + (1 - a_1 - a_2)(1 - b_1 - b_2), \tag{7}$$

subject to a constraint that the training sample, used to fit the model and requiring a greater sample size, include a certain fraction of the sample. For example, we might set

$$a_1 b_1 = z_0 a_2 b_2 = z_0(1 - a_1 - a_2)(1 - b_1 - b_2), \tag{8}$$

where $z_0 > 1$ is a constant capturing the size of the training set. This problem can be solved analytically and yields

$$a_1 = b_1 = \frac{2\sqrt{z_0} - z_0}{4 - z_0} \tag{9}$$

$$a_2 = b_2 = \frac{1}{2 + \sqrt{z_0}}. \tag{10}$$

Setting $z_0 = 8$ (which approximately matches the relative size of common training sets) implies that that 58.6% of the firms and time periods are assigned to the training sample, and each of the remaining 20.7% are assigned to validation and test, respectively. Even using this optimal breakdown, a majority of all examples (exactly 57.1%) in the off-diagonal must be dropped.

Step 2: *Tuning the model.* All machine learning algorithms will have a vector of parameters θ that need to be tuned. Studies will often provide guidance as to good choices of parameters, and software implementations of these algorithms will typically provide reasonable default values.

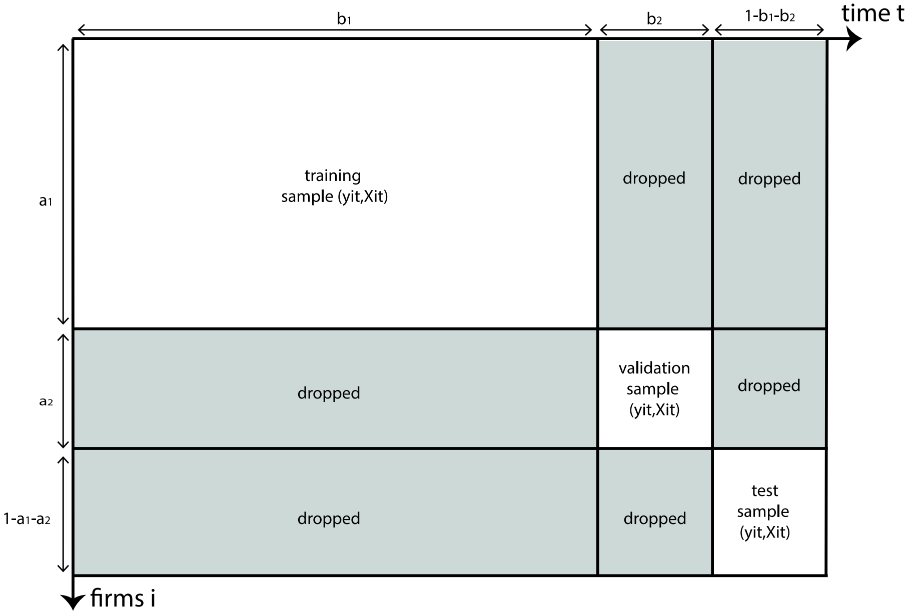

Fig. 2 Partitioning by firm and time

Unfortunately, while some reliance on recommended or default values is unavoidable, this should be done with caution as such guidance is usually obtained from datasets or simulations that have very little in common with accounting datasets. Up to the constraints of computational feasibility, it is therefore desirable to (re)tune as many of the primary parameters as possible.

To tune a model, a set of models $g^{ML}(., \theta_j)$ are built using the examples in T_{train} over a grid $(\theta_j)_{j \in J}$. The performance of each model is then measured using only examples in T_{val}. Two commonly used metrics are the mean-squared error for continuous responses and the area-under-the-curve for binary responses. There are, of course, many other reasonable choices, such as mean absolute error and F1 scores, among others, whose discussion goes far beyond our current purpose, see, e.g., Mohri et al. (2018) for a thorough treatment. The selected parameter θ^* maximizes performance on the grid $(\theta_j)_{j \in J}$.

Step 3: *Building and Assessing the Model.* The model $g^{ML}(., \theta^*)$ is then built with *both* T_{train} and T_{val} to use all examples except those assigned to T_{test}. We can compare the in-sample performance of the model to its out-of-sample performance when used to predict examples in T_{test}. The difference between in-sample and out-of-sample performance can indicate the extent to which an algorithm tends to overfit data.

For binary response models that classify examples, the receiver operating characteristic (ROC) curve aims to graphically represent the trade-off between predicting observations with $y_t = 1$, "true positives", i.e., the fraction of all true positives in the population (recall), versus the rate of false alarms, i.e., the probability that $y_t = 0$ is a predicted as a positive. On its own, an algorithm $g^{ML}(., \theta^*)$ provides a score that captures the propensity of an example to be a true positive. A classification can be made with a cutoff on this score. If the cutoff is arbitrarily low (high), both recall and false alarms will be one (zero) because all examples will be predicted as positives (negatives). The ROC plots the locations of false alarms and recall for any possible classification cutoff. The northwest direction in this plot indicates a better algorithm on both recall and false positives. Decision problems in which inspections are costly or difficult tend favor the west side of the ROC curve as more relevant, while large damages from failing to detect a true positive suggest focusing on the east side of the ROC curve.

In an accounting context, we should keep in mind that neither ROC nor other summary vizualization tools need to be designed to solve a practical problem and ultimately the raw predictions would likely be more helpful to a decision-maker. Ding et al. offer a good illustration of this problem: users of accounting information may not weigh equally all types of errors. An under-estimation of accruals may be more important, especially for a firm in difficulty or facing risks of lawsuits. Similarly, insurance regulators may be targetting a particular buffer of capital so that the insurer does not only accrue future payments on average but also for various stress tests. Hence, without more experimentation on reasonable decision problems and a more general measure of informativeness, Ding et al. cannot conclude that a lower absolute or mean-squared error “improves” accounting estimates.

Step 4: *Interpretation and Analyses.* In addition to context-specific interpretation tools in Section 1.2, there are several additional analyses that can provide additional insight. Most machine learning tools allow the researcher to rank features by how much they contribute to the model. There are, however, some limitations when relying solely on feature importance.

First, the ranking of features does not provide information about the directional association between a feature and the response. In a complex learning model, this association could be positive for certain examples while negative for others, so visualizing the role of a feature is more difficult than in a traditional linear model. While the importance of features can help focus on a subset of the most important features in a dataset with a large number of features, it can be useful to present a correlation matrix with the response, and compare averages and medians of a feature for true positives versus false positives.

Second, a notable problem specific to accounting is that many of the features are likely to be colinear, due to the mechanics accounting process (e.g., assets equate to liabilities and equity, and retained earnings increase by the current period income minus distributions). In my experience, colinearity problems in machine learning can be pernicious because they are common but difficult to systematically diagnose. To give an example, suppose we use assets and market capitalization as two size features, and then we use return on equity and price to earnings ratio. It is easily seen that at least one of these four features is a function of the three others, causing colinearity in the procedure. Unlike linear regression, most machine learning will complete without error, even given perfectly colinear data, churning out various performance metrics and tables of importance of features, because learning will occur by haphazardly picking features and transformations of these features. Even for partial colinearity, the common diagnostic of badly estimated coefficients is not immediately visible, given that the approach does not yield closed-form coefficients or standard errors.

Should the researcher worry about colinearity if the out-of-sample performance of the model is sufficient? Unfortunately, our ability to interpret variables and understand a model is lost with colinearity: variables that capture economic concepts important in the analysis can fall at the bottom of a table of importance if the set of features includes many correlated variables. Below, I make several (simple) practical recommendations that may help ex-ante mitigate this problem.

1. When starting from a kitchen sink approach to the inclusion of features in a model, filter out any feature that can be written as a function of one or more included features.
2. Limit the model to few features capturing size, i.e., variables such as revenue, assets or market capitalization in accounting, and scale other features by one of these features to conceptually separate the effect of size.
3. Consider winsorizing, log transforms, or dividing by standard-deviations or max − min for variables whose histograms cannot be visualized well or that have significant outliers. Features with poor statistical properties can be perceived by an algorithm as near constants with similar problems as colinearity.
4. For groups of variables likely to be colinear even after 1–3, consider assessing importance by summing the importance over groups of variables. For example, even if colinearity between accounting variables makes it difficult to assess the importance of an individual accounting variable, one may assess the cumulative importance of accounting versus market variables. For example, Ding et al. show that, while management estimates are usually more important than other features, the cumulative importance of all business-line features is typically greater.

2.3 Caveats, Tools and Tips

I discuss below a number of additional implementation notes that, while simple, can save time and facilitate the broader use of machine learning tools by reducing technical barriers to entry.

Parallelizing The main obstacle to building a model is the tuning step, which may involve building the model over a very large number of grid points. For example, a five-point grid over three parameters requires at least 125 models –this is a lower estimate given that the search space is unknown and the researcher would have to (ideally) change the grid so that θ^* lies in one of the 27 interior values of this grid! For many algorithms, up to five parameters need to be tuned, and some models, such as neural networks, have a potentially infinite number of layers to tune. Since machine learning is also more computing intensive, tuning can be a major obstacle.

Fortunately, many implementations of common algorithms have embedded parallelization that can leverage all resources of a machine. Typically, even machines that are not designed for machine learning would have multiple cores per CPU and multiple threads (units of execution) per core. To set ideas, an Intel i7-6950X has 10 cores and 20 logical threads for a maximum of 20 workers. Many ensemble learning methods, such as random forests, parallelize very well because the model subcomponent can be built independently. Even for algorithms that do not allow such parallelization, the grid search required to tune the model can be parallelized with a few lines of code. The code provided in the Appendix gives an example of both types of parallelization.

Libraries All common statistical suites include extensive libraries that, when used to their full extent, have sufficient power to build models on standard accounting datasets. The Python library scikit-learn offers one of the most extensive set of choices with reliable high-speed computing and ease of use. However, in my own work, I found implementations in Stata, Matlab and R to have very good performance as well, making it a matter of personal preference. Stata provides better integration with Stata's well-regarded data preparation language and visualization; R benefits from its large user base and a set of algorithms that is as extensive as Python; Matlab integrates better with applications to modelling decision problems.

Comparing algorithms In many applications, multiple algorithms may be potential candidates for a prediction model, and the researcher may be faced with the task of choosing the most suitable algorithm. By and large, more complex "ensemble" algorithms tend to outperform simpler ones but at the cost of higher computing needs and less easily interpretable predictions. Nevertheless, to choose across algorithms, it is possible to think about the algorithm choice as a generalized tuning problem: by comparing performance in a validation sample across algorithms, the researcher can choose the best model. This method comes with a caveat, as (because fitting is optimized on the validation sample) it will give an advantage to algorithms more prone to overfitting. Indeed, for a fair comparison to models that do not overfit, it is also useful to compare the performance of the best algorithms to algorithms with very little overfitting (i.e., whose performance in the training sample is similar to performance on the test sample) and consider supplementary tests along the lines of Fig. 2 .

Contamination The purpose of the research agenda on machine learning should not be to show to machine learning performs better than traditional methods; indeed, knowing when machine learning is inferior to simpler interpretable methods also offers a contribution. However, it is clear that researchers using machine learning may be, even unconsciously, biased to favor machine learning. To avoid this, I recommend strict (documented) research procedures to select the test samples *before* any assessments of performance are known. In fact, ideally, a secondary test sample should be revisited based on new data after a study is complete.

Classification versus Regression Many algorithms come with two versions: classifiers designed for discrete responses and regression algorithms designed for continuous responses (of the form used by Ding et al.). For binary response variables, both models will imply a score which can be mapped to probabilities of true positives and form of cutoffs and classify examples, so the difference between the approaches is usually more subtle than just assigning a method as a function of the nature of the data. In my experience, I found that classification methods work better when evaluating based on area-under-the-curve and F1, i.e., performance metrics that organize examples within (reasonably) balanced samples. However, regression methods when trying to accurately predict the probabilities of true positives or true negatives for more rare and unusual events. To summarize, the choice between methods is not solely a function of the type of data but also a function of the intended use of the predictions.

3 Perspectives for future accounting research

As it is uncontroversial that the tools of machine learning are revolutionizing empirical research, I shall dedicate the final parts of my discussion to opportunities for future research that continue the research path opened by Ding et al..

(i) *Embracing the training, validation and test mindsets in empirical research.* A common criticism of empirical research is that, because there are too many degrees of freedom in choosing the empirical model to be estimated (fixed effects, included or omitted variables, specification, etc.), the asymptotic distributions of conventional test statistics are invalid and provide far less than the claimed 1% or 5% confidence levels. This model multiplicity is inherently an overfitting problem that can be solved using the tools of machine learning. Using validation and test samples to choose which model to estimate, even when restricting to classes of interpretable linear models (i.e., which variables should be included, which fixed effects should be included, etc.) would standardize the procedure of model selection and reduce discretion in picking results consistent with a theory. A researcher using this approach would discriminate across all subsets of plausible models using a training and validation sample and proceed to further analyse the chosen model in a test sample.

(ii) *Better controls.* Testing a theory requires good controls for alternate mechanisms. However, standard methods restrict the number of variables that can be included in a research design, leading to disputes about chosen controls, adoption of particular controls in an ad-hoc manner because these were included in another study without (in the first place) particularly good reasons. In addition, linear specifications of controls exclude a large set of possible interactions that could reflect real world settings. Using machine learning can allow researchers to use a more robust procedure to include a large set of controls and their interactions and therefore better remove variation that is not related to a variable of interest.

(iii) *Enriching classic paradigms in accounting: earnings management and conservatism.* Many models of earnings management rely on extracting residuals from accruals after controlling for economic factors assumed to drive accruals but not managerial discretion (Dechow et al. 2010). The specification of these models has led to a proliferation of earnings management models and because simple models do not fit accruals well, the magnitudes of residuals seem too large for discretion to be a first-order effect. We can use machine learning to learn from a large set of controls and offer more accurately estimated residuals.

In the context of research on conservatism, commonly used empirical models measure asymmetry in reporting in terms of only few economic variables (Watts 2003), implying that machine learning may help better dissociate conservatism from other correlated economic factors.

(iv) *Re-discovering the missing link with theory.* Although many researchers probably do not wish admit it, the current testing method in which a researcher chooses a research design to "test" a theory is problematic, given incentives to validate a theory (or, at least, to organize a set of results along the predictions of a particular theory). This problem contaminates any perspective in which theory comes before the empirical model but, in the end, is considered at such a stylized level that it does not organize the empirical model. Unless one is willing to embrace the structural approach used in the hard sciences (where theories are not simply taken as directional predictions), machine learning offers a completely different solution. The machine learning exercise need not organize data without reference to theory and therefore is not contaminated by a goal to validate a theory. Yet, by reporting over important features and their interactions, it can provide insight as to which theories speak to a feature likely to explain a sample. In short, machine learning offers an approach in which evidence comes before theory.

(v) *Credible data mining in laboratory, field and natural experiments.* Registered reports have started to grow in accounting but presents unique challenges in requiring researchers to "pre-commit" to tests in settings such as field or natural experiments. But how do we measure over-fitting due to pre-commitments that are too broad, or, vice-versa, how can researchers form pre-commitments without access to data –or wouldn't priors formed from other studies themselves imply possible over-fitting? Machine learning can offer an entirely different perspective: by learning *all* possible interactions and using a test sample to assess the true existence of these interactions, it can provide for a data mining that is not only extensive enough to find all patterns but also can provide the tools to test whether these patterns are specific to the training sample or are present in test samples.

(vi) *Better financial ratios*. Financial ratios and multiples have proliferated in financial statement of analysis, leading practitioners to debate the suitable set of variables that best summarize the health and prospects of a business. Which indicators better summarize the state of a company? Progress in machine learning can help select information and design better ratios. Indeed, this research agenda further hints at systematic ways in which we might redesign the information provided in accounting numbers along the lines suggested by Lev and Gu's *End of Accounting*: which information should be given to investors and how should it be organized in accounting reports?

Acknowledgments I thank Xuan Peng and Ting Sun for being extremely patient and offering many of the critical insights that ultimately led to this discussion, and, especially, conducting additional analyses for purposes of discussion. I also thank Edwige Cheynel, Iván Marinovic, Eric Floyd, Wenqiang Pan, and Allan Timmerman for many fireside discussions that helped mature the ideas contained here and most gratefully thank Joey Engelberg for creating the discussion group that awoke my interest in machine learning.

Appendix: Random Forests with scikit-learn

```
# loads common packages
import pandas as pd
import numpy as np
import random
import time
import matplotlib.pyplot as plt
import os
from sklearn.ensemble import RandomForestClassifier
from sklearn import metrics
import scikitplot as skplt
import multiprocessing
from joblib import Parallel, delayed

# loads dataset, with identifiers gvkey-year, all features and response
dataset = pd.read_csv(’dataset.csv’)

# creates chronological training, validation and test samples
nbobs=len(dataset)
for i in range(nbobs):
    if dataset.at[i,’year’]<2010:
        dataset.at[i,’fold’]=1;
    if 2010<=dataset.at[i,’year’]<=2011:
        dataset.at[i,’fold’]=2
    if dataset.at[i,’year’]>2011:
        dataset.at[i,’fold’]=0
dataset_train=dataset[(dataset[’fold’]==1)]
Xtrain=dataset_train.drop(columns=[’gvkey’,’year’,’response’,’fold’])
Ytrain=dataset_train[’response’]
dataset_val=dataset[(dataset[’fold’]==2)]
Xval=dataset_val.drop(columns=[’gvkey’,’year’,’response’,’fold’])
Yval=dataset_val[’response’]
dataset_test=dataset[(dataset[’fold’]==0)]
Xtest=dataset_test.drop(columns=[’gvkey’,’year’,’response’,’fold’])
Ytest=dataset_test[’response’]

# defines cross-validation
def crossval(tun1,tun2,tun3):
    global Xtrain,Ytrain,Xval,Yval
    m = RandomForestClassifier(n_estimators=tun1,min_samples_leaf=tun2,max_features=tun3,n_jobs=32)
    m.fit(Xtrain, Ytrain)
    auc_train=metrics.roc_auc_score(Ytrain,m.predict_proba(Xtrain)[:,1])
    auc_val=metrics.roc_auc_score(Yval,m.predict_proba(Xval)[:,1])
    return [tun1,tun2,tun3,auc_train,auc_val]
def crossval_full(tun1):
    CROSSTAB=[]
    for tun2 in range2:
        for tun3 in range3:
            CROSSTAB.append(crossval(tun1,tun2,tun3))
    return CROSSTAB


# tunes hyperparameters (parallelized)
range1=[100,200,300,400,500,600]
range2=[5,10,15,20,40]
range3=[’auto’,5,10,20,30]
if __name__ == ’__main__’:
    results = Parallel(n_jobs=3)(delayed(crossval_full)(tun1) for tun1 in range1)
results = sum(results, [])
```

```
allresults=pd.DataFrame(data=results, columns=["tun1","tun2","tun3","AUC Train", "AUC Validation"])
allresults.to_csv("allresults.csv", index = False)
allresults = pd.read_csv("allresults.csv")
bestalgo=max(allresults[’AUC Validation’])
for i in range(len(allresults[’AUC Validation’])):
    temp=i
    if allresults.at[i,’AUC Validation’]==bestalgo:
        break
bestalgo=allresults.loc[temp]

# builds model
m =  RandomForestClassifier(n_estimators=int(bestalgo[0]),min_samples_leaf=int(bestalgo[1]),
max_features=int(bestalgo[2]),n_jobs=30)
X=Xtrain.append(Xval)
Y=Ytrain.append(Yval)
m.fit(X, Y)

# ranks most important features
importances = m.feature_importances_
tab_feat = []
for i in range(len(X.columns)):
    tab_feat.append([X.columns[i],m.feature_importances_[i]])
importances=pd.DataFrame(data=tab_feat, columns=["features","relative importance"]).sort_values([’relative␣
↪importance’],ascending=False)
importances.to_csv("importances.csv", index = False)

# plots ROC curve and confusion matrix
Ytest_prob=m.predict_proba(Xtest)[:,1]
Y_prob=m.predict_proba(X)[:,1]
print(metrics.roc_auc_score(Ytest,Ytest_prob))
print(metrics.roc_auc_score(Y,Y_prob))
metrics.plot_roc_curve(m, Xtest, Ytest)
ax = plt.gca()
metrics.plot_roc_curve(m, X, Y,ax=ax)
plt.savefig("roc.png")
metrics.plot_confusion_matrix(m,X,Y,normalize=’true’)
plt.savefig("confusion_train.png")
metrics.plot_confusion_matrix(m,Xtest,Ytest,normalize=’true’)
plt.savefig("confusion_test.png")
fpr, tpr, thresholds = metrics.roc_curve(Y, Y_prob)
roctab=pd.DataFrame(data=np.stack([fpr,tpr,thresholds],axis=1), columns=["false positives","true positives","thresholds"])
roctab.to_csv("roctab_train.csv", index = False)
fpr, tpr, thresholds = metrics.roc_curve(Ytest, Ytest_prob)
roctab=pd.DataFrame(data=np.stack([fpr,tpr,thresholds],axis=1), columns=["false positives","true positives","thresholds"])
roctab.to_csv("roctab_test.csv", index = False)
roctab=pd.DataFrame(data=np.stack([fpr,tpr,thresholds],axis=1), columns=["false positives","true positives","thresholds"])

# saves predictions
Yall=m.predict_proba(dataset.drop(columns=[’gvkey’,’year’,’response’,’fold’]))[:,1]
dataset[’pred’]=Yall
dataset.to_csv("dataset_pred.csv", index = False)
```

References

Bao, Y., Ke, B., Li, B., Yu, Y.J., Zhang, J. (2019). Detecting accounting fraud in publicly traded us firms using a machine learning approach. Available at SSRN 2670703.

Barth, M.E., Li, K., McClure, C. (2019). Evolution in value relevance of accounting information.

Bertomeu, J., Beyer, A., Taylor, D.J. (2016). From casual to causal inference in accounting research: The need for theoretical foundations. *Foundations and Trends in Accounting*, *10*(2-4), 262–313.

Bertomeu, J., Cheynel, E., Floyd, E., Pan, W. (2019). Using machine learning to detect misstatements. Available at SSRN 3496297.

Binz, O., Katherine, S., Stanridge, K. (2020). What can analysts learn from artificial intelligence about fundamental analysis?.

Chemla, G., & Hennessy, C.A. (2019). Rational expectations and the paradox of policy-relevant natural experiments. Journal of Monetary Economics.

Dechow, P.M., & Skinner, D.J. (2000). Earnings management: Reconciling the views of accounting academics, practitioners, and regulators. *Accounting Horizons*, *14*(2), 235–250.

Dechow, P., Ge, W., Schrand, C. (2010). Understanding earnings quality: A review of the proxies, their determinants and their consequences. *Journal of Accounting and Economics*, *50*(2-3), 344–401.

Deng, H. (2019). Interpreting tree ensembles with intreess. *International Journal of Data Science and Analytics*, *7*(4), 277–287.

Ding, K., Lev, B., Peng, X., Sun, T., Vasarhelyi, M.A. (2019). Machine learning improves accounting estimates. Review of Accounting Studies, forth.

Elliott, G., & Timmermann, A. (2013). Handbook of economic forecasting. Elsevier.

Gerakos, J.J., Richard Hahn, P., Kovrijnykh, A., Zhou, F. (2016). Prediction versus inducement and the informational efficiency of going concern opinions. Available at SSRN 2802971.

Gu, Z., & Wu, J.S. (2003). Earnings skewness and analyst forecast bias. *Journal of Accounting and Economics*, *35*(1), 5–29.

Horowitz, J.L. (2001). The bootstrap. In *Handbook of econometrics*, (Vol. 5 pp. 3159–3228): Elsevier.

Hugon, A., Kumar, A., Lin, A.-P. (2016). Analysts, macroeconomic news, and the benefit of active in-house economists. *The Accounting Review*, *91*(2), 513–534.

Li, F. (2010). The information content of forward-looking statements in corporate filings-a naïve bayesian machine learning approach. *Journal of Accounting Research*, *48*(5), 1049–1102.

Mohri, M., Rostamizadeh, A., Talwalkar, A. (2018). Foundations of machine learning. MIT Press.

Pagan, A., & Ullah, A. (1999). Nonparametric econometrics. Cambridge University Press.

Perols, J. (2011). Financial statement fraud detection: An analysis of statistical and machine learning algorithms. *Auditing: A Journal of Practice & Theory*, *30*(2), 19–50.

Perols, J.L., Bowen, R.M., Zimmermann, C., Samba, B. (2017). Finding needles in a haystack: Using data analytics to improve fraud prediction. *The Accounting Review*, *92*(2), 221–245.

Sun, T. (2019). Applying deep learning to audit procedures: An illustrative framework. *Accounting Horizons*, *33*(3), 89–109.

Wager, S., & Athey, S. (2018). Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, *113*(523), 1228–1242.

Watts, R.L. (2003). Conservatism in accounting part ii: Evidence and research opportunities. *Accounting Horizons*, *17*(4), 287–301.