PORTFOLIO MANAGEMENT RESEARCH
By With Intelligence

WINTER 2022 VOLUME 4 NUMBER 1

the journal of financial data science

JFDS.pm-research.com

Investable and Interpretable Machine Learning for Equities

Yimou Li, Zachary Simon, and David Turkington

Yimou (Andrew) Li

Yimou (Andrew) Li is an Assistant Vice President and Quantitative Researcher for the Portfolio Management Research team at State Street Associates. The Portfolio Management Research team collaborates with academic partners to develop new research on asset allocation, risk management, and investment strategy. Andrew's research focuses on leveraging quantitative models and machine learning to tackle investment challenges. Andrew works closely with institutional clients to deliver his research through practical applications, advisory projects, and thought leadership pieces. Andrew received his Bachelor of Science in Applied Mathematics and Economics from Brown University and Master of Finance from MIT.

Zachary Simon

Zachary Simon is a Senior Data Scientist at Polen Capital and a member of the Investment Analytics and Data Science team. The team works directly with investment and data professionals across the company to apply data science, quantitative methods, and advanced analytics to all aspects of the investment and portfolio management process.

Zachary holds an MBA from the MIT Sloan School of Management and a BA in Computer Science and Economics from Colby College.

David Turkington

David Turkington is Senior Managing Director and Head of State Street Associates, State Street Global Markets' decades-long partnership with renowned academics that produces innovative research on markets and investment strategy. Mr. Turkington is a frequent presenter at industry conferences, has published more than 30 research articles in a range of journals, and currently serves on the editorial board of the Journal of Alternative Investments. He is co-author of the books "A Practitioner's Guide to Asset Allocation" and "Asset Allocation: From Theory to Practice and Beyond," and his published research has received the 2013 Peter L. Bernstein Award, four Bernstein-Fabozzi/Jacobs-Levy Outstanding Article Awards, and the 2010 Graham and Dodd Scroll Award. Mr. Turkington graduated summa cum laude from Tufts University with a BA in mathematics and quantitative economics, and he holds the CFA designation.

Investable and Interpretable Machine Learning for Equities

Yimou Li, Zachary Simon, and David Turkington

Yimou Li
is an assistant vice president and machine learning researcher at State Street Associates in Cambridge, MA.
yli24@statestreet.com

Zachary Simon
is a senior data scientist at Polen Capital in Boston, MA.
zsimon@polencapital.com

David Turkington
is senior managing director and head of Portfolio and Risk Research at State Street Associates in Cambridge, MA.
dturkington@statestreet.com

KEY FINDINGS

- The authors argue that for machine learning models to be useful for stock selection, they should be investable, interpretable, and interesting.
- The authors focus on liquid US stocks and calibrate models for a reasonable turnover, use model fingerprint to interpret machine learning logics, and show results that outperform simpler models.
- By adjusting the goal and time horizon of model predictions, the authors evaluate how people can impart to models discretionary knowledge and preferences.

ABSTRACT

The authors propose three principles for evaluating the practical efficacy of machine learning for stock selection, and they compare the performance of various models and investment goals using this framework. The first principle is investability. To this end, the authors focus on portfolios formed from highly liquid US stocks, and they calibrate models to require a reasonable amount of trading. The second principle is interpretability. Investors must understand a model's output well enough to trust it and extract some general insight from it. To this end, the authors choose a concise set of predictor variables, and they apply a novel method called the model fingerprint to reveal the linear, nonlinear, and interaction effects that drive a model's predictions. The third principle is that a model's predictions should be interesting—they should convincingly outperform simpler models. To this end, the authors evaluate out-of-sample performance compared to linear regressions. In addition to these three principles, the authors also consider the important role people play by imparting domain knowledge and preferences to a model. The authors argue that adjusting the prediction goal is one of the most powerful ways to do this. They test random forest, boosted trees, and neural network models for multiple calibrations that they conclude are investable, interpretable, and interesting.

There are few barriers to entry in applying machine learning to stock selection. Most researchers and investors have access to plentiful data, open source code libraries, and ample computing power. Authors such as Rasekhschaffe and Jones (2019), Gu, Kelly, and Xiu (2020), and others[1] have reported incredibly strong

[1] A growing amount of literature has reported strong out-of-sample performance when applying machine learning to asset pricing predictions. Bryzgalova, Pelger, and Zhu (2020), Aldridge and Avellaneda (2019), Fischer and Krauss (2018), and Moritz and Zimmerman (2016), for example, reported superb portfolio returns in backtests when applying machine learning techniques such as neural network and tree-based models to form predictions.

performance for machine learning in hypothetical backtests. Despite such promising results, practitioners must overcome reasonable fears that backtest results do not fully represent the costs of trading, that their advantage will be competed away when such techniques become mainstream, that research results have a positive bias whereby only the best findings are reported, or that historical results reflect data errors or unscrupulous research methods. Moreover, investors may worry that machines could load up on concentrated sources of risk, fail to appreciate structural shifts in markets, or commit some other fatal flaw in reasoning—all of which may go undetected until it is too late. We believe that investment strategies based on machine learning have a lot to offer, but we argue that these strategies are not particularly useful in practice unless they are both investable and interpretable. They must also be interesting, in the sense that they reliably outperform simpler alternatives, but for practitioners, this third condition is only relevant once the first two are met.

To build strategies that are investable, we focus on a subset of liquid securities with large market capitalizations, and we ensure that trading does not exceed a reasonable amount. Therefore, we have little interest in models that mostly predict the returns of small or illiquid stocks or those that exploit short-term pricing effects that are prohibitively costly to trade.

To build strategies that are interpretable, we use a concise set of predictor variables, and we decompose the outputs of complex models into subcomponents using a method called the *model fingerprint*. These components may be compared to other models, theories, and hypotheses, which we argue is one of the most crucial parts of the modeling process.

Machine learning models consume a narrow set of data. They do not have the full range of experience that a person brings to the field of investing, such as intuition from life experience and financial theory. This domain knowledge is just as important as the model's technical specifications. People can inform models by choosing which data to include and by accepting or rejecting a model's logic based on its coherence with other theories and ideas.

Another way to impart knowledge and preference to a model is to adjust its predictive goal. Machine algorithms are laser-focused. If the goal is to predict stock returns in excess of the market index, the model will shift its focus away from market timing and toward stock selection. If the goal is to predict 12-month returns instead of 1-month returns, the model will search for persistent relationships that reduce turnover. In this way, we can direct our model's attention to whichever areas of the market present the best opportunities.

In our empirical analysis, we find compelling results for machine learning applied to stock selection. However, the benefits we show are modest compared to many other research papers, probably because of our emphasis on investability and interpretability. Investors should approach machine learning with reasonable expectations. As competitive markets evolve and adapt, we should not expect the same opportunities for superior risk-adjusted returns to last forever. One is right to be suspicious of results that look too good to be true. The applications we test could be described as intelligent factor investing. They use historical data to derive rules for (linear) factor preferences, nonlinear relationships, interactions among multiple factors, and ways to rescale these effects when there are regime shifts or sector-specific considerations. In our view, the performance enhancement from machine learning is more incremental than revolutionary.

We structure the remainder of the article as follows. First, we describe the data and the model fingerprint that provides interpretability. Next, we discuss our approach to modeling selection, calibration, and training. We present results for investment strategies based on a range of machine learning models, focusing on the key principles that results should be investable, interpretable, and interesting. We then expand

EXHIBIT 1
Prediction Inputs

Type	Category	Variable	Definition	Motivation and Related Literature
Attribute	**Company Value**	1. Size	Market capitalization	Banz (1981)
		2. Value	Price-to-book ratio	Rosenberg, Reid, and Lanstein (1985)
	Past Price Trends	3. Short-Term Mean Reversion	Last month return	Jegadeesh and Titman (1993)
		4. Momentum	Last year return/Last month return	Jegadeesh (1990)
		5. Sector Momentum	Momentum—Average momentum of sector	Moskowitz and Grinblatt (1999)
		6. Long-Term Mean Reversion	Last four-year return/Last year return	Jegadeesh and Titman (1993)
	Riskiness	7. Volatility	Trailing year volatility	Ang et al. (2006)
		8. Beta	Trailing year market beta	Fama and MacBeth (1973)
	Return On Equity	9. Leverage	Assets/Equity	Bhandari (1988)
		10. Profitability	EBITDA/Assets	Balakrishnan, Bartov, and Faurel (2010)
	Operating Model	11. Investment	YoY asset growth	Cooper, Gulen, and Schill (2008)
		12. Dividend Yield	Dividend yield	Litzenberger and Ramaswamy (1982)
	Sector	13. Economic sector	Dummy variables for sector classification	
Regime	**Economic Regimes**	14. Financial Turbulence	Mahalanobis distance of sector returns	Kritzman and Li (2010)
		15. Recession Likelihood	KKT index built from macro variables	Kinlaw, Kritzman, and Turkington (2021)
		16. Recession Likelihood (Shift)	Standardized one-year change in KKT index	Kinlaw, Kritzman, and Turkington (2021)

on these results by changing the prediction goal from total returns to excess returns and from short-term returns to longer-term returns. The final section concludes.

DATA AND METHODOLOGY

Our ultimate goal is to build portfolios. To do that, we use machine learning to forecast returns. We then form portfolios of stocks with the highest or lowest forecasted returns. With this two-step process, we keep portfolio construction simple and interpretation straightforward. Our predictions come from supervised learning, in which we define inputs (X) and a target (Y). We structure the problem as a panel regression, in which a given model is tasked with predicting every stock at every point in time.

Prediction Inputs

Our investment universe consists of stocks in the S&P 500 between December 1992 and September 2020. We limit our attention to this set of companies because they represent the most liquid and accessible securities in one of the largest equity markets. We consider a concise list of predictive variables based on theory and intuition. As shown in Exhibit 1, the predictors fall into two groups: company-level attributes, which are measured for every company every month, and regime indicators, which are measured once per month. These inputs align with the type of data a human analyst or traditional regression model might use.

Security attributes are based on data from Refinitiv, and regime variables are obtained from State Street Global Markets. We include the economic sector of each company, based on the Global Industry Classification System (GICS), using a collection of dummy variables: They assume a value of 1 for companies in that sector and zero otherwise, which allows the models to favor certain factors in some industries and not others. To prevent outliers from distorting the results, we transform all of the attribute values (other than the sector indicators) into cross-sectional ranks.

Prediction Target

Our base-case model predicts each stock's total return for the following month. Later, we present results in which we vary the prediction target. First, we introduce various benchmarks and target excess returns. We consider the capital asset pricing model (CAPM; Sharpe 1964) as well as a six-factor model composed of the Fama and French (2015) five factors plus momentum (Jegadeesh 1990). We create these target returns by regressing each stock's trailing one-year returns on the returns of the relevant factors and recording the excess return (regression intercept plus residual) for the latest data point. Second, we adjust the time horizon from 1 month to 12 months.

Model Selection and Calibration

In order of increasing complexity, the models we test are ordinary least squares (OLS) linear regression, least absolute shrinkage and selection operator (LASSO), random forest, boosted trees, and neural network. It is outside the scope of this article to review each model's theoretical foundation in detail, but we refer the reader to prior research and summarize our reason for including them.

As the workhorse of traditional quantitative finance, OLS is an obvious place to start. Its simplicity is both appealing and limiting. LASSO (Tibshirani 1996) attempts to improve on OLS by penalizing the total magnitude of beta coefficients; ideally, it will identify a subset of the most reliable variables and neutralize the rest. Random forest (Hastie, Tibshirani, and Friedman 2008) consists of many individual tree models built from randomly selected subsets of the data. Each tree identifies thresholds that explain nonlinear patterns in the data, and their votes are aggregated. Boosted trees (Friedman 2001) works in a similar fashion, but trees are used iteratively to explain relationships that the previous iterations may have missed. This mechanism allows the model to find extreme or unusual effects. Neural networks (Goodfellow, Bengio, and Courville 2016) are the most complex of this set. They work by applying layers of transformation to the inputs, allowing them to capture more intricate dependencies among the variables.

The models require some design decisions, such as the number of predictors to be sampled by each tree in a random forest, the number of trees in a boosted trees model, and the number of neurons in a neural network. We make these choices based on judgment and applied knowledge from other fields. Appendix B describes the model specifications in detail. Other parameters, called *hyperparameters*, are calibrated as part of the training process, which we describe next.

Training and Testing

We split our data into a training period, from December 1992 to December 2014, and a testing period, from January 2015 to September 2020. The testing sample is not used until after the models have been calibrated on the training sample. This single train/test split is overly simplistic because it does not allow the models to learn and recalibrate with each passing month. However, it allows us to show the intuition of the models more clearly. As a result, our performance estimates may err on the conservative side.

It is helpful to think about the training of a model in two steps. First, we separate our panel of training data into 10 non-overlapping blocks of time. We choose a set of calibration parameters (hyperparameters) and evaluate the predictive efficacy (using mean squared error) of the model trained on every combination of nine time blocks on the corresponding remaining blocks. We specify the initial search ranges of calibration parameter values and repeat the calibration process in search of the combination of

parameter values with the most robust results, as measured by the average mean squared error of the validation blocks. This process, called *cross validation*, simulates how the model performs on unseen data that are, essentially, manufactured from within the training sample. Second, once we have identified the best calibration, we train the models using the best calibration and the entire set of training data.

The final rule gives a return prediction for any set of input values. Next, we describe how we interpret the internal logic of such a rule.

Interpretation with Model Fingerprints

An issue especially relevant to investment applications of machine learning is interpreting and understanding machine learning models. We apply the model fingerprint (fingerprint for short) proposed by Li, Turkington, and Yazdani (2020) as a framework for machine learning interpretation. The fingerprint method is a model-agnostic tool that provides insights into how predictors contribute to predictions at both global and local levels. Model fingerprint isolates the linear and nonlinear effects for each variable and the interaction effects for each pair of variables. These quantities are measured with partial predictions that vary one (or two) predictors at a time. The fingerprint then states the relative importance of each component in the same units as the predictions themselves. In other words, for each predictor, it shows the average extent to which changes in the predictor influence the prediction globally. In addition, when given a specific observation, the fingerprint can map predictor values to their partial predictions, thereby attributing a local prediction to each univariate variable, pairwise interaction effect, and the residual higher-order effects.

Similar to methods devised to estimate Shapley values for machine learning models, such as Shapley sampling (Štrumbelj and Kononenko 2013) and kernelSHAP (Lundberg and Lee 2017), the fingerprint calculates partial predictions by isolating a subset of predictors and sampling the complement set from the marginal distribution. This is the interventional conditional distribution presented by Janzing, Minorics, and Blöbaum (2020), which argues that such an approach is conceptually desirable for causal inference.

Unlike methods that estimate Shapley values, the fingerprint is based on the notion of partial dependence introduced by Friedman (2001). As a result, the fingerprint is more efficient computationally because it does not require computing the prediction outcome of all possible coalitions of variables. Although it is computationally efficient, the fingerprint's novel local prediction attribution method adheres to desirable properties of symmetry, dummy, additivity, and completeness, as defined by Shapley (1953).

- **Symmetry:** If two predictor variables are interchangeable in the model, the fingerprint values of the two predictors are the same.
- **Dummy:** If a variable is a dummy that does not change the model prediction in any way when added to the model, its fingerprint value should be 0.
- **Additivity:** If two models are combined such that the overall prediction is the sum of each subprediction, the fingerprint value of the overall model equals the sum of the fingerprint values corresponding to each submodel.
- **Completeness:** By construction, the fingerprint attribution of a local prediction sums up to the original prediction.

The fingerprint's local attribution, like Shapley estimation methods, converges to classical Shapley values when sampling from the marginal distributions and does not create unrealistic data points. It also augments the classical Shapley with interaction attributions.

To illustrate the procedure, imagine we are applying it to a linear model. We trace out the partial dependence function for a variable by setting it equal to many different values and, each time, computing the average prediction the model would make when combined with every available combination of the remaining variables. For a linear regression, this process will trace a straight line with slope equal to the regression beta. We then compute the mean absolute deviation of the linear predictions around their average value. Variables that cause large prediction changes are more influential than those that cause small prediction changes (note that variables are inherently standardized because we consider their full range of values, or at least a representative sample).

For most machine learning models, though, the procedure traces a nonlinear pattern. In this case, the fingerprint fits a straight line to the curve and computes the mean absolute deviation for that linear effect, as well as the mean absolute deviation of the full effect in excess of the linear component, which is the nonlinear component. Finally, the fingerprint captures the interaction effects for each pair of variables by computing the partial predictions for every combination of two variables in excess of the predictions those variables would give in isolation. The interaction effect equals the mean absolute deviation of these values. The linear, nonlinear, and interaction effects are all measured in comparable units that are the units of the model's predictions.

For categorical variables, such as economic sectors that are nominal, partial predictions can be calculated with the same procedure by iterating through all categories of the variable. The decomposition of linear and nonlinear effects by line fitting, however, is not applicable to such nominal variables that lack intrinsic ordering. We compute the mean absolute deviation of the sector variable's partial predictions around their average value, weighted by the frequency of each category in the training set. Like the fingerprint calculations of other variables, such measurements capture how influential the categorical sector variable is in the overall model predictions and are directly comparable because they remain in the same units as the model's predictions. We report the univariate effect of the sector variable in the linear effect category of the prediction fingerprints reported in this article.

For investment applications, Li, Turkington, and Yazdani (2020) also defined a performance fingerprint. The calculation takes the subcomponents of the prediction fingerprints, forms portfolios based on their various combinations, and distills cumulative returns for the linear, nonlinear, and interaction components.

BASE-CASE RESULTS

In this section, we present results for a base case in which models are trained to predict the total returns of each stock. The models are calibrated on panel data of monthly stock returns from December 1992 through December 2014, as described earlier. Once each model's predictive rule is set, we do not alter it during the testing process. We simply feed in the data for each stock in a given month (accounting for publication lags to represent data that would have been available at the time), obtain its return predictions, rank stocks accordingly, and form a portfolio that is long the 20% of stocks with the highest forecasted returns and short the 20% of stocks with the lowest forecasted returns. We assign equal weights to stocks within the long and short baskets.

Exhibit 2 shows the annualized return, volatility, return-to-risk ratio, and average turnover for each strategy. For reference, the act of liquidating all long and short

EXHIBIT 2

Performance of Base-Case Models in the Testing Sample (January 2015–September 2020)

Model	Return	Risk	Ratio	Turnover
Equal Factor Weights	1.7%	8.4%	0.20	3.1
OLS	0.0%	6.8%	–0.01	3.2
LASSO	0.5%	7.1%	0.06	3.6
Random Forest	2.5%	19.3%	0.13	1.9
Boosted Trees	3.8%	11.1%	0.34	4.9
Neural Network	4.0%	10.2%	0.39	4.8

positions in a portfolio and replacing them with new stocks would incur a turnover of 2× (we define the long and short baskets each as having weights that sum to 100%). Thus, if we assume, say, 20 bps of roundtrip transaction cost to substitute a position, the annual cost of the neural network strategy would be 4.8 times 0.20%, or 0.96%. As noted earlier, we only consider stocks in the S&P 500 universe, which have large capacity and are reasonably liquid. Although we report our results as long–short portfolios, they could be implemented as smaller overweight and underweight positions relative to a benchmark that does not require any actual shorting. In this case, the return, risk, and turnover would decrease in proportion to the smaller-sized tilts, and the return-to-risk ratio would remain constant.

We would like to highlight a few observations based on Exhibit 2. First, consider the linear strategies. "Equal factor weights" predicts returns as a simple average of the 12 security attributes (it does not include the sector dummy variables). It performs reasonably well in spite of its simplicity (although later, we show that it fails for other prediction objectives). In theory, OLS could have selected equal weights, yet it found a different solution that ended up performing worse out of sample. It may be overfitting to factors that performed well in the training data and underfitting the nonlinear nuance of the factors. LASSO sidesteps OLS's overfitting problem by relying on a more robust subset of predictors, but it still underperforms equal weighting.

Now, let's consider the more sophisticated machine learning models. Random forest stands out with less trading (turnover is 1.9×) and higher risk (volatility is 19.3%). It beats equal weighting in terms of raw returns, but its risk-adjusted returns are weak. Because its nature is to average across many fine-grained partitions of the data, random forest tends to rely on small nuances. By contrast, boosted trees and neural networks find more dramatic interaction effects. Both models trade more aggressively than random forest and generate higher returns. Neural network outperforms boosted trees by a small margin in every metric.

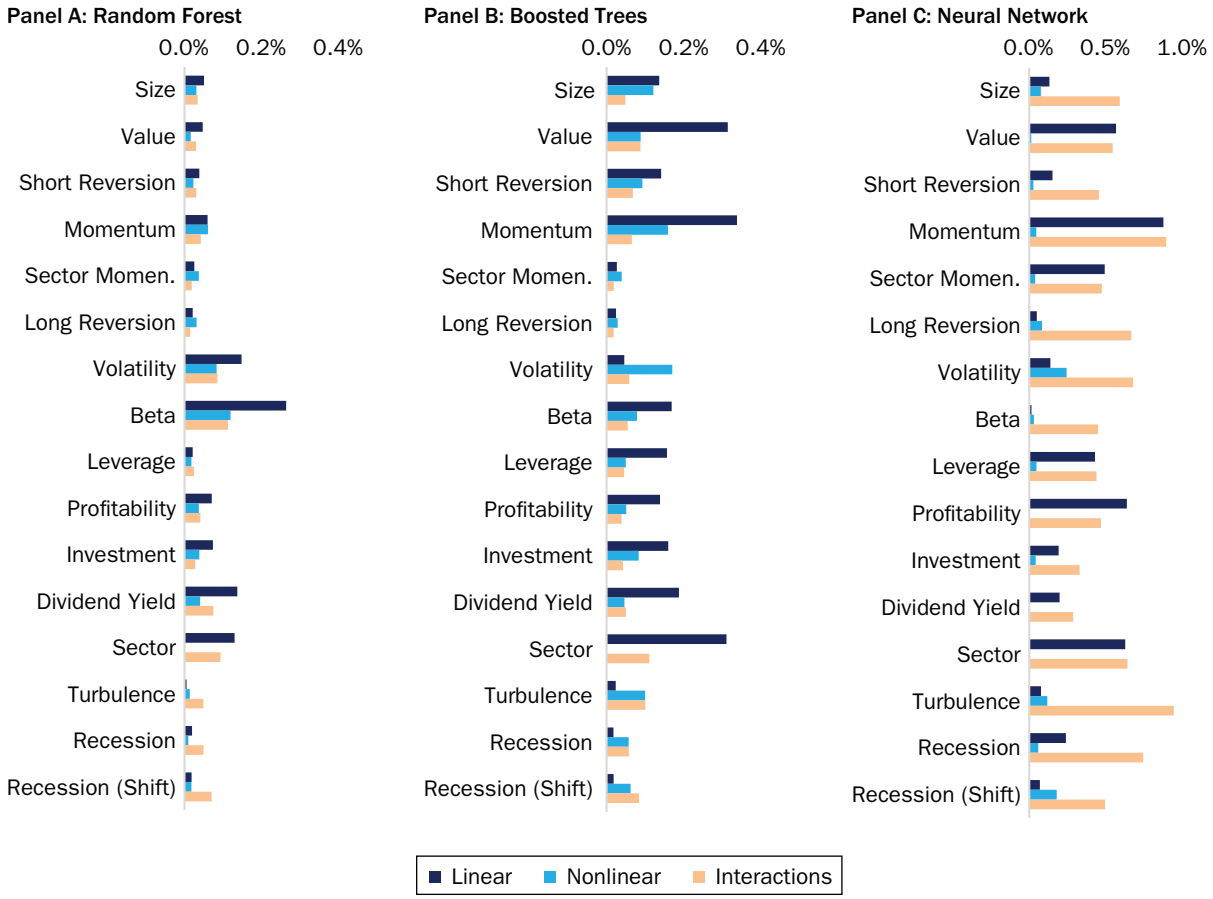
Interpretation

We now use the fingerprint methodology to explain the behavior of each machine learning model. Exhibit 3 shows the fingerprints for random forest, boosted trees, and neural network. We do not report the fingerprints for the linear models, but we note that they would contain linear effects only, with nonlinear and interaction terms equal to zero. It is important to remember that the prediction fingerprints derive from the training data only—the testing sample is not used in any way here.

It is clear from Exhibit 3 that the models learn different rules, even though they train on the same data. No model is perfect, so there is value in diversity. The linear effects are broadly similar for boosted trees and neural network, which offers a degree of comfort: Both load heavily on momentum and value, along with a collection

EXHIBIT 3

Prediction Fingerprints



of other factors.[2] Random forest is quite different, however, focusing on volatility and beta. Boosted trees has the most nonlinearity, and neural network has the most interactions. In Exhibit 3, we summarize interactions at the variable level, attributing half of each pairwise effect to its component parts.[3] As we might expect, the regime variables mainly condition the effects of the attributes, rather than suggesting strong directional effects of their own.

It is notable that the short reversion factor plays only a small role here. Prior machine learning studies, such as those by Gu, Kelly, and Xiu (2020) and Cong et al. (2020), ranked short-term reversal among the most important factors. We suspect the difference comes from our focus on investability. The reversal factor is consistent with frictions to trading: It is empirically robust for small stocks but mostly absent for large ones.[4] Exploiting this factor requires frequent trading in less-liquid stocks, which is costly to implement. Our models do not find it particularly useful.

We can dig deeper into the logic of the machine learning models by studying their interaction effects. Exhibit 4 lists the three most important interactions for each

[2] The effects occur on different scales across the three models. All of the effects are in comparable units because the models are tasked with predicting the same returns. The magnitudes are larger for neural network because they often cancel each other out owing to interactions.

[3] Higher-order effects also exist among collections of three or more variables, but they are too numerous to list. In general, we find that pairwise interactions account for a large share, if not most, of the total interaction effects for this application.

EXHIBIT 4

Most Important Interactions

	Random Forest	Boosted Trees	Neural Network
Top 3 Overall	Beta, Recession (shift)	Size, Turbulence	Volatility, Turbulence
	Volatility, Recession (shift)	Value, Turbulence	Size, Turbulence
	Beta, Turbulence	Sector, Turbulence	Momentum, Turbulence
Top 3 (without regime variables)	Beta, Volatility	Short Reversion, Sector	Momentum, Sector
	Beta, Yield	Momentum, Sector	Long Reversion, Momentum
	Beta, Sector	Value, Sector	Beta, Momentum

model, which in this case all happen to involve regime variables. We also list the three most important interactions that do not involve regime variables.

Random forest uses the recession indicator to condition beta and volatility, which are its dominant linear effects. Boosted trees conditions classic factors on market turbulence and discriminates across sectors in applying short reversion, momentum, and value. In addition to turbulence, neural network adjusts its preference for momentum stocks based on sector, long-term reversion, and beta.

In Exhibit 5, we map out two interaction effects for every possible combination of the variables. Panel A shows that boosted trees views low-volatility stocks more favorably when market turbulence is high, but when turbulence is low, its preference flips. This logic operates for the bottom quintile of low-volatility stocks, and it changes abruptly above that threshold. For the top two quintiles of high-volatility stocks, the relationship is more muted and in the opposite direction. Panel B shows that neural network has the most conviction in long-term reversion for stocks with low (negative) momentum. The model takes an asymmetric view: Reversion has a stronger pull when prices are depressed than when they are elevated.

Whether a model's predictions work well or not is a separate question. The performance fingerprint addresses this issue by attributing portfolio performance to each predictive component. Consider neural network as an example. Exhibit 6 shows the cumulative returns (in log scale) for the linear, nonlinear, and interaction effects, which add up to the model's total return.

In the training sample, the linear effect provides buy-and-hold exposure to a collection of factors, while interactions add substantial value. The interactions appear to act as a hedge, rising during risk events such as the dotcom crash in 2002–2003 and the global financial crisis in 2008. In the testing sample, the linear component performed poorly but was more than offset by the interactions, leading to a net gain for the strategy. Nonlinear effects contributed positive returns.

In Exhibit 7, we report the correlations between each model's subcomponent returns. The numbers in bold compare components within models, and the shaded boxes compare models within components. In the training sample, all three linear components are highly correlated. Boosted trees and neural networks also perform similarly in their nonlinear and interaction effects. These models must be picking many of the same stocks, or at least stocks with similar characteristics. The bottom left portion of the exhibit shows that each model's interactions act as a hedge to its linear component. Most of these relationships persist in the testing sample, with the notable exception that random forest seems to stray from the other models.

EXHIBIT 5

Sample Pairwise Effects (in basis points)

Panel A: Boosted Trees

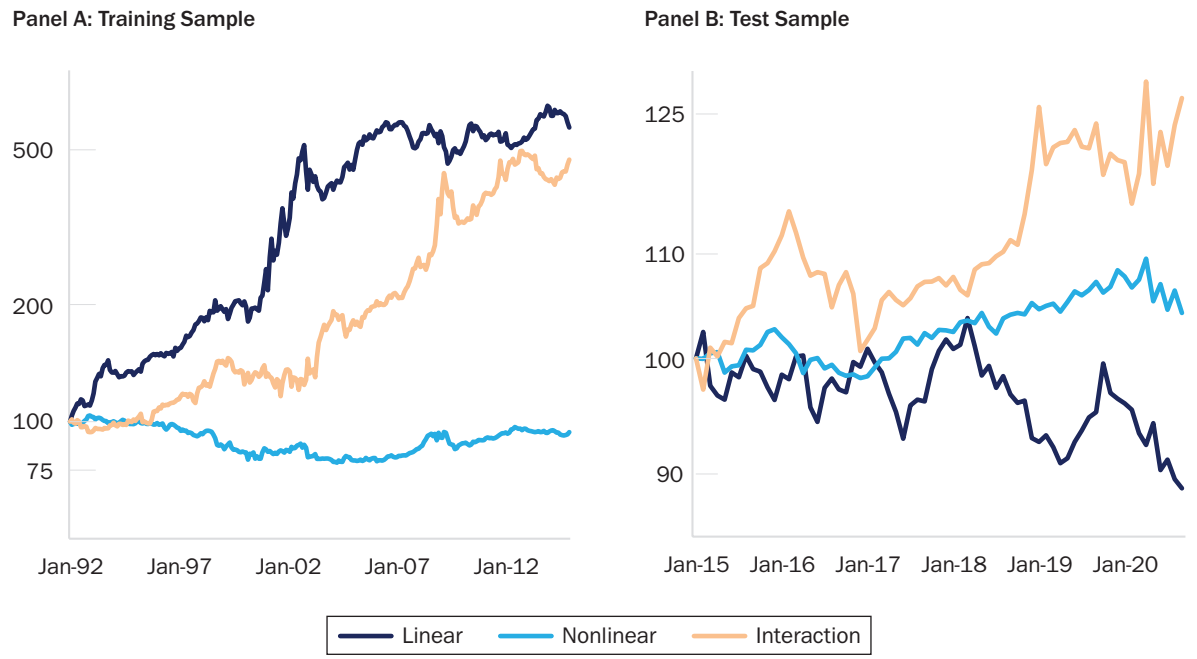
Market Turbulence \ Stock Volatility										
	6	6	2	1	0	–1	–5	–5	–6	–6
	6	6	2	1	0	–1	–5	–5	–6	–4
	4	4	0	–1	–2	–1	–2	–2	–2	0
	6	6	–1	–2	–3	–2	–3	–3	–3	0
	5	5	–2	–1	–2	–2	–3	–3	–2	0
	3	3	–1	0	–1	–1	–2	–2	–1	1
	4	4	–1	0	–1	–1	–2	–2	–1	–1
	–10	–10	–1	0	1	1	6	6	7	4
	–11	–11	–1	–1	2	3	8	8	8	5
	–13	–13	–1	0	3	3	8	8	9	6

Panel B: Neural Network

Momentum \ Long-Term Reversion										
	–19	–18	–16	–14	–11	–6	0	7	16	27
	–17	–16	–15	–12	–10	–6	–1	6	13	23
	–14	–13	–12	–10	–8	–5	–1	4	10	17
	–9	–9	–8	–7	–5	–4	–1	2	5	9
	–3	–3	–2	–2	–2	–2	–2	–1	–1	–1
	6	6	5	4	2	0	–2	–6	–9	–14
	19	17	15	12	8	3	–4	–11	–20	–30
	35	32	27	21	14	5	–6	–19	–33	–50
	55	49	42	33	21	7	–9	–28	–50	–74
	79	71	60	46	29	10	–13	–40	–69	–102

EXHIBIT 6

Performance Fingerprint for Neural Network



GOAL SETTING

Changing the prediction goal is a simple, direct, and effective way to adjust a model's behavior toward a set of prior beliefs and preferences. For example, if we do not believe that market timing is a good idea, we can remove it from consideration by asking the model to predict returns in excess of the market. Or, instead of imposing constraints to control trading costs, we can ask the model to predict longer-horizon returns that are inherently more stable. We can learn even more about the models by watching how they adapt to these changing objectives.

We vary the objective along two dimensions. First, we redefine the goal as returns in excess of a one-factor (CAPM) model or a six-factor model composed of the Fama and French (2015) five factors plus momentum (Jegadeesh 1990). Second, we rede-

EXHIBIT 7

Monthly Return Correlations for Performance Fingerprint Components

	Linear			Nonlinear			Interaction		
	RF	BT	NN	RF	BT	NN	RF	BT	NN
Panel A: Training Sample									
Linear									
Random Forest	1.00								
Boosted Trees	0.55	1.00							
Neural Network	0.60	0.83	1.00						
Nonlinear									
Random Forest	**–0.05**	–0.04	–0.05	1.00					
Boosted Trees	0.56	**0.03**	0.11	0.34	1.00				
Neural Network	0.43	0.08	**0.11**	0.28	0.67	1.00			
Interaction									
Random Forest	**–0.38**	–0.18	–0.15	**–0.17**	–0.26	–0.20	1.00		
Boosted Trees	–0.09	**–0.38**	–0.23	0.00	**–0.02**	–0.06	0.38	1.00	
Neural Network	–0.02	–0.35	**–0.44**	0.05	0.15	**–0.08**	–0.01	0.57	1.00
Panel B: Testing Sample									
Linear									
Random Forest	1.00								
Boosted Trees	–0.06	1.00							
Neural Network	0.10	0.63	1.00						
Nonlinear									
Random Forest	**0.10**	–0.32	–0.14	1.00					
Boosted Trees	0.50	**–0.49**	–0.18	0.68	1.00				
Neural Network	0.26	–0.30	**–0.32**	0.64	0.72	1.00			
Interaction									
Random Forest	**–0.14**	0.12	0.03	**–0.24**	–0.24	–0.10	1.00		
Boosted Trees	0.40	**–0.31**	–0.08	0.29	**0.36**	0.33	–0.04	1.00	
Neural Network	0.45	–0.50	**–0.42**	0.41	0.57	**0.45**	–0.07	0.60	1.00

fine the goal as 12-month returns instead of 1-month returns. Exhibit 8 shows the return-to-risk ratios in the testing sample for every combination of these goals.

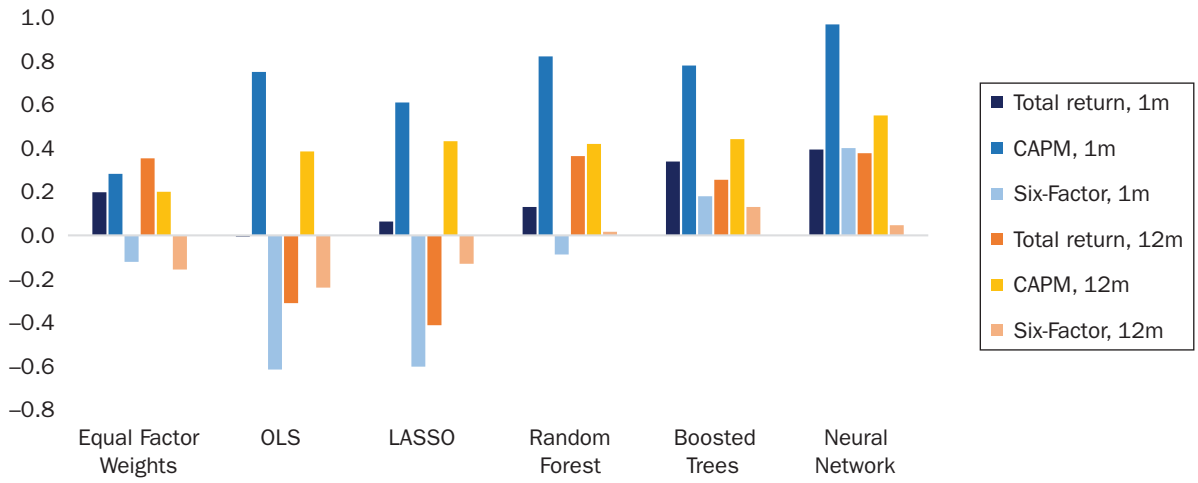
We evaluate each model relative to its stated benchmark. For the total return objective, we compute annualized return divided by annualized volatility. For the CAPM and six-factor objectives, we compute return and volatility in excess of an ex post regression-fitted benchmark. Full results are presented in Appendix A.

We introduce a slightly different rebalancing rule for the 12-month case. Instead of completely revising the weights each month, we revise 1/12th of the portfolio on a rolling basis. This approach aligns the holding period to the prediction horizon by ensuring that each set of chosen stocks stays in the portfolio for a full year, while also allowing the composition to gradually evolve from month to month.

Equal weighting all factors generates modest outperformance against a total return or CAPM target, but it fails to outperform the six-factor benchmark. OLS and LASSO have erratic performance, and their attempts to outperform the six-factor benchmark are especially misguided. The machine learning models added value consistently in this out-of-sample test. However, we stress that the magnitude of their performance advantage is smaller than what is reported in many other articles. In our view, the magnitude of the outperformance we show is realistic for an implementable and interpretable strategy.

EXHIBIT 8

Return-to-Risk Ratios in the Testing Sample (January 2015–September 2020)



The 12-month models have substantially lower turnover, with a maximum of 0.9× per year, compared to 4.9× per year for the 1-month equivalent (see Appendix A for the full set of results). Exhibit 9 shows the prediction fingerprints for the 12-month CAPM models. Comparing these results to the one-month total return model in Exhibit 3, we see some intuitive shifts. Although sector and profitability remain important for neural network, the model has shifted emphasis from momentum at both stock and sector levels to attributes such as beta and value among all linear effects. Neural network prefers the slower-moving economic conditions (recession likelihood) to the faster-moving financial turbulence indicator. Interestingly, random forest picks up many more interactions at the 12-month horizon. Boosted trees looked more like neural network for one-month total returns, but now its linear effects tend to align with random forest.

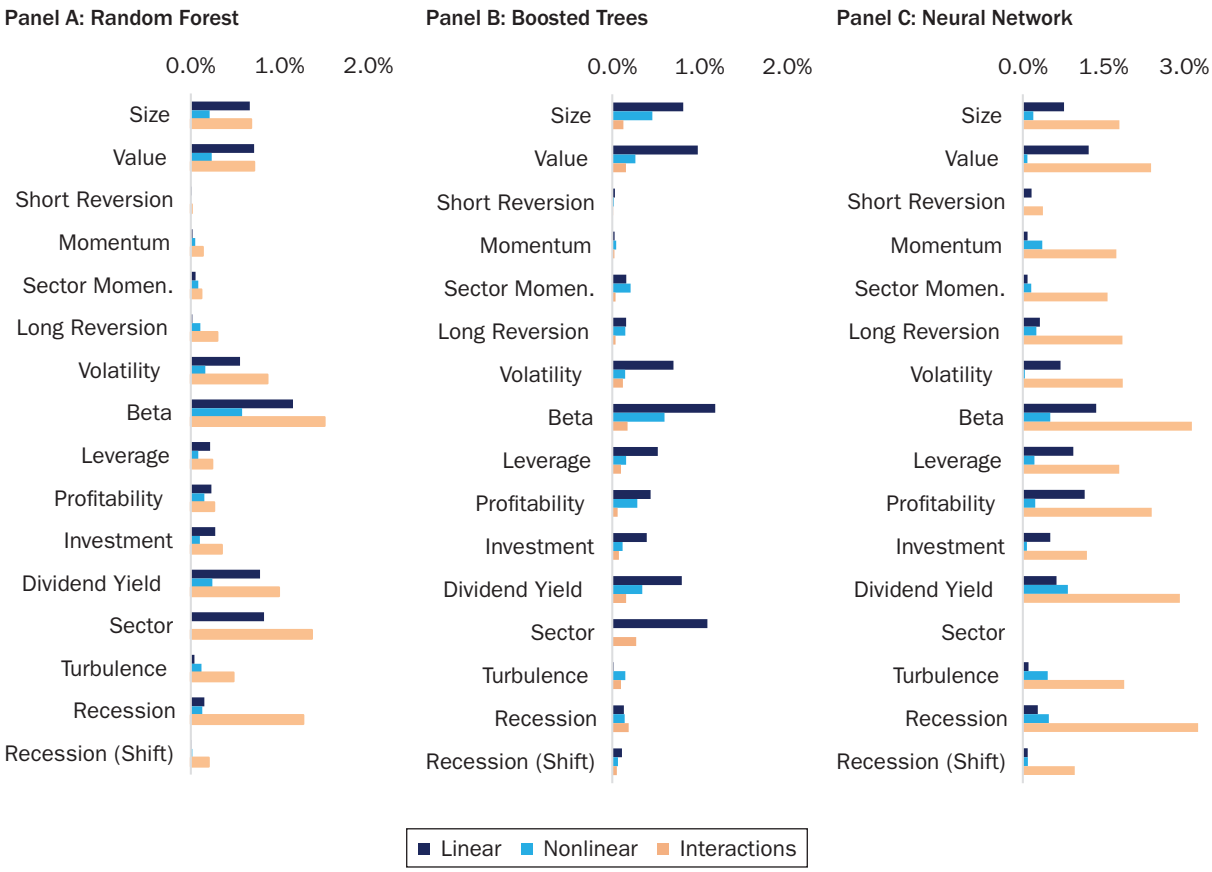
CONCLUSION

We propose a practical implementation of machine learning for stock selection, in which machine learning serves as a complement to good judgment, rather than a substitute for it. We posit that, from a practitioner's standpoint, complex models must be investable, interpretable, and interesting. As a result, our analysis differs from prior research in a few key ways. We focus on a narrow universe of stocks for investability. We focus on a narrow universe of predictors for interpretability. And we apply a method called the model fingerprint to reveal the logic behind predictions and the dynamics that drive performance.

There are many ways to extend and apply this framework. For example, it can be used for other asset classes or market segments. The models can be calibrated more often, configured with different goals, and fed different inputs. The fingerprints for interpretation can be used to study other machine learning models, and they can be adapted to explain—in real time—the rationale for any individual model prediction. The predictions can feed more-sophisticated portfolio construction models, or

EXHIBIT 9

Prediction Fingerprints (targeting 12-month returns above CAPM)



they can be directly combined with a qualitative forecasting process. Alternatively, the models might be used as a robustness check to intuition or as a way to foster an open dialog about investment ideas.

Machine learning lies somewhere between linear regression, which is understood through traditional statistics, and human judgment, which is understood through shared experience and discussion. By keeping models (relatively) simple and striving to understand them in new and better ways, we believe machine learning can become a practical tool for stock selection as opposed to just a curiosity.

APPENDIX A

ADDITIONAL RESULTS

Exhibit A1 shows the annualized return, risk, return-to-risk ratio, and turnover for strategies with various performance objectives. Return and risk are measured in excess

of the fitted linear regression exposure of each time series to the market factor (for CAPM) or the Fama and French (2015) five-factor model plus momentum (for six factor).

Exhibit A2 shows the prediction fingerprints for random forests, boosted trees, and neural networks with different prediction objectives. All prediction fingerprints are derived from the training data only, thereby reflecting how the models consider predictors when the predictive rules are formed. The linear, nonlinear, and interaction effects are measured in comparable units, the units of the model's predictions, so they reflect the average extent to which a predictor influences model predictions.

Exhibit A3 lists the three most important interactions for random forests, boosted trees, and neural networks with various prediction objectives. To account for the fact that regime variables participate in the majority of pairwise interaction terms, we also list the three most important interactions that do not involve regime variables.

APPENDIX B

MODEL SPECIFICATIONS

EXHIBIT A1

Model Performance in the Testing Sample (January 2015–September 2020)

Model	Return	Risk	Ratio	Turnover
Panel A: Total Return (1-month)				
Equal Factor Weights	1.7%	8.4%	0.20	3.1
OLS	0.0%	6.8%	–0.01	3.2
LASSO	0.5%	7.1%	0.06	3.6
Random Forest	2.5%	19.3%	0.13	1.9
Boosted Trees	3.8%	11.1%	0.34	4.9
Neural Network	4.0%	10.2%	0.39	4.8
Panel B: Total Return (12-month)				
Equal Factor Weights	1.9%	5.3%	0.35	0.7
OLS	–2.1%	6.8%	–0.31	0.6
LASSO	–2.9%	7.0%	–0.41	0.6
Random Forest	2.7%	7.5%	0.36	0.7
Boosted Trees	1.4%	5.6%	0.26	0.7
Neural Network	1.8%	4.7%	0.38	0.8
Panel C: CAPM (1-month)				
Equal Factor Weights	2.4%	8.4%	0.28	3.1
OLS	9.1%	12.1%	0.75	2.7
LASSO	8.7%	14.3%	0.61	2.3
Random Forest	12.3%	14.9%	0.82	1.9
Boosted Trees	9.7%	12.4%	0.78	4.6
Neural Network	12.4%	12.8%	0.97	4.0
Panel D: CAPM (12-month)				
Equal Factor Weights	1.0%	5.2%	0.20	0.7
OLS	2.9%	7.6%	0.39	0.6
LASSO	4.5%	10.4%	0.43	0.6
Random Forest	4.0%	9.4%	0.42	0.7
Boosted Trees	4.1%	9.3%	0.44	0.7
Neural Network	4.2%	7.6%	0.55	0.8
Panel E: Six-Factor (1-month)				
Equal Factor Weights	–0.7%	6.1%	–0.12	3.1
OLS	–1.9%	3.1%	–0.62	3.2
LASSO	–1.8%	3.0%	–0.60	3.1
Random Forest	–0.3%	3.2%	–0.09	2.8
Boosted Trees	0.6%	3.4%	0.18	3.2
Neural Network	2.2%	5.6%	0.40	4.6
Panel F: Six-Factor (12-month)				
Equal Factor Weights	–0.6%	3.8%	–0.16	0.7
OLS	–0.8%	3.3%	–0.24	0.8
LASSO	–0.4%	3.1%	–0.13	0.8
Random Forest	0.0%	2.9%	0.02	0.8
Boosted Trees	0.5%	3.5%	0.13	0.8
Neural Network	0.2%	3.8%	0.05	0.8

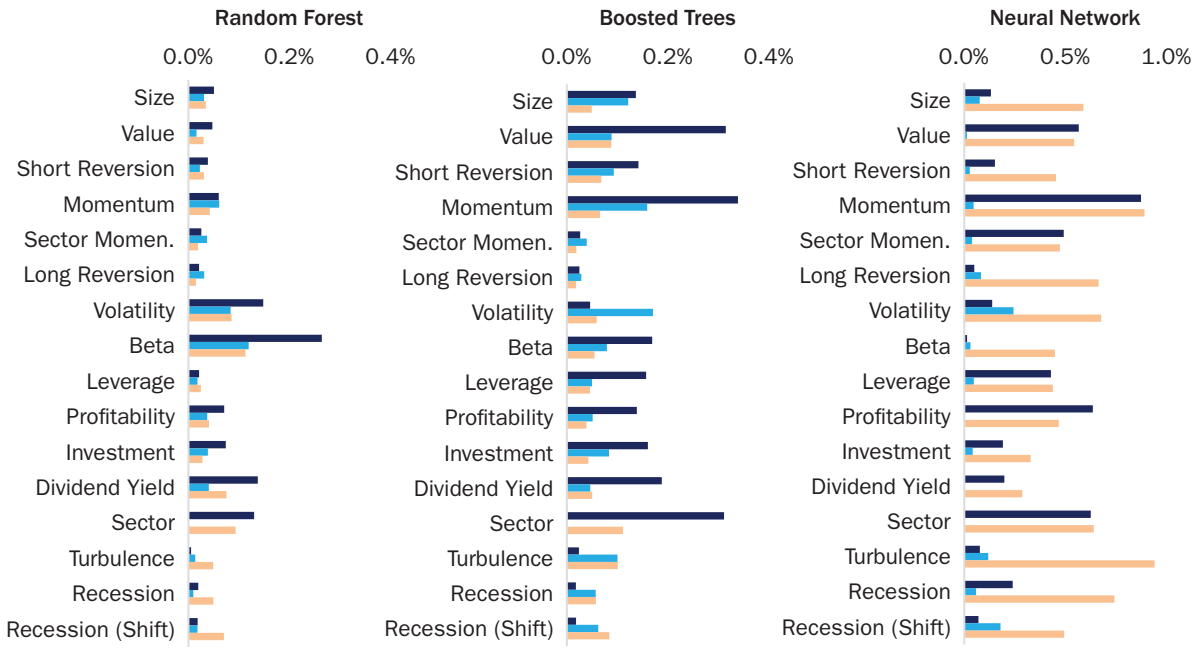
REFERENCES

Aldridge, I., and M. Avellaneda. 2019. "Neural Networks in Finance: Design and Performance." *The Journal of Financial Data Science* 1 (4): 39–62.

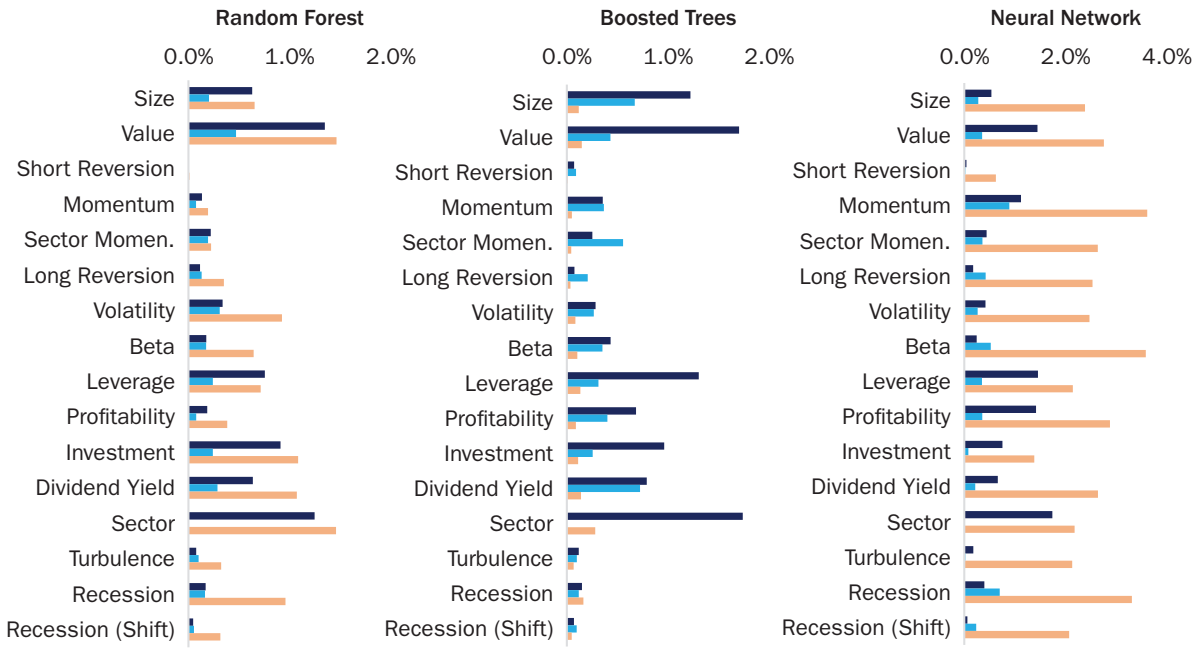
EXHIBIT A2

Model Prediction Fingerprints

Panel A: Total Return (1-Month)



Panel B: Total Return (12-Month)

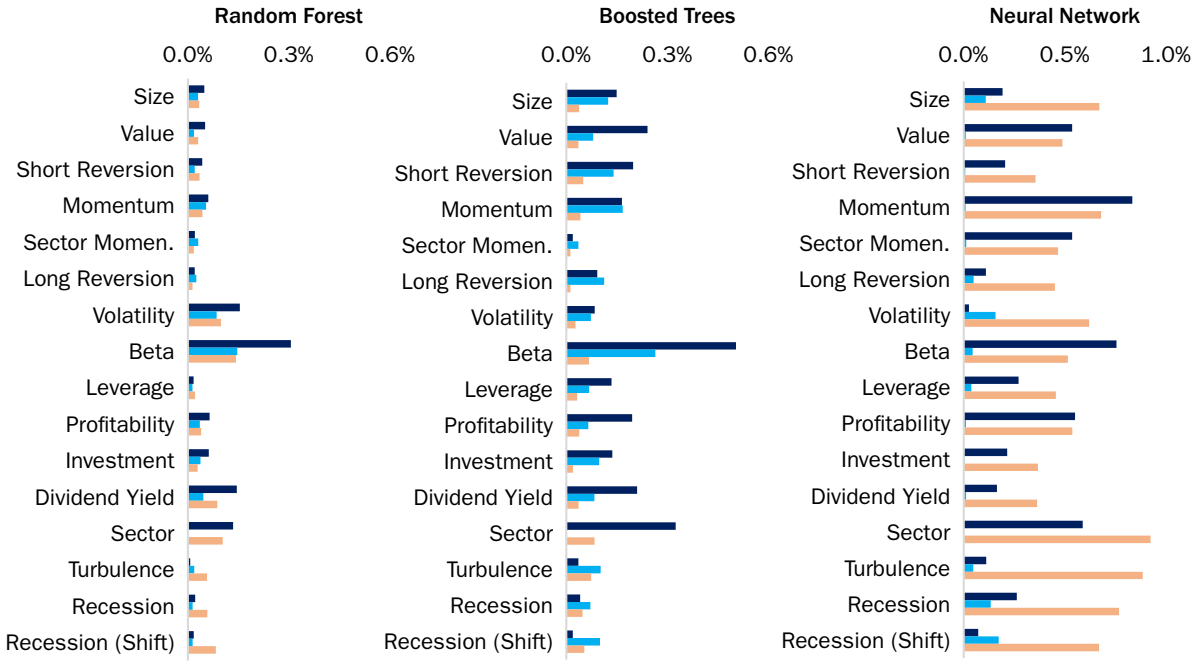


(continued)

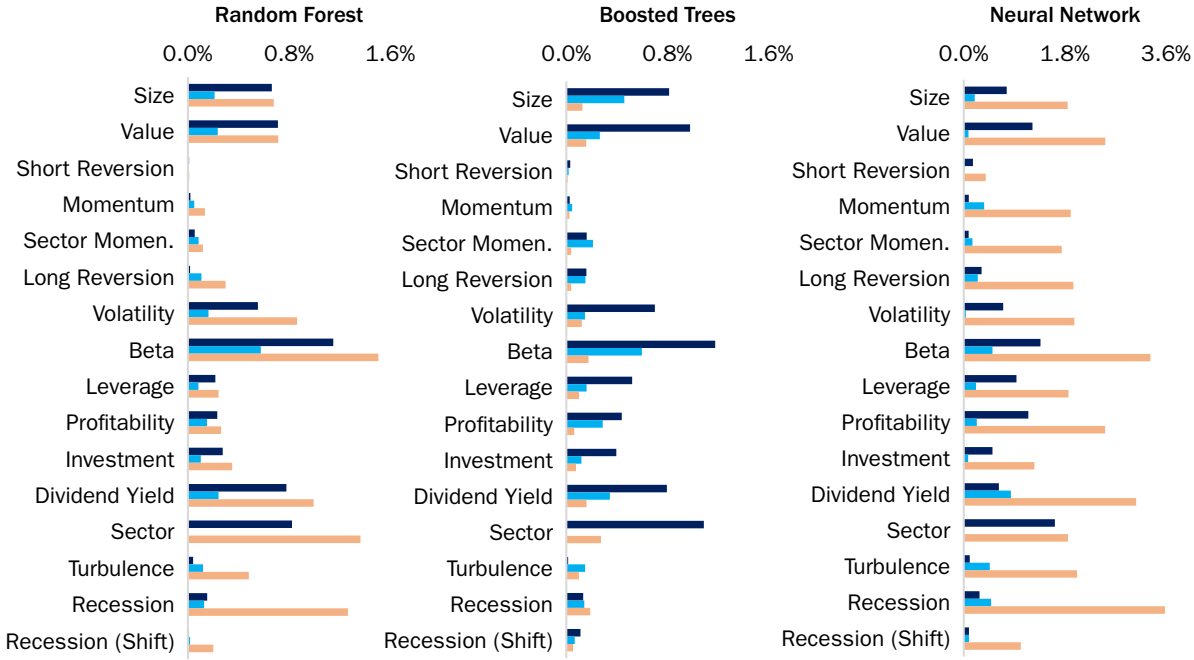
EXHIBIT A2 *(continued)*

Model Prediction Fingerprints

Panel C: CAPM (1-Month)



Panel D: CAPM (12-Month)

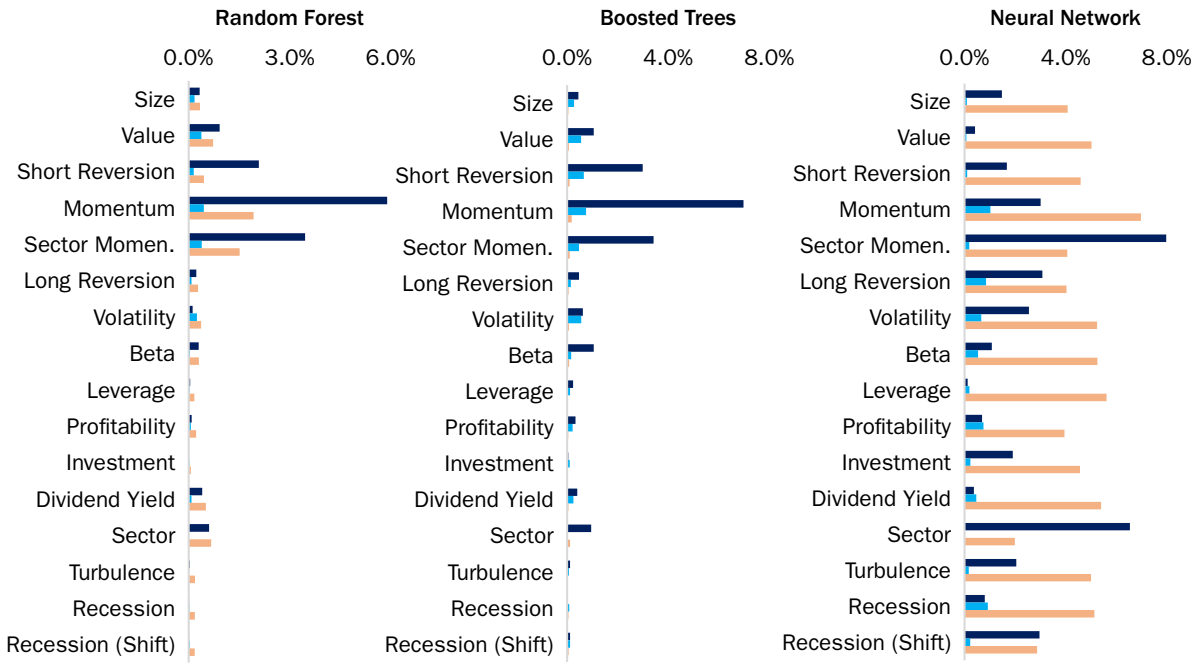


(continued)

EXHIBIT A2 *(continued)*

Model Prediction Fingerprints

Panel E: Six-Factor (1-Month)



Panel F: Six-Factor (12-Month)

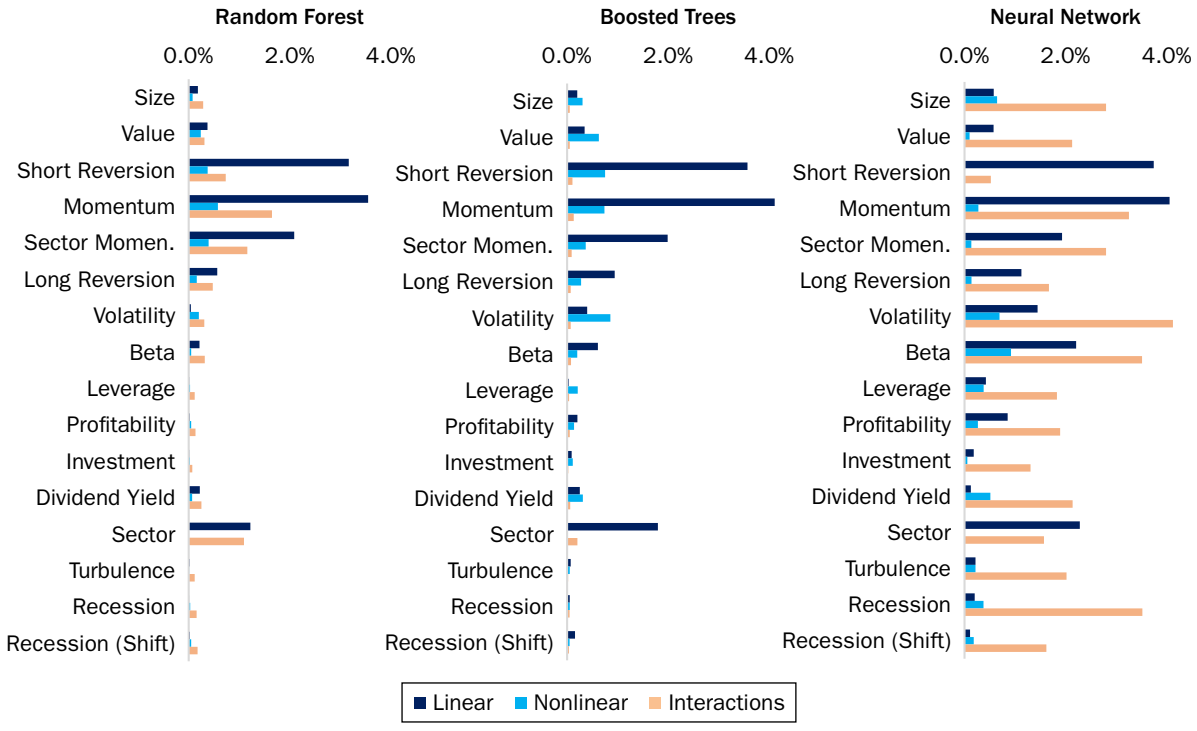


EXHIBIT A3

Most Important Interactions

	Random Forest	Boosted Trees	Neural Network
Panel A: Total Return (1-month)			
Top 3 Overall	Beta, Recession (shift) Volatility, Recession (shift) Beta, Turbulence	Size, Turbulence Value, Turbulence Sector, Turbulence	Volatility, Turbulence Size, Turbulence Momentum, Turbulence
Top 3 (without regime variables)	Beta, Volatility Beta, Yield Beta, Sector	Short Reversion, Sector Momentum, Sector Value, Sector	Momentum, Sector Long Reversion, Momentum Beta, Momentum
Panel B: Total Return (12-month)			
Top 3 Overall	Value, Sector Investment, Value Value, Volatility	Leverage, Sector Sector, Recession Value, Sector	Beta, Momentum Beta, Yield Size, Recession
Top 3 (without regime variables)	Value, Sector Investment, Value Value, Volatility	Leverage, Sector Value, Sector Yield, Sector	Beta, Momentum Beta, Yield Momentum, Sector Momentum
Panel C: CAPM (1-month)			
Top 3 Overall	Beta, Recession (shift) Beta, Recession Volatility, Recession (shift)	Beta, Recession (shift) Short Reversion, Sector Beta, Turbulence	Size, Turbulence Volatility, Recession Momentum, Sector
Top 3 (without regime variables)	Beta, Volatility Beta, Yield Beta, Sector	Short Reversion, Sector Value, Sector Beta, Sector	Momentum, Sector Short Reversion, Sector Momentum, Profitability
Panel D: CAPM (12-month)			
Top 3 Overall	Beta, Recession Sector, Recession Beta, Yield	Sector, Recession Value, Sector Yield, Sector	Beta, Recession Beta, Yield Beta, Value
Top 3 (without regime variables)	Beta, Yield Beta, Sector Value, Sector	Value, Sector Yield, Sector Leverage, Sector	Beta, Yield Beta, Value Volatility, Yield
Panel E: Six-Factor (1-month)			
Top 3 Overall	Momentum, Sector momentum Momentum, Value Momentum, Sector	Momentum, Sector momentum Momentum, Short Reversion Momentum, Sector	Leverage, Yield Short Reversion, Turbulence Leverage, Value
Top 3 (without regime variables)	Momentum, Sector momentum Momentum, Value Momentum, Sector	Momentum, Sector momentum Momentum, Short Reversion Momentum, Sector	Leverage, Yield Leverage, Value Momentum, Yield
Panel F: Six-Factor (12-month)			
Top 3 Overall	Momentum, Sector momentum Momentum, Sector Sector Momentum, Sector	Momentum, Sector Momentum, Short Reversion Beta, Sector	Beta, Volatility Beta, Momentum Size, Volatility
Top 3 (without regime variables)	Momentum, Sector momentum Momentum, Sector Sector Momentum, Sector	Momentum, Sector Momentum, Short Reversion Beta, Sector	Beta, Volatility Beta, Momentum Size, Volatility

Ang, A., R. J. Hodrick, Y. Xing, and X. Zhang. 2006. "The Cross-Section of Volatility and Expected Returns." *The Journal of Finance* 61 (1): 259–299.

EXHIBIT B1

Random Forest

Target	Number of Variables to Sample	Minimum Number of Observations in a Leaf	Maximum Number of Splits
Total return, 1 month	3	1,000	11
Total return, 12 months	8	1,100	60
CAPM, 1 month	4	800	11
CAPM, 12 months	7	100	110
Six-factor, 1 month	4	60	300
Six-factor, 12 months	7	50	170

NOTES: The number of trees is not tuned when training random forests because increasing this number in a random forest does not lead to overfitting. We fix the number of trees at 2,500 because we observe that increasing beyond this number merely leads to an increase in computational time with minimal changes in the predictions.

EXHIBIT B2

Boosted Trees

Target	Number of Variables to Sample	Minimum Number of Observations in a Leaf	Number of Trees
Total return, 1 month	5	650	40
Total return, 12 months	6	300	85
CAPM, 1 month	5	615	56
CAPM, 12 months	5	515	51
Six-factor, 1 month	5	400	100
Six-factor, 12 months	6	500	100

NOTES: Unlike the random forest setting, increasing the number of trees in a boosted trees model will rapidly lead to overfitting. We keep the maximum number of splits in our boosted trees fixed at 10 to avoid overfitting too early in the training process.

EXHIBIT B3

Neural Network

Target	Number of Nodes in a Layer	Weight/Bias Optimization Method	Epoch
Total return, 1 month	7	Resilient back-propagation	82
Total return, 12 months	11	Bayesian regularization	253
CAPM, 1 month	4	Resilient back-propagation	154
CAPM, 12 months	10	Bayesian regularization	211
Six-factor, 1 month	14	Gradient descent with momentum	1,001
Six-factor, 12 months	12	Bayesian regularization	232

NOTES: The total number of layers is fixed at three for our neural network models. The amount of data in this research is not suitable for training a deep network. We observe that changing the number of layers in this research will quickly lead to the model being stuck at local minima and thus yielding trivial solutions.

Avramov, D., T. Chordia, and A. Goyal. 2006. "Liquidity and Autocorrelations in Individual Stock Returns." *The Journal of Finance* 61 (5): 2365–2394.

Balakrishnan, K., E. Bartov, and L. Faurel. 2010. "Post Loss: Profit Announcement Drift." *Journal of Accounting and Economics* 50 (1): 20–41.

Banz, R. 1981. "The Relationship between Return and Market Value of Common Stocks." *Journal of Financial Economics* 9 (1): 3–18.

Bhandari, L. C. 1988. "Debt/Equity Ratio and Expected Common Stock Returns: Empirical Evidence." *The Journal of Finance* 43 (2): 507–528.

Bryzgalova, S., M. Pelger, and J. Zhu. 2020. "Forest through the Trees: Building Cross-Sections of Stock Returns." SSRN, https://dx.doi.org/10.2139/ssrn.3493458.

Campbell, J. Y., S. J. Grossman, and J. Wang. 1993. "Trading Volume and Serial Correlation in Stock Returns." *The Quarterly Journal of Economics* 108 (4): 905–939.

Cong, L., K. Tang, J. Wang, and Y. Zhang. 2020. "Alpha Portfolio for Investment and Economically Interpretable AI." SSRN, https://dx.doi.org/10.2139/ssrn.3554486.

Cooper, M. J., H. Gulen, and M. J. Schill. 2008. "Asset Growth and the Cross-Section of Stock Returns." *The Journal of Finance* 63 (4): 1609–1651.

Fama, E., and K. French. 2015. "A Five-Factor Asset Pricing Model." *Journal of Financial Econometrics* 116 (1): 1–22.

Fama, E., and J. D. Macbeth. 1973. "Risk, Return, and Equilibrium: Empirical Tests." *Journal of Political Economy* 81 (3): 607–636.

Fischer, T., and C. Krauss. 2018. "Deep Learning with Long Short-Term Memory Networks for Financial Market Predictions." *European Journal of Operational Research* 270 (2): 654–669.

Friedman, J. H. 2001. "Greedy Function Approximation: A Gradient Boosting Machine." *Annals of Statistics* 29 (5): 1189–1232.

Goodfellow, I., Y. Bengio, and A. Courville. 2016. *Deep Learning.* Cambridge, MA: MIT Press.

Gu, S., B. Kelly, and D. Xiu. 2020. "Empirical Asset Pricing via Machine Learning." SSRN, https://dx.doi.org/10.2139/ssrn.3159577.

Hastie, T., R. Tibshirani, and J. Friedman. 2008. *The Elements of Statistical Learning, Data Mining, Inference, and Prediction*. New York: Springer.

Janzing, D., L. Minorics, and P. Blöbaum. 2020. "Feature Relevance Quantification in Explainable AI: A Causal Problem." *International Conference on Artificial Intelligence and Statistics, PLMR* 108: 2907–2916.

Jegadeesh, N. 1990. "Evidence of Predictable Behavior of Security Returns." *The Journal of Finance* 45 (3): 881–898.

Jegadeesh, N., and S. Titman. 1993. "Returns to Buying Winners and Selling Losers: Implications for Stock Market Inefficiency." *The Journal of Finance* 48 (1): 65–91.

Kinlaw, W., M. Kritzman, and D. Turkington. 2021. "A New Index of the Business Cycle." *Journal Of Investment Management* 19 (3): 4–19.

Kritzman, M., and Y. Li. 2010. "Skulls, Financial Turbulence, and Risk Management." *Financial Analysts Journal* 66 (5): 30–41.

Li, Y., D. Turkington, and A. Yazdani. 2020. "Beyond the Black Box: An Intuitive Approach to Prediction with Machine Learning." *The Journal of Financial Data Science* 2 (1): 61–75.

Litzenberger, R. H., and K. Ramaswamy. 1982. "The Effects of Dividends on Common Stock Prices Tax Effects or Information Effects?" *The Journal of Finance* 37 (2): 429–443.

Lundberg, S., and S-I., Lee. 2017. "A Unified Approach to Interpreting Model Predictions." *arXiv* 1705.07874.

Moritz, B., and T. Zimmermann. 2016. "Tree-Based Conditional Portfolio Sorts: The Relation between Past and Future Stock Returns." SSRN, http://dx.doi.org/10.2139/ssrn.2740751.

Moskowitz, T. J., and M. Grinblatt. 1999. "Do Industries Explain Momentum?" *The Journal of Finance* 54 (4): 1249–1290.

Rasekhschaffe, K. C., and R. C. Jones. 2019. "Machine Learning for Stock Selection." *Financial Analysts Journal* 75 (3).

Rosenberg, B., K. Reid, and R. Lanstein. 1985. "Persuasive Evidence of Market Inefficiency." *The Journal of Portfolio Management* 11 (3): 9–17.

Shapley, L. S. 1953. "A Value for n-Person Games." *Contributions to the Theory of Games* 2 (28): 307–317.

Sharpe, W. F. 1964. "Capital Asset Prices: A Theory of Market Equilibrium under Conditions of Risk." *The Journal of Finance* 19 (3): 425–442.

Štrumbelj, E., and I. Kononenko. 2013. "Explaining Prediction Models and Individual Predictions with Feature Contributions." *Knowledge and Information Systems* 41: 647–665.

Tibshirani, R. 1996. "Regression Shrinkage and Selection via the LASSO." *Journal of the Royal. Statistical Society. Series B* 58 (1): 267–288.