Int Adv Econ Res (2021) 27:1–16
https://doi.org/10.1007/s11294-021-09815-6

An Introduction to Machine Learning for Panel Data

James Ming Chen[1,2]

Published online: 4 March 2021

Abstract Machine learning has dramatically expanded the range of tools for evaluating economic panel data. This paper applies a variety of machine-learning methods to the Boston housing dataset, an iconic proving ground for machine learning. Though machine learning often lacks the overt interpretability of linear regression, methods based on decision trees score the relative importance of dataset features. In addition to addressing the theoretical tradeoff between bias and variance, this paper discusses practices rarely followed in traditional economics: the splitting of data into training, validation, and test sets; the scaling of data; and the preference for retaining all data. The choice between traditional and machine-learning methods hinges on practical rather than mathematical considerations. In settings emphasizing interpretative clarity through the scale and sign of regression coefficients, machine learning may best play an ancillary role. Wherever predictive accuracy is paramount, however, or where heteroskedasticity or high dimensionality might impair the clarity of linear methods, machine learning can deliver superior results.

Keywords Machine learning · Bias-variance tradeoff · Decision trees · Random forests · Extra trees · XGBoost · Learning ensembles · Boosting · Support vector machines · Neural networks

JEL C18 · C23 · C33 · C45 · R31

Introduction

Perhaps no task is more prevalent, or more useful, in economics than the prediction of a numerical value through panel data. By far the most popular tool is linear regression via

✉ James Ming Chen
chenjame@law.msu.edu

[1] Justin Smith Morrill Chair in Law, Michigan State University, East Lansing, MI, USA

[2] Silver Leaf Capital LLC, New York, NY, USA

the ordinary least squares (OLS) method. The scale and sign of coefficients, along with *p*-values, *t*-statistics, and confidence intervals, communicate valuable information among economists.

Though widely and readily understood, linear regression may not provide the most accurate predictions from panel data. This paper introduces basic machine-learning methods. Many machine-learning methods use decision trees to divide data, variable by variable. Ensembles of decision trees harness the Delphic wisdom of numerous miniature predictors. Boosting combines weak learners into a stronger, more accurate predictor. Data suitable for trees and forests can also enable regression through support vector machines and neural networks.

Machine-learning methods lack the overt interpretability of linear regression. Tree- and forest-based methods offset the opacity of these black boxes by scoring the relative importance of dataset features. This paper will address the bias-variance tradeoff as well as the importance of training, validation, and reserving a holdout dataset for testing. Machine learning also sheds light on the primacy of data over algorithms and the wisdom of retaining all outliers. Where interpretability remains paramount, machine learning can support traditional regression methods. Machine learning excels in settings emphasizing predictive accuracy.

Data: The Boston Housing Study

This overview of machine learning revisits Harrison and Rubinfeld's (1978) effort to predict housing prices in Boston's 506 census tracts. A popular proving ground for machine learning (Miller 2015), the Boston housing dataset is included in the SciKit-Learn (Python 2021) package. Table 1 summarizes that dataset.

Splitting and Scaling

Supervised machine learning requires the splitting of data into randomized subsets for training and testing. This practice, rare in conventional economics, ensures that machine learning does not merely memorize values associated with data to be predicted (Müller and Guido, 2017, pp. 17–18). Holding out 25% (a typical if arbitrary proportion) of the data ensures the generalizability of any supervised learning method to data not seen during training (Müller and Guido, 2017, pp. 17–18).

Many machine-learning algorithms perform more accurately on scaled data (Müller and Guido, 2017, pp. 134–142). Standard scaling ensures that machine learning evaluates all variables and reports results in terms of Gaussian *z*-scores. Critically, the scaling of test data must proceed according to the distribution of values in the training data in order to prevent data leakage (Müller and Guido, 2017, pp. 138–140).

As Table 2 shows, OLS regression allows a linear model to be expressed and interpreted in closed form. Accuracy, as measured by r^2 for predictions in the Boston

Table 1 Statistical summary of Boston housing dataset variables

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT	PRICE
Count	506	506	506	506	506	506	506	506	506	506	506	506	506	506
Mean	3.6135	11.3636	11.1368	0.0692	0.5547	6.2846	68.5749	3.7950	9.5494	408.2372	18.4555	356.674	12.6531	22.5328
Std dev	8.6015	23.3225	6.8604	0.2540	0.1159	0.7026	28.1489	2.1057	8.7073	168.5371	2.1649	91.2949	7.1411	9.1971
Min	0.0063	0	0.46	0	0.385	3.561	2.9	1.1296	1	187	12.6	0.32	1.73	5
25%	0.0820	0	5.19	0	0.449	5.8855	45.025	2.1002	4	279	17.4	375.378	6.95	17.025
50%	0.2565	0	9.69	0	0.538	6.2085	77.5	3.2074	5	330	19.05	391.44	11.36	21.2
75%	3.6771	12.5	18.1	0	0.624	6.6235	94.075	5.1884	24	666	20.2	396.225	16.955	25
Max	88.9762	100	27.74	1	0.871	8.78	100	12.1265	24	711	22	396.9	37.97	50

Source: Boston housing dataset on SciKit-Learn (Python 2021)

Table 2 OLS model of the Boston housing dataset (based on a train/test split)

Variable	Beta coefficient	Significance: p <0.001: ***; 0.01: **; 0.05: *; 0.10, +:
CRIM	−0.120264	**
ZN	0.150448	***
INDUS	0.029518	
CHAS	0.074704	*
NOX	−0.280434	***
RM	0.221709	***
AGE	0.021906	
DIS	−0.352755	***
RAD	0.299396	***
TAX	−0.202809	*
PTRATIO	−0.239119	***
B	0.063051	+
LSTAT	−0.452595	***

Source: Author’s own calculations based on data from the Boston housing dataset on SciKit-Learn (Python 2021)

housing dataset, is quite respectable for traditional regression: 0.716806 for the training set and 0.778941 for the test set. Most variables are statistically significant.

Distinct Statistical and Machine-Learning Cultures

Linear regression is by far the most popular method for evaluating panel data. The dominant statistical culture giving rise to this method assumes that data stem from a specific type of stochastic model (Breiman 2001). Machine learning represents a competing algorithmic culture (Breiman 2001). The suspension of assumptions regarding the generation and distribution of data opens the door to algorithms beyond generalized linear methods (Breiman 2001). The algorithmic culture seeks greater accuracy and deeper understanding of data at any scale.

The no-free-lunch theorem holds that it is impossible to know in advance which machine-learning model is best suited to a particular dataset (Wolpert 1996). Consequently, the most practical approach lies in applying as many methods as feasible. Though economics has been slow to accept machine learning, economists should draw liberally from either side of the cultural divide between statistical and algorithmic traditions (Athey and Imbens, 2019).

Because machine-learning alternatives to linear regression are so easily implemented, the practical case for combining statistical and algorithmic methods becomes even more compelling. Panel data, once rendered in a two-dimensional format compatible with Excel or statistical software such as Stata or SPSS, can be exported as comma-separated values (CSV). Data in CSV format can be imported into Python and put immediately to work, with minimal preprocessing, in every machine-learning model evaluated in this paper.

Dendrological Methods: Decision Trees and Forests

The classification and regression tree (CART) algorithm supports a dazzling constellation of methods (Breiman et al., 1984; Loh 2008). Decision trees and their stochastic ensembles (or forests) may be known by the fanciful name, dendrological machine learning.

Because its predictions are not rigidly linear, dendrological machine learning often outperforms linear regression. All dendrological algorithms are robust in the presence of outliers. These algorithms are also quite forgiving of misspecified models. The inclusion of weakly predictive or even non-predictive variables rarely weakens a decision tree or forest ensemble.

Bifurcating the data according to values for each independent variable generates a decision tree predicting the average price per house in each of Boston's 506 census tracts. This basic machine-learning model instantly improves r^2 relative to the OLS baseline. r^2 for the training set improves from 0.716806 to 0.920483. More importantly, test set r^2 improves by nearly +0.10 from 0.778941 to 0.876399.

However, dendrological methods are incompatible with conventional tests of statistical significance. Machine-learning theorists debate the relative merits of less accurate but readily interpreted white box models and more accurate but heuristically opaque black box models (Rudin 2019). Methodological diversity generates a subtler gray spectrum of solutions offering different mixtures of accuracy and interpretability (Pintelas et al., 2020). In practice, different applications will demand blends of white and black box models (Loyola-González 2019).

Decision trees and ensembles of trees do quantify the contribution of each predictive variable. Tree-based methods in SciKit-Learn report feature importances, a vector of values reporting each regressor's contribution to the model's predictions (Géron 2019, pp. 198–199). Feature importances represent "a weighted average, where each node's weight" in a decision tree or across all trees in a forest "is equal to the number of training samples that are associated with it" (Géron 2019, p. 198). Like any other vector of probabilities, their sum is always 1.

Feature importances most closely resemble standardized regression coefficients (or beta coefficients) in conventional statistics (Newman and Browner, 1991), whose use in causal inference is itself controversial (Greenland et al., 1986). Feature importances do differ in a crucial way. Whereas beta coefficients can be positive, negative, or zero, feature importances are invariably non-negative. Consequently, they convey no information regarding the positive or negative correlation between a predictor and the target variable.

Figure 1 reports feature importances for the core CART model. The number of residents of lower socioeconomic status and the average number of rooms per house account for more than 84% of the predictive power of a basic decision tree within the Boston housing dataset.

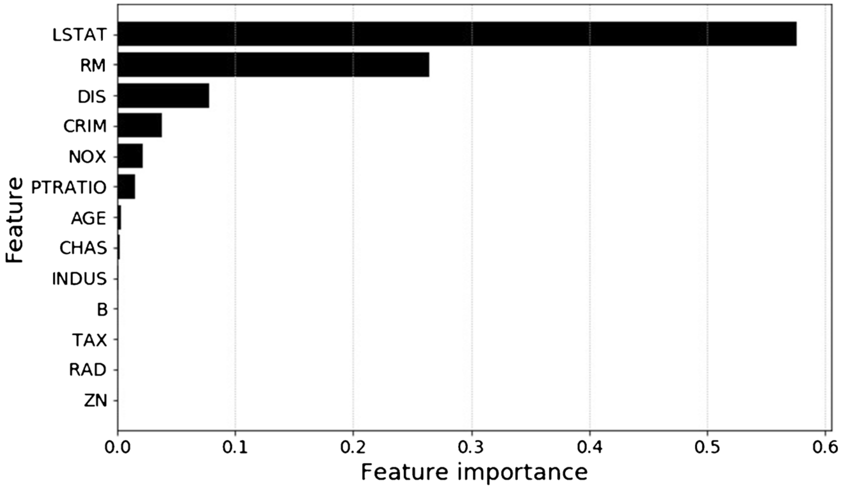

Fig. 1 Feature Importances generated by the CART decision tree algorithm. Source: Author's own calculations based on data from the Boston housing dataset on SciKit-Learn (Python 2021)

The Bias-Variance Tradeoff

A brief interlude on the bias-variance tradeoff serves as a prelude to an exploration of ways to enhance the accuracy of the CART algorithm. The tension between bias and variance arises from an intrinsic property of supervised machine learning. Greater inaccuracy, or bias, in the estimates of model parameters can reduce the variance among parameter estimates across samples (Kohavi and Wolpert, 1996). The bias-variance tradeoff holds the key to the application of machine learning beyond the data on which these algorithms have been trained (Geman et al., 1992). Since it "is impossible to simultaneously achieve the lowest possible variance and bias," the "challenge is to generate a model with (reasonably) low variance and low bias" as the approach "most likely to generalize well to external sets" (Dankers et al., 2019, p. 107).

Bias refers to a method's overall accuracy, particularly in training. Excessive bias yields a model that underfits its data. As often happens with polynomial variants of generalized linear methods, highly accurate models do not provide reliable results unless they generalize well to new, unseen data. High-variance models tend to overfit training data. Therefore, variance affects the generalizability and consistency of results with new data. At optimal complexity, a model strikes the ideal balance between underfitting and overfitting data.

Hyperparameter Tuning

Most machine-learning models reconcile bias and variance through hyperparameter testing. Hyperparameters set the rate at which a machine-learning model learns or the number of splits within a decision tree. In practice, many machine-learning models offer a daunting list of adjustable hyperparameters. If these settings are not properly tuned, a machine-learning model may fall far short of its predictive potential. Ways to explore a potentially vast hyperparameter space include grid search and random search (Müller and Guido, 2017, pp. 267–282).

Training, Validation, and Test Data

Hyperparameter tuning presents a further challenge: Tuning can consume almost all available data. Sufficiently large datasets provide the luxury of a three-way split between training, validation, and test subsets. For example, the Modified National Institute of Standards and Terminology (MNIST) dataset of handwritten digits (a vital contributor to optical character recognition) is divided into 60,000 observations for training, 10,000 for validation, and 10,000 for testing (Kussul and Baidyk, 2004). An intermediate validation subset enables machine learning to strike the optimal balance between bias and variance before optimized hyperparameters are applied to the final holdout subset of test data.

With 506 observations, the Boston housing dataset is relatively small. One way to test different hyperparameters without contaminating the training process is *k*-folds cross-validation (Müller and Guido, 2017, pp. 258–267). The division of training data into *k* subsets enables the use of each of those folds as a synthetic validation set without data leakage.

Ensemble and Boosting Methods

Bagging and Pasting

The simplest way to diversify the results of a decision tree is to sample training instances, either with or without replacement. Bagging, short for bootstrap aggregation, samples with replacement (Breiman 1996; pasting samples without replacement, Breiman 1999). Because $1/e$ of any dataset will escape sampling even if an infinite number of samples are drawn (Géron 2019, p. 195 & footnote 6), the out-of-bag subset of training instances not chosen in bagging provides additional validation of a decision tree's generalizability to previously unobserved data.

Bagging improves both the training and the test performance of the decision tree algorithm on the Boston housing dataset. Training r^2 improves from 0.920483 to 0.941659, and test r^2 from 0.876399 to 0.900210. The loss of accuracy in the out-of-bag score relative to the test score, from 0.900210 to 0.854082, counsels some caution in the interpretation of these results.

Random Forests and Extra Trees

Ensemble and boosting methods aggregate numerous decision trees. This broad and diverse class of methods includes random forests, extremely randomized trees (extra trees), adaptive boosting (AdaBoost), gradient boosting, and extreme gradient boosting (XGBoost). All of these methods use the same syntax within SciKit-Learn's application programming interface. Code for one method applies, with slight modifications, to all others.

Random forests may be the simplest of ensemble methods (Ho 1995). Instead of searching for the best feature when splitting a node, random forests search for the best feature within a random subset. They require the tuning of only two hyperparameters: the maximum number of features in a randomized tree, plus the maximum depth of each tree. Randomizing the thresholds for each feature, as opposed to searching for the optimal threshold, yields an even more stochastic algorithm called extremely random trees, or extra trees (Geurts et al., 2006).

Adaptive and Gradient Boosting

Boosting represents a special class of ensembles that combine weak learners into a strong learner (Drucker and Cortes, 1996). Each step in the sequential training of predictors seeks to correct mistakes made by its predecessor (Géron 2019, p. 199).

The AdaBoost algorithm relies upon decision stumps, or decision trees truncated after a single split (Freund and Schapire, 1997). After each training instance, AdaBoost updates weights for each predictor (Freund and Schapire, 1997). Sequential learning makes it difficult to implement AdaBoost through parallel computing and to scale it to larger datasets (Géron 2019, p. 201).

The gradient boosting algorithm also adds predictors sequentially to an ensemble. Rather than adjusting the weights for each instance, as AdaBoost does, gradient boosting fits each new predictor to the previous predictor's residual errors (Breiman 1998a, 1998b; Friedman 2001). Hyperparameters in gradient boosting control the ensemble's learning rate as well as the depth and growth of decision trees within the ensemble (Géron 2019, p. 204).

Machine learning offers many variants of gradient boosting. XGBoost overcomes limits on speed and scalability that have plagued other boosting algorithms (Chen and Guestrin, 2016). Training on random subsamples yields stochastic gradient boosting, which trades higher bias for lower variance and faster training (Friedman 2002).

Support Vector Machines and Neural Networks

Decision trees and ensemble methods do not exhaust the machine-learning arsenal. Though support vector machines and neural networks deserve extensive examination in their own right, this article considers them for a very simple and practical reason. Panel data preprocessed for evaluation by trees, forests, and boosting methods in SciKit-Learn can be fed, with no further modifications, into a support vector machine and a multilayer perceptron.

Support vector machines and neural networks represent two very different approaches to machine learning. Better suited for small- to medium-sized datasets, support vector machines are versatile enough to handle tasks such as classification, error and fraud detection, and even clustering, a form of unsupervised learning beyond the reach of most other supervised methods (Ben-Hur et al., 2001). Neural networks supply the muscle behind ambitious applications of computer vision, natural language processing, reinforcement learning, and robotics. Both methods perform regression tasks easily (Drucker et al., 1997; Murtagh 1991). SciKit-Learn implements two support vector machines and a multilayer perceptron for regression.

Results

CART and bagging delivered test and out-of-bag r^2 scores from 0.854 to 0.900. These basic methods produced a considerable improvement over the r^2 of 0.779 that linear regression attained. Table 3 combines these results with those from more advanced ensemble and boosting methods, as well as results from SciKit-Learn's support vector machine and multilayer perceptron.

In Table 3, the columns of test results are more informative than the training columns. r^2 and the root mean square error (RMSE) scores improve as the models progress in complexity from linear regression through trees and forests and ultimately the multilayer perceptron.

All machine-learning models exceeded the accuracy of the baseline linear regression model. Random forest, extra trees, and XGBoost, typical of ensembles and boosting models, outperformed the basic CART model and its first-order improvement through bagging. The support vector machine and multilayer perceptron were even more accurate.

As impressive as gains of +0.10 to +0.16 in r^2 are, reducing RMSE by 37 to 48% represents an even greater improvement. Relative to OLS, the best machine-learning methods cut each prediction's average error by nearly one-half. Figures 2, 3, 4, and 5 depict improvements along the ladder of model-based complexity from linear regression to the CART decision tree, ensemble learning and boosting, the support vector machine, and the multilayer perceptron.

Table 3 Summary of machine-learning results on the Boston housing dataset

	Training		Test	
Model	r^2	RMSE	r^2	RMSE
Linear	0.716806	0.532160	0.778941	0.525247
Decision tree	0.920483	0.281988	0.876399	0.392754
Bagging	0.941659	0.241539	0.900210	0.352901
Random forest	0.981473	0.136114	0.912558	0.330347
Extra trees	0.997930	0.045500	0.916564	0.322691
XGBoost	0.998408	0.039894	0.912709	0.330061
Support vector machine	0.945452	0.233555	0.924545	0.306870
Multilayer perceptron	0.954146	0.214135	0.939349	0.275124

Source: Author's own calculations based on data from the Boston housing dataset on SciKit-Learn (Python 2021)

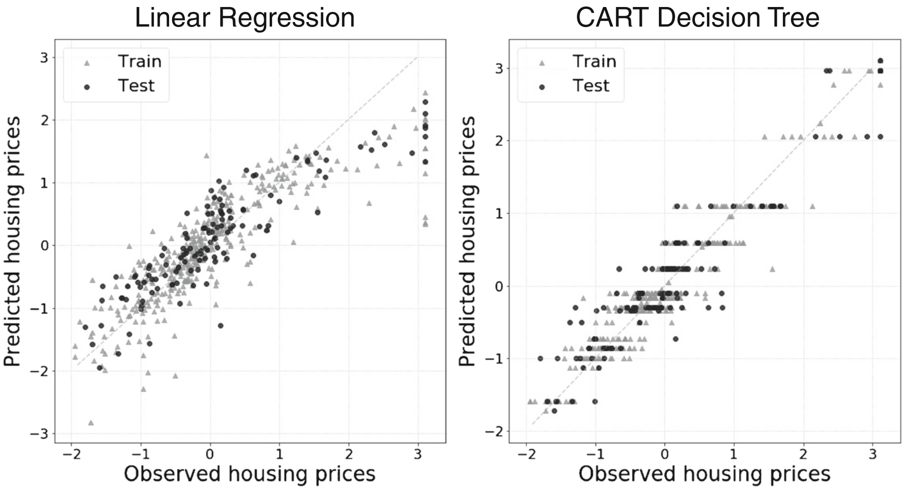

Fig. 2 Boston housing dataset – Linear regression and CART decision tree. Notes: In the linear regression model, Train: $r^2 = 0.716806$ and RMSE = 0.532160. Test: $r^2 = 0.778941$ and RMSE = 0.525247. In the CART decision tree model, Train: $r^2 = 0.920483$ and RMSE = 0.281988. Test: $r^2 = 0.876399$ and RMSE = 0.392754. Source: Author's own calculations based on data from the Boston housing dataset on SciKit-Learn (Python 2021)

Discussion

Vastly improved accuracy alone justifies the application of machine learning to panel data. Dendrological models can be interpreted alongside linear regression. Machine learning's superior handling of data also counsels reconsideration of conventional approaches to outliers.

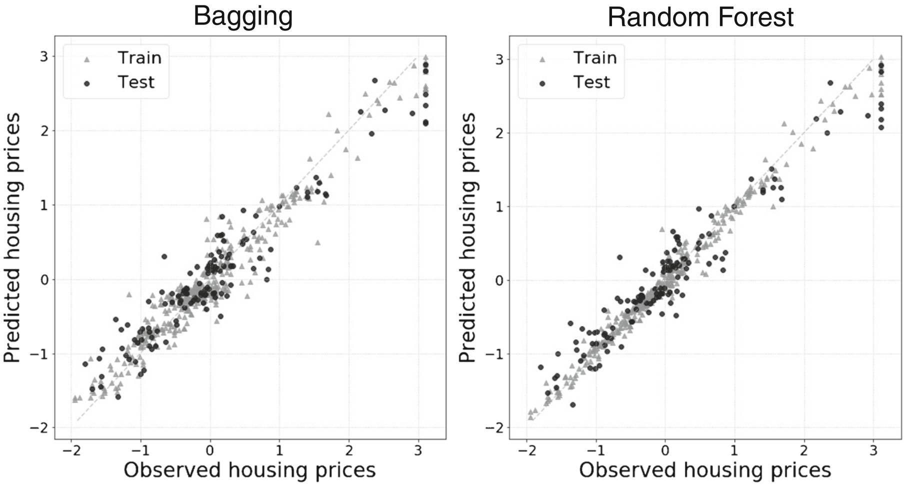

Fig. 3 Boston housing dataset – Bootstrap aggregation (bagging) and random forest. Notes: In the bagging model, Train: $r^2 = 0.941659$ and RMSE = 0.241539. Test: $r^2 = 0.900210$ and RMSE = 0.352901. In the random forest model, Train: $r^2 = 0.981473$ and RMSE = 0.136114. Test: $r^2 = 0.912558$ and RMSE = 0.330347. Source: Author's own calculations based on data from the Boston housing dataset on SciKit-Learn (Python 2021)

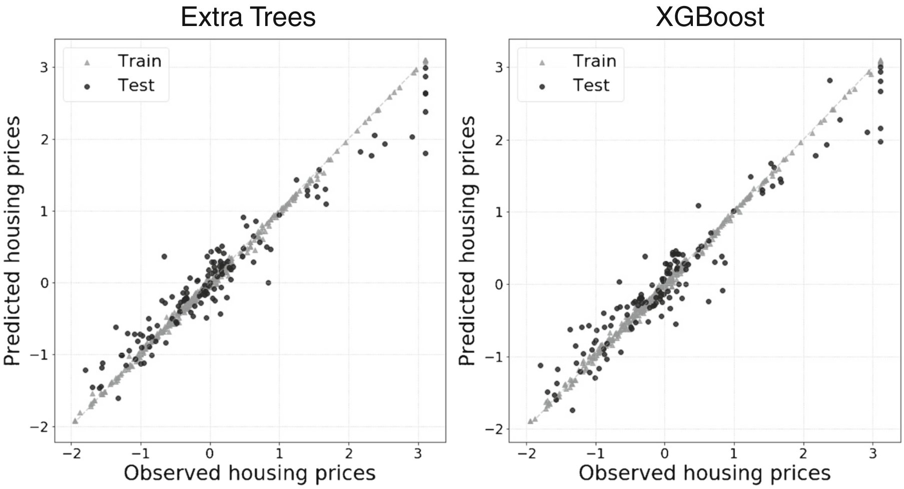

Fig. 4 Boston housing dataset – Extra trees and XGBoost. Notes: In the extra trees model, Train: r^2 = 0.997930 and RMSE = 0.045500. Test: r^2 = 0.916564 and RMSE = 0.322691. In the XGBoost model, Train: r^2 = 0.998408 and RMSE = 0.039894. Test: r^2 = 0.912709 and RMSE = 0.330061. Source: Author's own calculations based on data from the Boston housing dataset on SciKit-Learn (Python 2021)

Interpretability through Feature Importances

Support vector machines and neural networks are black boxes, notoriously hard to express or interpret in terms comparable to the sign and scale of coefficients in a linear or polynomial model. However, random forests, XGBoost, and extra trees report feature importances based on the probability that an independent variable affects a

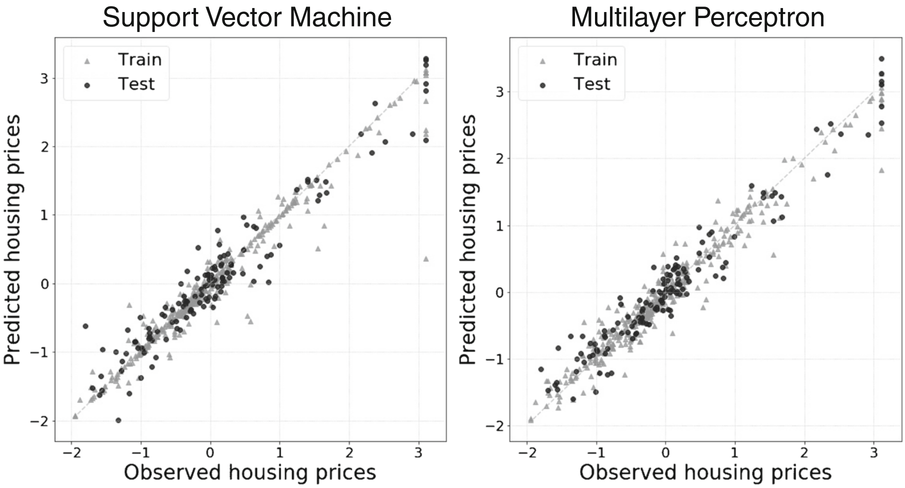

Fig. 5 Boston housing dataset – Support vector machine and multilayer perceptron. Notes: In the support vector machine model, Train: r^2 = 0.945452 and RMSE = 0.233555. Test: r^2 = 0.924545 and RMSE = 0.306870. In the multilayer perceptron model, Train: r^2 = 0.954146 and RMSE = 0.214135. Test: r^2 = 0.939349 and RMSE = 0.275124. Source: Author's own calculations based on data from the Boston housing dataset on SciKit-Learn (Python 2021)

prediction. These vectors can be interpreted, even though they cannot convey additional information embedded in the sign of standardized regression coefficients.

Figures 6, 7, and 8 report the feature importance of three ensemble and boosting models. According to Fig. 6, the random forest model assigns nearly 80% of its predictive weight to the two variables that dominate the CART model: Each census tract's proportion of residents having lower socioeconomic status and the average number of rooms per house. XGBoost, shown in Fig. 7, assigns nearly as much weight (roughly 75%) to those variables. In each of these models, weighted distance to five employment centers trails badly in third place, adding less than 10% of total predictive weight. In contrast, the slightly more accurate extra trees model assigns more balanced feature importances to socioeconomic status and rooms per house. Figure 8 places these weights at roughly 0.30 and 0.26, respectively.

Strikingly, these machine-learning models' feature importances challenge the premises underlying the original Boston housing study. Harrison and Rubinfeld (1978) conjectured that housing prices would reflect the negative impact of air pollution. In feature importances reported by random forests and XGBoost, nitrogen oxide levels as a proxy for pollution lagged behind other variables, scarcely reaching 3% in predictive ability. Extra trees assigned less than 7% in predictive importance to nitrogen oxide.

The contrast between feature importances and linear coefficients casts doubt upon the illusory clarity of OLS regression. The linear model supported the original hypothesis that real estate prices reflect the negative impact of pollution. Feature importances reduce the weight otherwise attributable to this factor. Many experts might agree that average home size and a neighborhood's character, as a thinly disguised euphemism for the presence of poor people, have a greater impact on home prices. To the extent that pollution does affect housing prices, its impact may reflect environmental racism, or the tendency with which pollution is directed toward nonwhite inhabitants (Bullard 2001). Machine learning thus sharpens inferences from more traditional predictive methods and from expert human judgment.

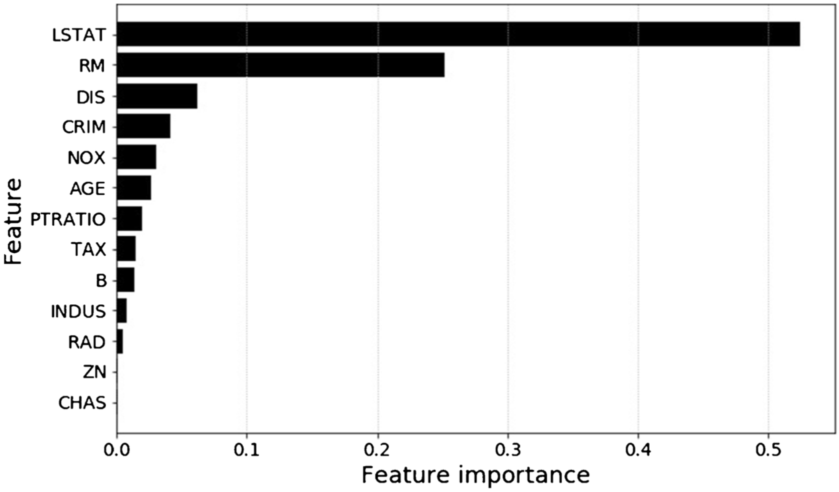

Fig. 6 Feature importances for the random forest model. Source: Author's own calculations based on data from the Boston housing dataset on SciKit-Learn (Python 2021)

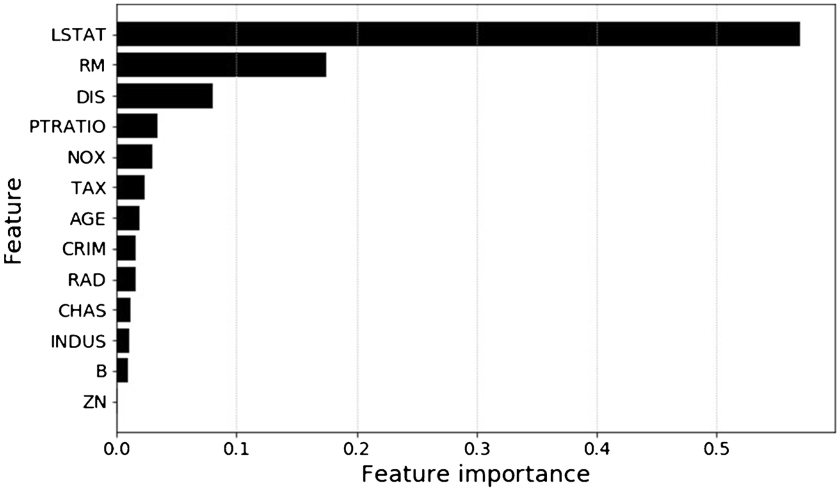

Fig. 7 Feature importances for the XGBoost Model. Source: Author's own calculations based on data from the Boston housing dataset on SciKit-Learn (Python 2021)

Treatment of Outliers in Light of Lessons from Machine Learning

Machine-learning algorithms are not simply more accurate. As Fig. 2 shows, their superior performance is most pronounced among extreme observations. Machine learning outperforms linear regression in predicting prices in Boston's most expensive neighborhoods. Data that traditional methods might otherwise discard as outliers become more tractable.

Traditional statistics uses numerous devices to manage purported outliers. Trimming crudely discards data beyond points presumed to be too extreme (Clarke 1994; Lusk et al., 2011). Slightly less destructive winsorizing clips outliers at an arbitrary level and

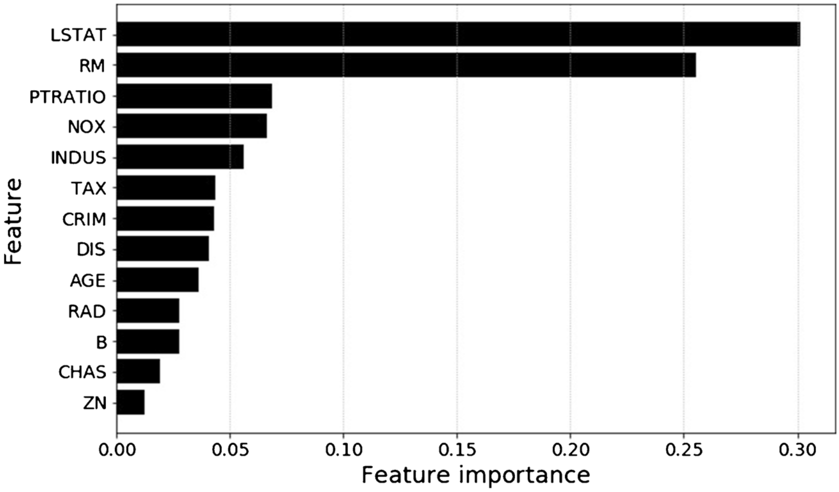

Fig. 8 Feature importances for the extra trees model. Source: Author's own calculations based on data from the Boston housing dataset on SciKit-Learn (Python 2021)

assigns the corresponding minimum or maximum value (Dixon 1960; Hastings et al., 1947; Tukey 1962).

In contrast, the intrinsic robustness of machine learning counsels the retention of all data. Machine learning has revealed "the unreasonable effectiveness of data" (Halery et al., 2009, p. 9). Given sufficient data, very different algorithms attain almost identical results on complex problems such as natural language disambiguation (Banko and Brill, 2001). Performative convergence despite differences in algorithmic complexity suggests the primacy of data over theoretical elaboration and experimental design. "[I]nvariably, simple models and a lot of data trump more elaborate models based on less data" (Halery et al., 2009, p. 9).

A key corollary of the unreasonable effectiveness of data is a systematic preference in machine learning for retaining all data as observed, with neither trimming nor winsorizing. The unreasonable-effectiveness hypothesis neutralizes concerns "about the curse of dimensionality and overfitting of models to data" (Halery et al., 2009, p. 9). Machine learning disfavors the discarding of observations at either extreme, because the phenomena of greatest economic interest "consist[] of individually rare but collectively frequent events" (Halery et al., 2009, p. 9).

Conclusion

Machine learning can dramatically improve accuracy in predictions based on panel data. Models based on decision trees report feature importances that enable these black boxes to be interpreted in ways akin to coefficients and signs in linear regression. Once data have been properly split into training, validation, and test sets and scaled for machine learning, researchers should apply all feasible machine-learning models, without trimming or winsorizing the data.

The primary limitation of machine learning is its lack of interpretive clarity. Unless based on CART or a (boosted) ensemble of decision trees, machine-learning methods are effectively black boxes. Even feature importances generated by dendrological methods cannot convey information associated with the sign of coefficients and correlations in linear models.

The choice between conventional linear methods and machine-learning alternatives hinges on this balance between accuracy and interpretability. This tradeoff in deployment parallels the balance between bias and variance in the tuning of machine-learning models. In applications or circumstances emphasizing predictive accuracy, machine learning may dominate conventional regression.

Datasets that are inherently difficult to interpret within linear models may benefit from immediate resort to machine learning. For instance, highly heteroskedastic data are inherently opaque (Glejser 1969; Rigobon 2003). Machine learning can overcome uneven variability within predictors, especially in time-series forecasting (Hassan et al., 2013). Because even basic methods work well, perhaps even optimally, in high-dimensional settings, machine learning may even transform the curse of dimensionality (Taylor 1993; Trunk 1979) into an affirmative blessing (Gorban and Tyukin, 2018; Gorban et al., 2020).

In contrast, where the sign and scale of regression coefficients matter more than predictive accuracy, machine learning might best play an ancillary role. At a minimum, machine-learning deployments should include at least one generalized linear model and exploratory data analysis so that information such as positive and negative correlations and confidence intervals can be obtained. In these settings, feature importances should be regarded as complements for linear coefficients (whether standardized or not), rather than substitutes.

Ultimately, the difference between cases where generalized linear methods dominate machine learning or vice versa is not mathematical. Rather, the preference for one class of methods versus the other is practical. By and large, OLS and other linear methods offer superior interpretability, while machine learning promises, and typically delivers, superior accuracy.

References

Athey, S., & Imbens, G. W. (2019). Machine learning methods that economists should know about. *Annual Review of Economics, 11*, 685–725.

Banko, M. & Brill, E.D. (2001). Scaling to very, very large corpora for natural language disambiguation. *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, pp. 26–33. https://www.aclweb.org/anthology/P01-1005/

Ben-Hur, A., Horn, D., Siegelmann, H., & Vapnik, V. D. (2001). Support vector clustering. *Journal of Machine Learning Research, 2*, 125–137.

Breiman, L. (1996). Bagging predictors. *Machine Learning, 24*(3), 123–140.

Breiman, L. (1998a). Arcing classifiers. *Annals of Statistics, 26*, 801–824.

Breiman, L. (1998b). Arcing the edge. *Annals of Probability, 26*, 1683–1702.

Breiman, L. (1999). Pasting small votes for classification in large databases and on-line. *Machine Learning, 36*(1), 85–103.

Breiman, L. (2001). Statistical modeling: The two cultures. *Statistical Science, 16*, 199–231.

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

Bullard, R. D. (2001). Environmental justice in the 21st century: Race still matters. *Phylon, 49*(3/4), 151–171.

Chen, T. & Guestrin, C.E. (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. https://doi.org/10.1145/2939672.2939785

Clarke, B. R. (1994). Empirical evidence for adaptive confidence intervals and identification of outliers using methods of trimming. *Australian Journal of Statistics, 36*, 45–58.

Dankers, F. J. W. M., Traverso, A., Wee, L., & van Kuijk, S. M. J. (2019). Prediction modeling methodology. In P. Kubben, M. Dumontier, & A. Dekker (Eds.), *Fundamentals of clinical data science* (pp. 101–120). Cham: Springer.

Dixon, W. J. (1960). Simplified estimation from censored normal samples. *Annals of Mathematical Statistics, 31*(2), 385–391.

Drucker, H., & Cortes, C. (1996). Boosting decision trees. *Advances in Neural Information Processing Systems, 8*, 479–485.

Drucker, H., Burges, C. C., Kaufman, L., Smola, A. J., & Vapnik, V. N. (1997). Support vector regression machines. *Advances in Neural Information Processing Systems, 9*, 155–161.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and its application to boosting. *Journal of Computer and System Sciences, 55*, 119–139.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics, 29*, 1189–1232.

Friedman, J. H. (2002). Stochastic gradient boosting. *Computational Statistics and Data Analysis, 38*, 367–378.

Geman, S., Bienenstock, É., & Doursa, D. (1992). Neural networks and the bias/variance dilemma. *Neural Computation, 4*, 1–58.

Géron, A. (2019). *Hands-on machine learning with SciKit-learn, Keras & TensorFlow: Concepts, tools, and techniques to build intelligent systems*. Sebastopol: O'Reilly.

Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning, 63*, 3–42.

Glejser, H. (1969). A new test for heteroskedasticity. *Journal of the American Statistical Association, 64*, 316–323.

Gorban, A. N., & Tyukin, I. Y. (2018). Blessing of dimensionality: Mathematical foundations of the statistical physics of data. *Philosophical Transactions of the Royal Society A, 376*(2118), 20170237.

Gorban, A. N., Makarov, V. A., & Tyukin, I. Y. (2020). High-dimensional brain in a high-dimensional world: Blessing of dimensionality. *Entropy, 22*(1), 82.

Greenland, S., Schlesselman, J. J., & Criqui, M. H. (1986). The fallacy of employing standardized regression coefficients and correlations as measures of effect. *American Journal of Epidemiology, 123*, 203–208.

Halery, A., Norvig, P., & Pereira, F. (2009). The unreasonable effectiveness of data. *IEEE Intelligent Systems, 24*(2), 8–12.

Harrison, D., & Rubinfeld, D. L. (1978). Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management, 5*, 81–102.

Hassan, M., Hossny, M., Nahavandi, S. & Creighton, D. (2013). Quantifying heteroskedasticity using slope of local variances index. 2013 UKSim 15th International Conference on Computer Modelling and Simulation, pp. 107–111. https://doi.org/10.1109/UKSim.2013.75.

Hastings, C., Mosteller, F., Tukey, J. W., & Winsor, C. P. (1947). Low moments for small samples: A comparative study of order statistics. *Annals of Mathematical Statistics, 18*, 413–426.

Ho, T.K. (1995). Random decision forests. Proceedings of 3rd International Conference on Document Analysis and Recognition, 1, 278–282.

Kohavi, R. & Wolpert, D.H. (1996). Bias plus variance decomposition for zero-one loss functions. ICML '96: Proceedings of the Thirteenth International Conference on International Conference on Machine Learning, pp. 275–283. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.48.4661

Kussul, E., & Baidyk, T. (2004). Improved method of handwritten digit recognition tested on MNIST database. *Image and Vision Computing, 22*(12), 971–981.

Loh, W.-Y. (2008). Classification and regression tree methods. In F. Ruggeri, R. S. Kennet, & F. W. Faltin (Eds.), *Encyclopedia of statistics in quality and reliability* (pp. 315–323). Hoboken: Wiley.

Loyola-González, O. (2019). Black-box vs. white-box: Understanding their advantages and weaknesses from a practical point of view. *IEEE Access, 7*, 154096–154113.

Lusk, E. J., Halperin, M., & Heilig, F. (2011). A note on power differentials in data preparation between trimming and Winsorizing. *Business Management Dynamics, 1*(2), 23–31.

Miller, T. W. (2015). *Marketing data science: Modeling techniques in predictive analytics with R and Python*. Old Tappan: Pearson Education.

Müller, A. C., & Guido, S. (2017). *Introduction to machine learning with Python: A guide for data scientists*. Sebastopol: O'Reilly.

Murtagh, F. (1991). Multilayer perceptrons for classification and regression. *Neurocomputing, 2*, 183–197.

Newman, T. B., & Browner, W. S. (1991). In defense of standardized regression coefficients. *Epidemiology, 2*, 383–386.

Pintelas, E., Livieris, I. E., & Pintelas, P. (2020). A grey-box ensemble model exploiting black-box accuracy and white-box intrinsic interpretability. *Algorithms, 13*(1), 17.

Python (2021). SciKit-Learn library. https://scikit-learn.org/stable/index.html (visited January 14, 2021).

Rigobon, R. (2003). Identification through heteroskedasticity. *Review of Economics and Statistics, 85*, 777–792.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence, 1*, 206–215.

Taylor, C. R. (1993). Dynamic programming and the curses of dimensionality. In C. R. Taylor (Ed.), *Applications of dynamic programming to agricultural decision problems* (pp. 1–10). New York: Westview Press.

Trunk, G. V. (1979). A problem of dimensionality: A simple example. *IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-1*(3), 306–307.

Tukey, J. W. (1962). The future of data analysis. *Annals of Mathematical Statistics, 33*, 1–67.

Wolpert, D. (1996). The lack of *a priori* distinctions between learning algorithms. *Neural Computation, 8*, 1341–1390.