ENSEMBLE LEARNING IN INVESTMENT: AN OVERVIEW

Alireza Yazdani, PhD
Senior Vice President, Data Science and AI, Investor and Issuer Services, Citi

Introduction

The use of quantitative methods in finance and investment applications is not a new phenomenon, particularly among more sophisticated practitioners. This usage spans a wide range of methods, including but not limited to probabilistic, statistical, and stochastic models; univariate and multivariate time-series techniques; discrete and continuous simulations; linear and nonlinear programming; dynamic programming; and mathematical optimization. Notwithstanding the great success of many such quantitative methods, they have known limitations, including the parametric or prespecified functional forms that may lack flexibility, theoretical data distribution assumptions that are regularly violated, and computational challenges associated with the so-called curse of dimensionality. An ever-growing list of federated datasets, alongside the development of tools that transform such data into useful variables for explanatory and predictive tasks, calls for a paradigm shift in how financial data are used and processed.

In recent years, financial data science and machine learning (ML) methods have been widely adopted as prediction and decision-support means in various investment applications. The following are some of the contributing factors to such widespread adoption:

- Numerical power and flexibility of ML in dealing with unstructured, private, and microlevel data beyond the traditional econometric methods (López de Prado 2019)
- Scalability of ML in distilling large datasets consisting of many variables (factor zoo) into subsets that reliably predict cross-sectional variations in stock returns (Gu, Kelly, and Xiu 2020)
- The efficacy of the ML models in predicting cross-section of stock returns, compared with ordinary least-squares (OLS) regression, attributable to their ability to uncover nonlinear patterns and robustness to multicollinear predictors (Gu et al. 2020)
- The possibility of demystifying (i.e., breaking down and explaining) ML predictions into linear, nonlinear, and interactive components (Li, Turkington, and Yazdani 2020)

Simonian and Fabozzi (2019) argued that financial data science is to be regarded as a discipline in its own right and not a mere application of data science methods to finance or a branch of classical econometrics.

One major branch of machine learning is supervised learning, where available data are used to train a model that is to establish a mapping between input variables (also known as features or predictors) and output variables (also known as the labels, the response, or the dependent variable). If the response variable is continuous numeric, the model is called a regression, and if the response is categorical, the model is called a classification. This mapping is then used for inference and scoring, to explain the relationship between the input and response, or ideally to make predictions about the new values of the response variable given the new input.

Among the widely used supervised learning models are ensembles, which are based on the presumption of the wisdom of crowds. In this view, combining diverse opinions from a committee of several individuals can lead to greater wisdom. In other words, combining multiple predictions obtained from numerous somewhat independent methods often leads to better predictions than identifying a single best prediction. In a regression, this combining can be simple averaging or other sophisticated weighting mechanisms, whereas in a classification, majority voting or other weighting mechanisms may be used. Technically, the question is how effectively an ensemble model can decompose bias and variance terms, to be able to more generally work well on previously unseen test data not used for model training. Recall that the expected model prediction error for a regression model, as measured by the mean squared error (MSE), is formulated as

$$\text{MSE} = E[(\hat{y} - y)^2] = E[(\hat{y} - E[\hat{y}])^2] - (E[\hat{y}] - y)^2.$$

This relationship can be interpreted as:

$$\text{MSE} = \text{Variance} + \text{Bias}^2.$$

Here, bias refers to the error committed by the learning algorithm, where higher values may indicate underfitting as a result of the model failing to adequately establish the relationships between features and output. Variance refers to the error from the sensitivity of the model to another subset of training data, where higher values indicate overfitting. Generally, the more complex the model, the lower the bias and the higher the variance will be. Instances of model complexity include using many features, a greater number of parameters, and complex model design or architecture. The bias–variance trade-off concerns minimizing test error by finding the right model complexity.

Before delving into the various categories and the qualities of ensemble models, it is worth mentioning that the idea of combining predictions is not an entirely new one, and it has previously been studied by statisticians and econometricians, particularly in the context of time-series forecasting. In a seminal work, Bates and Granger (1969, pp. 451–68) observed that "combined forecasts yield lower mean-square error than either of the original forecasts." Over time, several combination methods have been developed, including simple averaging, time-weighted and performance-weighted estimations, nonlinear combination, and combining by learning (Wang, Hyndman, Li, and Kang 2023). Combining is directly related to a special case of ensemble learning known as stacking, which I will discuss later. By aggregating predictions, ensemble models reduce the impact of errors and noise that may exist in individual predictions, enabling them to achieve greater overall accuracy, reliability, and stability.

Ensemble Methods

In principle, ensemble methods share two main components: a diverse set of predictive "base learner" models trained on sampled subsets of training data and a combination mechanism, such as averaging or voting, used to aggregate the predictions from these base learners (Flach 2012). Broadly speaking, ensemble methods fall into one of three categories: bagging, boosting, and meta-learning. In this section, we review these categories, as well as some properties and possible drawbacks of ensemble learning methods.

Bagging

Short for "bootstrapped aggregating," bagging is an instance of ensemble averaging where a diverse group of base models is developed and combined with the end goal of improving

the stability and accuracy of classification and regression algorithms. These base models are built independently through repeatedly bootstrapping (i.e., random sampling with replacement) from the training data (see **Exhibit 1**). Bagging is particularly effective because of the possibility of the calculation of so-called out-of-bag error: the average of the errors from each independently developed model evaluated on the unselected subset of training data during the bootstrapping. Out-of-bag error estimation is an effective mechanism similar to cross-validation, leading to better model calibration and overall improvement of predictions.

Perhaps the best-known example of bagging in machine learning is the *random forests* algorithm (Breiman 2001). Random forests are generated by building a large collection of de-correlated and possibly deep decision trees and then combining them through averaging in regression or majority voting in classification (see **Exhibit 2**). Decision trees learn by recursive partitioning of the feature space and classifying members of a population by separating

Exhibit 1. A Bootstrapped Aggregation Ensemble

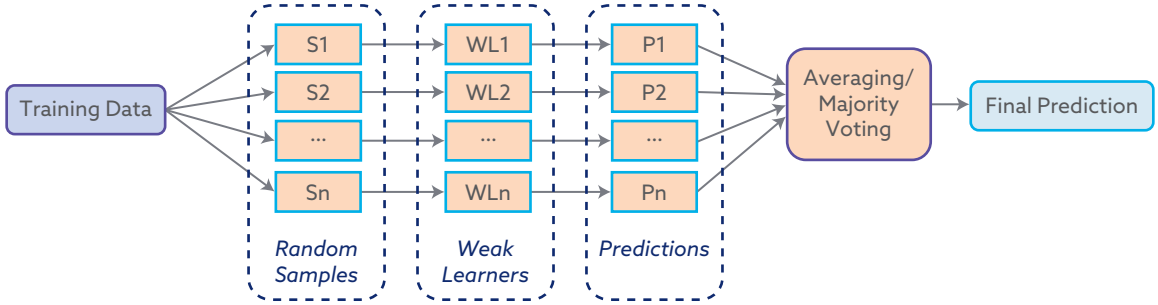
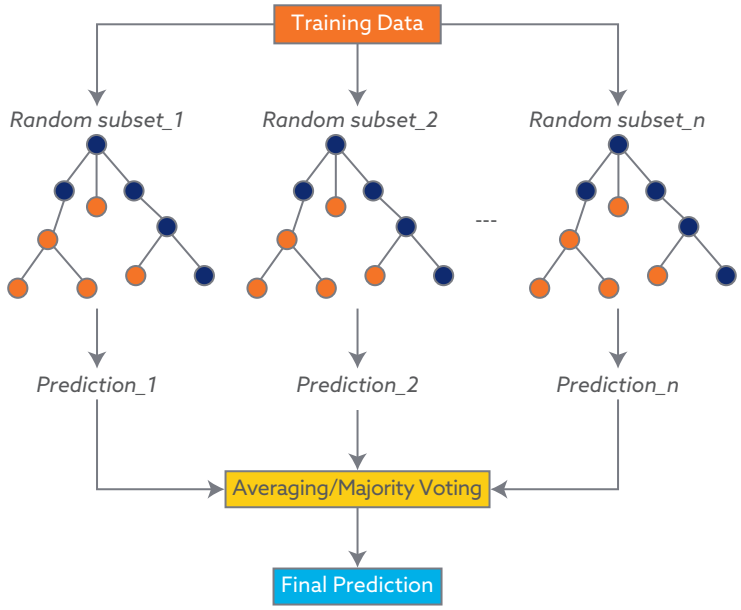


Exhibit 2. A Random Forest Model



it into subpopulations based on various dichotomous independent variables. Decision trees have desirable properties, including automatic variable selection, handling mixed features and missing values, and scalability. The depth and breadth of decision trees enables them to capture local patterns in training data and reduce bias. Because these trees are identically distributed, an ensemble consisting of the average of those trees has the same bias as the individual trees. If these trees are grown on the random subsets of data and input variables, their pairwise correlation is reduced to the extent that the bagged ensemble consisting of a sufficiently large number of trees can achieve an overall lower variance (Hastie, Tibshirani, and Friedman 2017).

Technically, for m independently and identically distributed decision tree predictions with variance σ^2, their average has a variance of $\frac{\sigma^2}{m}$. When predictions are not independent yet their maximum pairwise correlation is bounded by ρ, the variance of their average is bounded by $\rho\sigma^2 + \frac{(1-\rho)\sigma^2}{m}$. In other words, a random selection of input variables will reduce the maximum pairwise correlation, ρ (hence reducing the first term), and an increase in the number of decision trees, m, will reduce the second term, resulting in overall reduction of the variance. For an elegant exposition and technical discussion on several properties of random forests, see Breiman (2001). The implementation of ensemble-based methods for classification, regression, and anomaly detection is straightforward in practice. For example, in Python, many such models can be created using, among others, the scikit-learn library. The following is an example code for creating the random forest classification model in Python:

```
from sklearn.ensemble import RandomForestClassifier

from sklearn.model_selection import train_test_split

#create a synthetic dataset

from sklearn.datasets import make_classification

X, y = make_classification(n_samples=1000, n_features=10, n_informative=6,
random_state=123)

#Split the dataset into 80% training and 20% testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=123)

#train a Random Forest model and predict

model = RandomForestClassifier(random_state=123)

model.fit(X_train, y_train)

model.predict(X_test)

#mean accuracy

model.score(X_test,y_test)
```

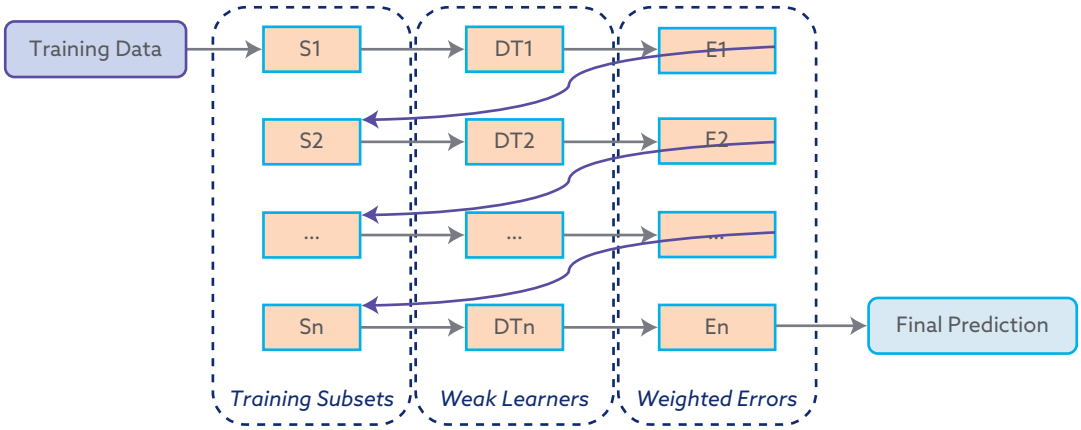
Boosting

Boosting is similar to bagging in that it benefits from data sampling and variable sampling, as well as in the property of combining several base learners. In boosting, however, unlike bagging, the basic mechanism is bias reduction, particularly using shallow trees. In addition, base learners are trained sequentially to correct the errors of the previous learners. The original boosting classifier, AdaBoost, proposed by Freund and Schapire (1996), works by sequentially training weak classifiers that focus on the mistakes of the previous ones by assigning higher weights to misclassified samples. The final prediction is a weighted sum of the individual learners' outputs (see **Exhibit 3**). Notwithstanding its superior predictive accuracy, the AdaBoost method is less readily interpretable than decision trees and requires a longer training time.

The *gradient boosted model* (GBM) proposed by Friedman (2001) has proved to be a popular alternative boosting method because it mitigates some of AdaBoost's problems. The GBM works by sequentially adding new tree predictors and trying to fit the new predictor to the residual errors by the previous tree, which gradually improves predictions as new trees are added (Géron 2019). Notably, recent advancements in boosting algorithms—for example, XGBoost (Chen and Guestrin 2016) and LightGBM (Ke, Meng, Finely, Wang, Chen, Ma, Ye, and Liu 2017)—focus on regularization, adaptability, efficiency, and scalability, while maintaining high accuracy. The following is an example code for creating the gradient boosted regression model in Python:

```
from sklearn.datasets import make_regression

from sklearn.ensemble import GradientBoostingRegressor

from sklearn.model_selection import train_test_split

#create a synthetic dataset

X, y = make_regression(n_samples=1000, n_features=10, n_informative=6,
random_state=123)

#Split the dataset into 80% training and 20% testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=123)

#train a model, predict and score

model = GradientBoostingRegressor(random_state=123)

model.fit(X_train, y_train)

model.predict(X_test)

#the coefficient of determination R-squared

model.score(X_test, y_test)
```

Exhibit 3. Example Boosting Mechanism Consisting of Decision Tree Base Learners



Meta-Learning

Meta-learning involves training a set of base models and also training a meta-model to combine them in order to achieve the greatest bias–variance trade-offs. For example, using as new input features the predictions from several diverse base learners (e.g., consisting of a linear model, multiple trees, and a neural network), a linear meta-model can be used to learn the weights to best combine these predictions (Flach 2012).

Alternatively, a logistic regression, a decision tree, or an even more complex model can be used as a meta-model by following the same idea that predictions from base learners are input features to the meta-model (as shown in Exhibit 3). This approach is also known as *stacked generalization* (Wolpert 1992) or simply *model stacking*, as shown in **Exhibit 4**. Stacking usually involves estimating weights for each base learner using the least squares method. These weights can be restricted to values greater than zero and with a total sum of one that solve the associated quadratic programming. This approach has the advantage that estimated weights can be interpreted as probability values and also avoids giving too much weight to base models with high complexity. One key consideration while training meta-models is to use a validation sample that is different from the data used in training base learners, so as to avoid propagating any biases in the base learners.

The following is an example Python code for creating a stacked classifier by using a random forest and a support vector classifier as the base models and logistic regression as the meta-learner:

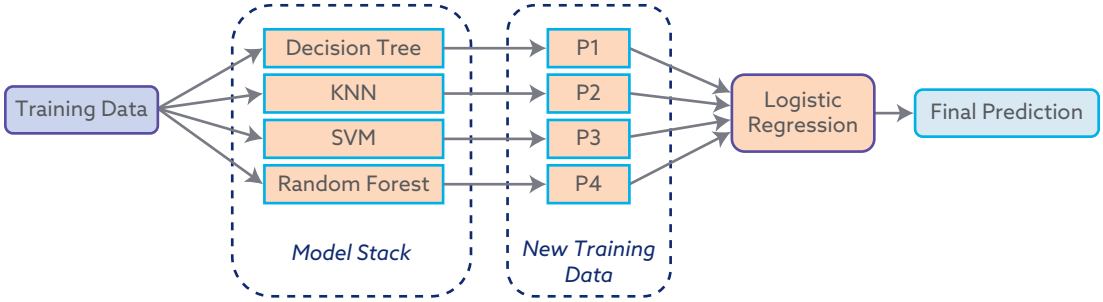
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import LinearSVC
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline
from sklearn.ensemble import StackingClassifier
#create a synthetic dataset
from sklearn.datasets import make_classification
X, y = make_classification(n_samples=1000, n_features=15, n_informative=6,
random_state=123)
#Split the dataset into 80% training and 20% testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=123)
#train a meta-model and predict
estimators = [('rf', RandomForestClassifier(n_estimators=100, random_state=123)),
('svr', make_pipeline(StandardScaler(),
LinearSVC(random_state=123)))]
clf = StackingClassifier(estimators=estimators, final_estimator=LogisticRegression())
clf.fit(X_train, y_train)
score(X_test, y_test)
```

Properties of Ensemble Learning Methods

Before listing major instances of the financial applications of ensemble models, I want to highlight important features and mechanisms associated with such models that are key to their widespread adoption:

- The bootstrapped aggregation or iterative correction mechanisms described previously imply that ensemble models may achieve optimal bias–variance trade-offs, reduced risk of overfitting, and improved robustness to noise and outliers. These features make ensembles

Exhibit 4. Example Model Stacking, Where Base Learners' Predictions Are Passed Through a Meta-Learner



a powerful choice for predictive tasks that require good out-of-sample generalizations on high-variance, low-signal-to-noise financial datasets.

- Ensemble models scale well with large datasets, making them suitable for big data in finance. In particular, they can handle various independent or interdependent variables, including firm fundamentals, macroeconomic factors, and market sentiments, and focus on the most relevant variables that generate the strongest signals.
- Financial systems are complex, nonlinear, and ruled by different market regimes and economic cycles. They consist of numerous variables that may behave and interact unexpectedly under volatile and abnormal circumstances. Heterogeneous ensemble methods, particularly regression trees, combine models of different types to capture multi-way predictor interactions, allowing flexibility in capturing various patterns in financial data and subsequently adapting to nonlinear and short-lived dynamics.
- Ensemble models can easily accommodate many data preprocessing and adjustment strategies (such as balanced bagging or weighted boosting) to handle rare and imbalanced data, often encountered in fraud detection, credit default prediction, economic recession, and extreme risk events.
- Explainability techniques for tree-based ensembles may be used to make these models more transparent by explaining their predictions. For example, feature importance scores identify influential predictors, and Shapley values (Shapley 1953) provide local attribution analysis of the predictions. These techniques greatly aid in understanding the decision-making processes.

Ensemble Learning Challenges

Despite strong predictive performance, using ensemble learning may come with certain challenges, although arguably not as much as with some other machine learning and deep learning techniques. First, ensemble models may be considered black boxes without using interpretability tools, making it a challenge to explain predictions to nontechnical stakeholders and reducing trust in critical applications. In addition, training and predicting with ensemble models often require significant computational resources, particularly for large datasets or complex models. As such, compared with simpler models, deploying ensemble models in production

environments can be more challenging because of their larger size, more complex pipelines, and extra computational demands.

In particular, gradient boosting algorithms can be slow to train because of their sequential tree-building process and the complexity of the hyperparameter tuning process. Without adequate calibration, either overfitting or underfitting is possible. Finally, following the "no free lunch" theorem (Wolpert 1996), which states that there is no best single machine learning algorithm for all possible prediction problems, one could argue that there are numerous real-world datasets and prediction problems for which ensemble models will not be the top performers. In such circumstances, other models and algorithms are required, depending on the "speed–accuracy–complexity trade-offs" (Wolpert 2002).

Applications of Ensemble Learning in Finance

A quick survey of the literature published in the last few years points to several financial applications of ensemble learning, usually with a considerable level of success compared with classical methods. It is not possible to review or validate all the instances listed in the literature, but I will highlight a small subset of such applications from different categories, which should provide a good perspective on the variety and comparative performance of ensemble models. These highlighted examples can be grouped into the following (loosely defined, somewhat overlapping) three categories: portfolio management, volatility forecasting and option pricing, and miscellaneous applications.

Portfolio Management

Portfolio management encompasses various activities related to investment selection, portfolio construction, and asset allocation optimization in order to meet the short- and long-term financial objectives and risk tolerance of an investor or an institution. This data-intensive process typically involves analyzing factors and historical returns, estimating model parameters, forecasting risk premium or conditional expected stock returns in excess of the risk-free rate, and occasionally estimating portfolio weights.

Cross-Section of Stock Returns

Asset return prediction and the identification of predictors of returns are among the fundamental topics in asset pricing research. Although traditional factor models—such as the CAPM and its extensions, including the three-factor and five-factor Fama–French models—are widely used to explain historical returns of diversified portfolios, practitioners consider them empirically inadequate to predict returns under various market regimes. Subsequently, a plethora of factors (the "factor zoo") have been proposed as candidate predictors of cross-sectional variations in expected returns.

Among the various groups of such factors are financial, macro, microstructure, behavioral, and accounting factors, many of which may be viewed as belonging to the broader categories of common sources of risk or idiosyncratic characteristics that pertain to an individual firm or portfolio (Harvey, Liu, and Zhu 2016). Naturally, one may ask which one of these factors provides effective and independent information about the cross-section of returns. Here, the emphasis is placed on efficacy and independence within the context of the inadequacy of simple linear models in out-of-sample predictions in the presence of multicollinearity and noise.

Literature reviews point to the growing adoption of machine learning methods in empirical asset pricing to address the problem of the factor zoo (Bagnara 2024). These methods primarily include regularization and dimension reduction as processing steps, followed by the use of random forests, neural networks, and comparative analyses that typically involve ensemble methods. Among the important rationales stated for using machine learning in asset pricing are "return prediction is economically meaningful" and "relative to traditional empirical methods in asset pricing, machine learning accommodates a far more expansive list of potential predictor variables and richer specifications of functional form" (Gu et al. 2020, p. 2224).

Ensemble methods, in particular, have been found to provide superior results over traditional linear factor models in reducing the risk of overfitting, thanks to the forecast combination mechanism. Among the various instances of forecast combination mechanisms are combining forecasts from different classes of algorithms, combining forecasts based on different training windows, combining forecasts that use different factor libraries, and combining forecasts for different horizons to increase forecast diversity and reduce the risk of overfitting (Rasekhschaffe and Jones 2019).

It is important to point out that the superiority of a method is evaluated by quantifying its performance over a "testing" subsample that has been held out of the training sample (used for estimation) and the validation sample (used for model tuning). These testing data are truly out of sample and may reflect a method's predictive performance more realistically. The metrics used for quantifying model performance include the following:

- Standard algorithm performance evaluation metrics, such as the coefficient of determination, R^2; root mean square error; and mean absolute percentage error
- Aggregate portfolio performance metrics, such as mean and standard deviation of returns, cumulative return, maximum drawdown, and information ratio for the risk-adjusted return related to a benchmark

The algorithm performance evaluation metrics are estimated on the basis of a direct comparison of the realized versus predicted returns of the testing period. In contrast, aggregate portfolio performance metrics may be estimated on the basis of out-of-sample (e.g., one-month-ahead) stock return predictions of machine learning, going long on the top decile of stocks with the highest predicted return stocks, going short on the bottom decile, and calculating the backtest returns. Following Gu et al. (2020), the next-period expected return $r_{i,t+1}$ of stock i can be modeled as

$$\begin{cases} r_{i,t+1} = E_t(r_{i,t+1}) + \varepsilon_{i,t+1} \\ \text{s.t. } E_t(r_{i,t+1}) = L\left(X_{i,t}; \bar{h}\right), \end{cases}$$

where L is a (machine) learning algorithm acting on the p-dimensional vector, $X_{i,t}$ represents current-period features (factors), and $\bar{h}$ is the algorithm's tunable hyperparameters—for example, the number of trees or the maximum tree depth in a random forest. The OLS regression is a special case of this equality when L is a linear unbiased estimator for which the error terms, $\varepsilon_{i,t+1}$, have zero mean and constant variance and are uncorrelated.

The OLS estimator is an optimal model among linear unbiased estimators because of its lowest variance and is traditionally a popular choice. With a large set of candidate predictors, however, estimating an OLS regression may pose computational challenges and yield unreliable

predictions. First, a linear model is restricted in its functional form and may not detect interactions among features and their nonlinear dynamics. Second, when the number of features increases, there is an elevated risk of in-sample spurious correlation and overfitting. In addition, multicollinearity may be strong in the presence of many features, resulting in unreliable parameter estimates, because of high correlation or linear dependence among regressors.

Machine learning models are desirable when traditional linear models are not feasible, because they can be used for high-dimensional feature sets containing both cross-sectional firm characteristics and macroeconomic factors. Additionally, they can be regularized to reduce overfitting, and they efficiently search and select among many model specifications.

This notion has been demonstrated by Gu et al. (2020). The authors carried out a large-scale study on monthly total individual equity return data spanning 60 years, 1957–2016 (with the latest 30 years as out-of-sample testing). The studied data contain more than 30,000 stocks (6,000+ per month) for firms listed on the NYSE, AMEX, and NASDAQ. The Treasury bill rate was used for the risk-free rate to calculate individual excess returns, and for stock-level predictors, the authors used more than 90 characteristics based on the cross-section of stock returns, as well as industry dummies for Standard Industrial Classification codes and several macroeconomic predictors.

Gu et al. (2020) found that shallow neural networks and ensembles of regression trees are the best-performing methods across a variety of portfolio constructions (see **Exhibit 5** and **Exhibit 6**). They attributed this performance to the detection of nonlinear interactions missed by other methods and effective "shallow" learning (compared with "deep" learning) resulting from the scarcity of data and the low signal-to-noise ratio in asset pricing. The authors also reported greater success from ML in forecasting larger and more liquid stock returns and portfolios and observed agreement among ML models on a small set of dominant predictive signals, including price trends (return reversal and momentum), measures of stock liquidity, stock volatility, and valuation ratios.

Risk Factor Analysis

Machine learning can enrich risk analysis by providing insight into relationships between variables that are unaccounted for in more-traditional factor models. Several categories of "factor model" exist, including macroeconomic, statistical, and fundamental models (Connor 1995). In such models, the security return is the dependent variable, and the risk factors,

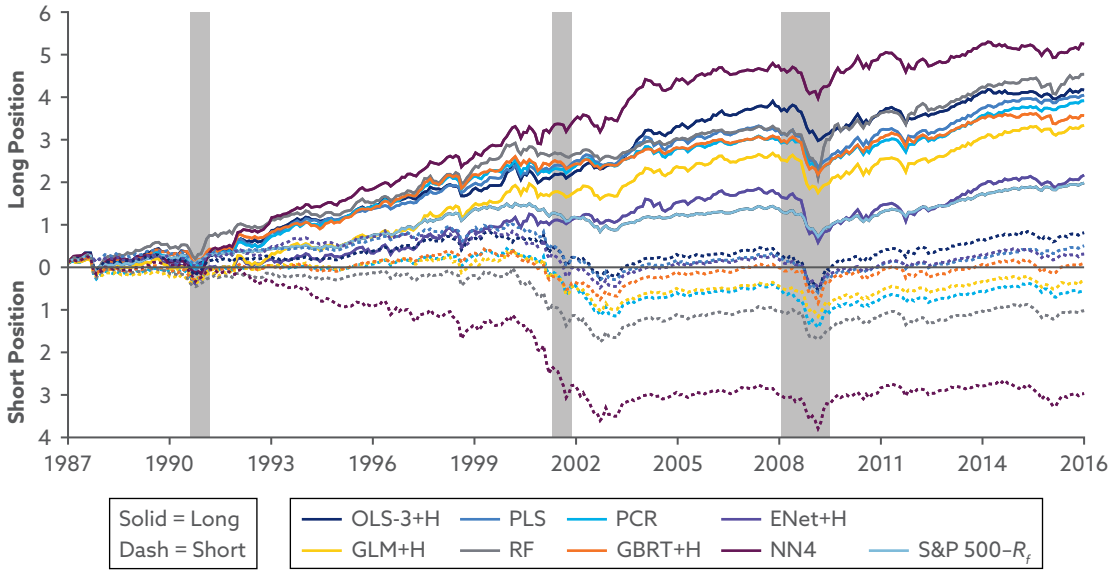
Exhibit 5. Performance of ML Portfolios

	OLS	Elastic Net	RF	GBRT	NN4
Mean return	1.34	2.11	2.38	2.14	3.33
Maximum drawdown percentage	84.74	33.70	46.42	37.19	14.72
Information ratio	1.30	0.93	1.30	1.60	2.40

Note: RF stands for random forest, GBRT stands for gradient boosted regression tree, and NN4 stands for four-hidden-layer neural network.

Source: Excerpts from Table 8 of Gu et al. (2020), used with permission.

Exhibit 6. Cumulative Log Returns of ML Portfolios Sorted on Out-of-Sample Forecasts (National Bureau of Economic Research recessions are shaded)



Source: Figure 9 of Gu et al. (2020), used with permission.

including macroeconomic variables and security characteristics, are the independent variables. Traditionally, estimation methods are based on time-series and/or cross-sectional regression, with OLS regression being the prevalent approach. The main assumptions of these models are that the selected group of independent variables adequately explains portfolio behavior and the factors have a low correlation for the model to be stable and statistically valid.

These assumptions may be easily violated under different market regimes, however, posing challenges to the estimation of the model parameters and explanation of the results. For example, as the dimensionality of the list of candidate factors grows to account for additional exposure, exacerbated by dramatic changes in factor relationships during major market events, both linear and parametric nonlinear models, which often heavily depend on sample data, become hard to estimate and are rendered inadequate and unstable.

As a remedy to these drawbacks, Simonian, Wu, Itano, and Narayanam (2019) developed a machine learning framework to uncover nonlinear relationships, discontinuities (e.g., threshold correlations), and interaction effects between factors in the well-known Fama–French–Carhart (FFC) equity factor model. These authors showed how the RF model provides a higher level of explanatory power relative to more commonly used frameworks, such as CAPM and FFC. They also provided useful information regarding the sensitivity of assets to factors by leveraging the importance of features to derive pseudo-betas. Additionally, the authors combined this approach with association rule learning to build a sector-rotation trading strategy. They showed that their strategy, using the RF-predicted return of a sector and the ratio of

short-term to long-term realized volatility as trading signals, outperforms the "no information" equal-weight portfolio, as measured by various out-of-sample performance metrics. The authors argue that this approach provides an effective way to inform both risk analysis and portfolio management.

Volatility Forecasting and Option Pricing

Quantitative finance, also known as financial mathematics, refers to the advanced mathematical models used for the pricing of derivative securities and portfolio risk management. A subcategory of quantitative finance is financial engineering, which uses computational simulations for pricing, trading, hedging, and other activities. This is a data-intensive process in which machine learning can be readily used.

Volatility Forecasting

An accurate measurement of volatility is of paramount importance in financial risk management, asset pricing, derivative pricing, and estimating expected returns of portfolios. Traditionally, volatility estimation models have relied on a handful of statistical inference methods and a limited group of predictors. In light of the availability of high-frequency price data, growing sets of candidate variables, and scalable computing power, machine learning methods enable researchers and practitioners to move beyond single-threaded forecasting methods typically dominated by a small set of particular features or algorithms and to develop scalable systems of volatility prediction.

Such systems can offer reduced human intervention during the selection of features and models, accommodating an expanded set of features and controlling overfitting through automated calibration. Notably, Li and Tang (2024) recently proposed an automated volatility forecasting system for the S&P 100 universe that leverages more than 100 features and five ML algorithms to predict the response variable of the realized variance (RV), an estimator of the quadratic variation of the log price process over a given period, using a set of features that include multiple realized features based on RV-based models, such as heterogeneous autoregressive (HAR), semivariance HAR (SHAR), and mixed data sampling (MIDAS).

In particular, after evaluating the out-of-sample performance of five learning algorithms—LASSO (least absolute shrinkage and selection operator), principal component regression (PCR), random forest (RF), gradient boosted regression trees (GBRT), and two-hidden-layer neural network (NN2)—the authors also constructed a simple average ensemble to combine all machine learning algorithms. The reported ensemble delivered superior performance across various forecast horizons; the automated volatility forecasting system acts dynamically and becomes increasingly powerful over time (**Exhibit 7**). For example, "in terms of utility gains, the ensemble model delivers 44 more bps per year to the mean–variance investor relative to using OLS for this large stock universe" (Li and Tang 2024, pp. 82–83).

Option Pricing

The motivation to apply machine learning methods for pricing options and derivatives is driven by the computational estimation challenges and limitations of classical parametric models. By considering options as functional mappings between the contracted terms' inputs (e.g., underlying asset and time to maturity) and the premium output (e.g., a fixed deviation

Exhibit 7. Out-of-Sample R^2 Relative to the Historical Mean of Realized Volatilities for OLS-Based Models

Model	Out-of-Sample R^2 Relative to Long-Run Mean			
	Daily	Weekly	Monthly	Quarterly
HAR	0.58	0.69	0.70	0.64
MIDAS	0.58	0.71	0.71	0.64
SHAR	0.58	0.70	0.70	0.64
OLS	0.61	0.73	0.72	0.63
LASSO	0.61	0.73	0.73	0.65
PCR	0.60	0.71	0.72	0.67
RF	0.59	0.71	0.73	0.66
GBRT	0.60	0.73	0.73	0.66
NN2	0.62	0.75	0.74	0.65
Ensemble	0.62	0.74	0.75	0.67

Source: Excerpts from Tables A.3 and A.4 of Li and Tang (2024), used with permission.

from a target price), data-driven and machine learning methods can be applied (Hutchinson, Lo, and Poggio 1994). In a study of European call options with West Texas intermediate (WTI) crude oil futures contracts as an underlying asset (Ivaşcu 2021), machine learning algorithms, particularly ensemble methods, such as XGBoost and LightGBM, largely outperformed classic methods, such as Black–Scholes and Corrado–Su, with both historical and implied volatility parameters.

Bid–Ask Spread

The foreign exchange (FX) market is both the largest and the most liquid financial market in the world. In particular, because of the uneven spread of liquidity across different currencies and different market times and conditions, predicting the likely cost of trading currencies as measured by bid–ask spreads is crucial for profitable trading. As such, the FX market is a fertile playground for data-driven and ML methods seeking to identify and benefit from market microstructures.

Recently, Kirkby and Andrean (2024) identified various predictors of daily bid–ask spreads, including temporal features, such as the day of the week and the hour of day and their interactions; various spreads, lags, and moving averages; and rate volatility, capturing the interplay between market volatility and bid–ask spreads and the spillover of volatility between related markets. They observed that from among supervised machine learning algorithms, subject to a modest level of hyperparameter tuning, the random forest model had the best out-of-sample performance, with an economically meaningful ability to forecast the next day's spread with a relative error of less than 20% on average.

Miscellaneous Applications

In addition to the previous examples, other applications of ensemble learning published in the financial data science literature include the following:

- Default prediction of commercial real estate properties and identification of the ratio of the capitalization rate spread to the average capitalization rate spread of property type as a top driver of defaults (Cowden, Fabozzi, and Nazemi 2019)
- Predictability of stock market bubbles and large near-term drawdowns based on patterns in stock price behavior (Simonian 2022)
- Prediction and estimation of the likelihood of recession in a given month using historical macroeconomic factor time series (Yazdani 2020)
- Comparing long short-term memory and gradient boosting algorithms in predicting the returns of S&P 500 constituents using technical indicators and principal component analysis (Cvetkovic 2024)

This list provides only a sample of the published literature on ensemble modeling, and many more may be identified and explored.

Explainability of Ensemble Methods

Broadly speaking, explainability, also known as interpretability, refers to processes used to understand how a machine learning model makes decisions and what its output means. Explainability is a way to increase reliability in the presence of uncertainty and reduce the risk of overfitting and irrelevant and unfair decision making. For a comprehensive study of explainability methods, see Molnar (2019). The following are some common explainability techniques:

- "Partial dependency plots show the marginal effect of an input feature on the predicted outcome."[1]
- The SHAP (SHapley Additive exPlanations) technique enables estimation and visualization of each input feature's contribution to the output (based on "Shapley values [that] measure the average marginal contribution of a feature across all possible combinations of features)."[2]
- Feature importance can be used to estimate the importance of a feature for the model, based on measuring the performance decrease when randomly shuffling the feature value across all samples.
- The LIME (local interpretable model-agnostic explanations) technique can be used to locally approximate a model's outputs with a simpler, interpretable model.
- Counterfactual explanations describe the smallest change to the feature values that change the prediction to a predefined outcome.
- Other methods—such as conformal prediction intervals (Vovk, Gammerman, and Shafer 2022)—that provide reliable estimates, even in highly unpredictable markets, are crucial for risk management and decision making under uncertainty.

---

[1] https://en.wikipedia.org/wiki?curid=54575571.

[2] https://en.wikipedia.org/wiki?curid=54575571.

In line with applications of ensemble models for financial forecasting and predictive modeling, explainability is used increasingly to demystify the prediction by such models. For example, Li et al. (2020) used ML to predict one-month spot returns of a portfolio of major currencies and subsequently developed a framework for demystifying ML model predictions called the "model fingerprints." This approach draws on partial dependence to identify feature interactions and estimate the average marginal effect when features have low correlations.

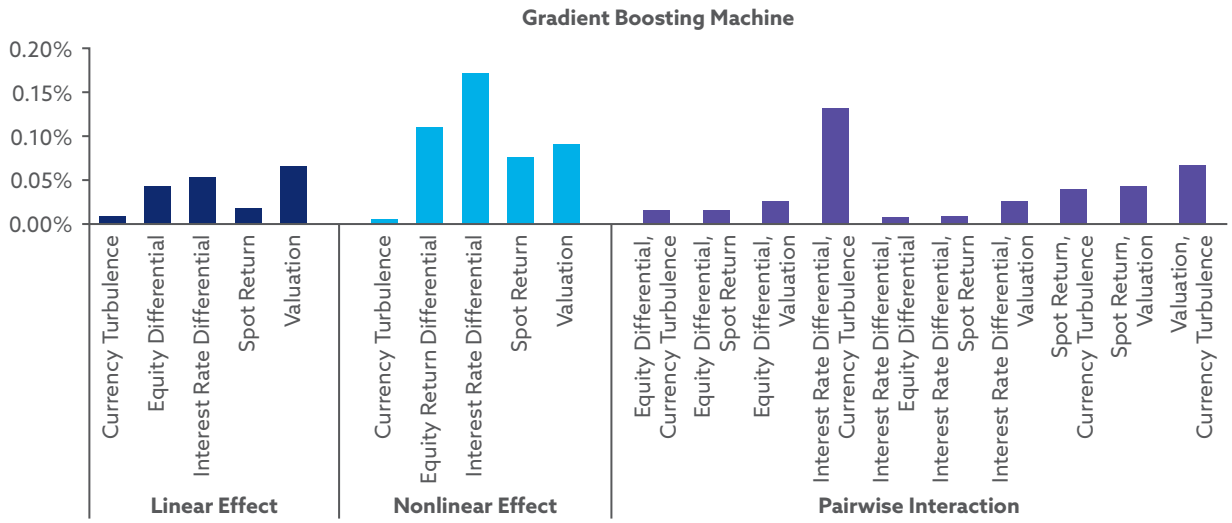
In particular, the model fingerprint decomposes predictions into the contributions from the following four sources: linear effect (*L*) for the time series returns of portfolios formed from linear predictions in isolation; pairwise interactions (*P*) for returns of portfolios formed from the combined linear and pairwise interaction predictions minus the returns from linear effect; nonlinear effect (*N*) for returns of portfolios from the combined linear, nonlinear, and pairwise interaction predictions minus the returns from linear and pairwise effects; and higher-order effects (*H*). Mathematically, for a model prediction function $\hat{y} = \hat{f}(x_1, \ldots, x_m)$ dependent on *m* input variables, this decomposition can be written as

$$\hat{y} = \sum_{i=1}^{m} L(x_i) + \sum_{i=1}^{m} N(x_i) + \sum_{i \neq j}^{m} P(x_i, x_j) + \sum_{i=1}^{m} H(x_i).$$

An example of such decomposition for the gradient boosting model prediction is shown in **Exhibit 8**. The model fingerprint method, when accompanied by other local and global model interpretability tools such as feature importance and SHAP force plots for attribution analysis, yields explanations for the model outcome, which is useful for comparing and interpreting various investment strategies.

Among other notable examples of ML model explainability is the framework developed by Davis, Lo, Mishra, Nourian, Singh, Wu, and Zhang (2023) in which the authors used ML models

Exhibit 8. Decomposition of Gradient Boosting Model Predictions Using the Fingerprint Method



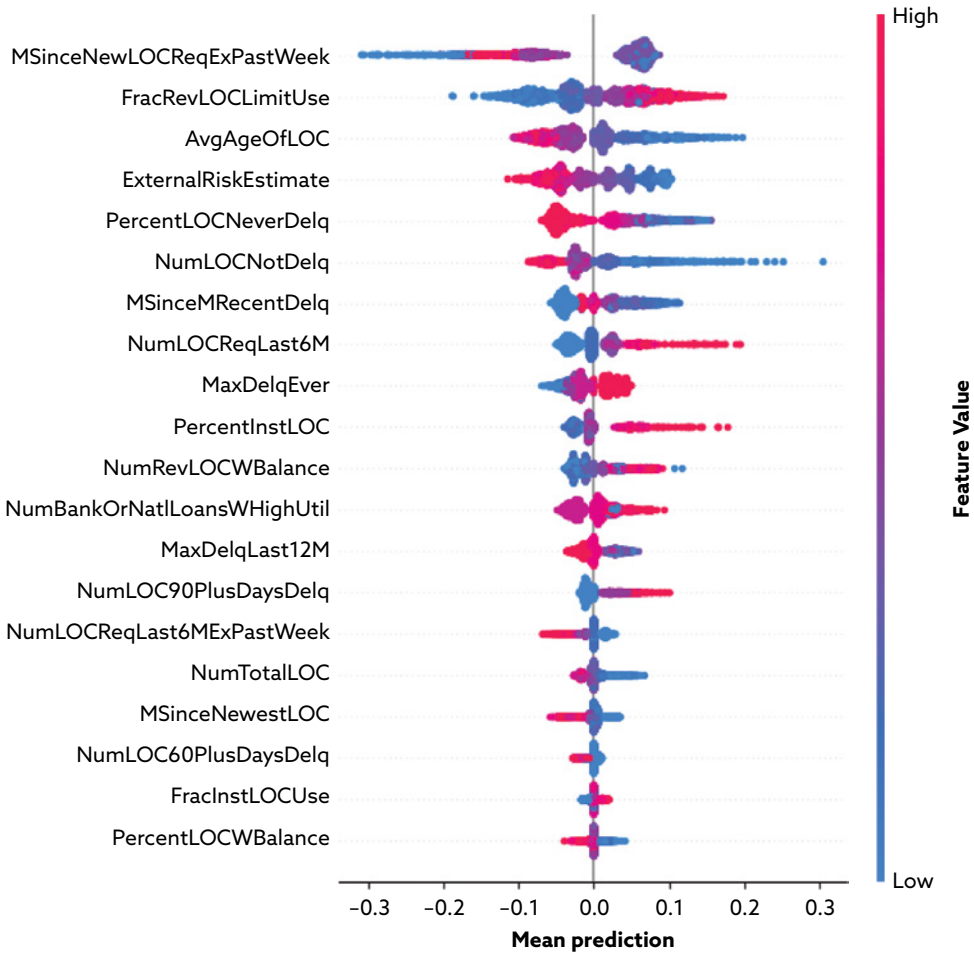
Source: Middle panel in Exhibit 3 of Li et al. (2020), used with permission.

to forecast individuals' home equity credit risk using a real-world dataset. The authors demonstrated methods to explain these ML models' output and to make them more accessible to end users. Notably, the authors developed different means of explainability to accommodate different requirements of the various stakeholders, including explanations for predictions of creditworthiness for loan companies (see **Exhibit 9**), stress tests for regulators, guiding counterfactuals for loan applicants, and explanatory rules for data scientists.

Summary and Future Directions

With the rapid development of artificial intelligence, which has led to the empowerment and automation of machine learning processes, one can expect further growth and adoption of data-driven methods in financial and investment decision-making processes. In particular, the

Exhibit 9. SHAP Feature Importance for the Probability of Default



Note: The larger values (in red) decrease the probability of default, while the smaller values (in blue) increase the probability of default.

Source: Figure 3(a) of Davis et al. (2023), used with permission.

emergence of large language models (LLMs) and generative AI, capable of generating a reasonable first-pass analysis of financial data and accompanying computer codes, is anticipated to further increase the adoption of data-driven methods. In addition, researchers are investigating methods to combine multiple LLMs (LLM ensembles) to enhance overall performance. Despite all of these attempts and because of the black-box perception associated with many ML and AI methods, key considerations remain in some critical applications of such methods—in particular, those involving high-risk decisions, fairness, and transparency. Frameworks that carefully blend both traditional and modern frameworks—including explanatory, explainability, and predictive—while drawing on the domain knowledge, may have a greater chance of success and survival in an environment typically characterized by the existence of competing agents, high noise-to-signal ratios, and rare arbitrage opportunities.

Acknowledgment

I would like to thank Tod McKenna for his support.

Disclaimer

The views expressed are those of the author and do not reflect the views of Citi or any of its affiliates. Any statements made have not been verified by Citi for accuracy and completeness.

References

Bagnara, Matteo. 2024. "Asset Pricing and Machine Learning: A Critical Review." *Journal of Economic Surveys* 38 (1): 27–56. doi:10.1111/joes.12532.

Bates, John M., and Clive W. J. Granger. 1969. "The Combination of Forecasts." *Journal of the Operational Research Society* 20 (4): 451–68. doi:10.1057/jors.1969.103.

Breiman, Leo. 2001. "Random Forests." *Machine Learning* 45 (1): 5–32. doi:10.1023/A:1010933404324.

Chen, Tianqi, and Carlos Guestrin. 2016. "XGBoost: A Scalable Tree Boosting System." In *KDD '16: Proceedings of the 22nd SIGKDD Conference on Knowledge Discovery and Data Mining*, 785–94. New York: Association for Computing Machinery. doi:10.1145/2939672.2939785.

Connor, Gregory. 1995. "The Three Types of Factor Models: A Comparison of Their Explanatory Power." *Financial Analysts Journal* 51 (3): 42–46. doi:10.2469/faj.v51.n3.1904.

Cowden, Chad, Frank J. Fabozzi, and Abdolreza Nazemi. 2019. "Default Prediction of Commercial Real Estate Properties Using Machine Learning Techniques." *Journal of Portfolio Management* 45 (7): 55–67. doi:10.3905/jpm.2019.1.104.

Cvetkovic, Miljana. 2024. "LSTM versus Gradient Boosting for Asset Pricing: A Case Study of the S&P 500." *Journal of Financial Data Science* 6 (4): 49–71. doi:10.3905/jfds.2024.1.169.

Davis, Randall, Andrew W. Lo, Sudhanshu Mishra, Arash Nourian, Manish Singh, Nicholas Wu, and Ruixun Zhang. 2023. "Explainable Machine Learning Models of Consumer Credit Risk." *Journal of Financial Data Science* 5 (4): 9–39. doi:10.3905/jfds.2023.1.141.

Flach, Peter. 2012. *Machine Learning: The Art and Science of Algorithms That Make Sense of Data*. Cambridge, UK: Cambridge University Press. doi:10.1017/CBO9780511973000.

Freund, Yoav, and Robert E. Schapire. 1996. "Experiments with a New Boosting Algorithm." In *Proceedings of the Thirteenth International Conference on Machine Learning (ICML'96)*, 148–56. San Francisco: Morgan Kauffman Publishers. https://cseweb.ucsd.edu/~yfreund/papers/boostingexperiments.pdf.

Friedman, Jerome H. 2001. "Greedy Function Approximation: A Gradient Boosting Machine." *Annals of Statistics* 29 (5): 1189–1232. www.jstor.org/stable/2699986.

Géron, Aurélien. 2019. *Hands-On Machine Learning with Scikit-Learn, Keras, and Tensorflow*, 2nd ed. Sebastopol, CA: O'Reilly Media.

Gu, Shihao, Bryan Kelly, and Dacheng Xiu. 2020. "Empirical Asset Pricing via Machine Learning." *Review of Financial Studies* 33 (5): 2223–73. doi:10.1093/rfs/hhaa009.

Harvey, Campbell R., Yan Liu, and Heqing Zhu. 2016. "... and the Cross-Section of Expected Returns." *Review of Financial Studies* 29 (1): 5–68. doi:10.1093/rfs/hhv059.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman. 2017. *The Elements of Statistical Learning*, 2nd ed. New York: Springer.

Hutchinson, James M., Andrew W. Lo, and Tomaso Poggio. 1994. "A Nonparametric Approach to Pricing and Hedging Derivative Securities via Learning Networks." *Journal of Finance* 49 (3): 851–89. doi:10.1111/j.1540-6261.1994.tb00081.x.

Ivaşcu, Codruț-Florin. 2021. "Option Pricing Using Machine Learning." *Expert Systems with Applications* 163 (January). doi:10.1016/j.eswa.2020.113799.

Ke, Guolin, Qi Meng, Thomas Finely, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tia-Yan Liu. 2017. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree." In *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, 3149–57.

Kirkby, Justin L., and Victor Andrean. 2024. "Forecasting Bid–Ask Spreads in Foreign Exchange: Analysis and Machine Learning Prediction." *Journal of Financial Data Science* 6 (4): 61–75. doi:10.2139/ssrn.4701477.

Li, Sophia Zhengzi, and Yushan Tang. 2024. "Automated Volatility Forecasting." Working paper (3 April). doi:10.2139/ssrn.3776915.

Li, Yimou, David Turkington, and Alireza Yazdani. 2020. "Beyond the Black Box: An Intuitive Approach to Investment Prediction with Machine Learning." *Journal of Financial Data Science* 2 (1): 61–75. doi:10.3905/jfds.2019.1.023.

López de Prado, Marcos. 2019. "Beyond Econometrics: A Roadmap Towards Financial Machine Learning." Working paper (19 September). doi:10.2139/ssrn.3365282.

Molnar, Christoph. 2019. *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. https://christophm.github.io/interpretable-ml-book.

Rasekhschaffe, Keywan Christoph, and Robert C. Jones. 2019. "Machine Learning for Stock Selection." *Financial Analysts Journal* 75 (3): 70–88. doi:10.1080/0015198X.2019.1596678.

Shapley, Lloyd S. 1953. "A Value for *n*-Person Games." In *Contributions to the Theory of Games, Volume II*, edited by H. W. Kuhn and A. W. Tucker, 307–17. Princeton, NJ: Princeton University Press. doi:10.1515/9781400881970-018.

Simonian, Joseph. 2022. "Forests for Fama." *Journal of Financial Data Science* 4 (1): 145–57. doi:10.3905/jfds.2021.1.086.

Simonian, Joseph, and Frank J. Fabozzi. 2019. "Triumph of the Empiricists: The Birth of Financial Data Science." *Journal of Financial Data Science* 1 (1): 10–13. doi:10.3905/jfds.2019.1.010.

Simonian, Joseph, Chenwei Wu, Daniel Itano, and Vyshaal Narayanam. 2019. "A Machine Learning Approach to Risk Factors: A Case Study Using the Fama–French–Carhart Model." *Journal of Financial Data Science* 1 (1): 32–44. doi:10.3905/jfds.2019.1.032.

Vovk, Vladmir, Alexander Gammerman, and Glenn Shafer. 2022. *Algorithmic Learning in a Random World*, 2nd ed. London: Springer. doi:10.1007/978-3-031-06649-8.

Wang, Xiaoqian, Rob J. Hyndman, Feng Li, and Yanfei Kang. 2023. "Forecast Combinations: An Over 50-Year Review." *International Journal of Forecasting* 39 (4): 1518–47. doi:10.1016/j.ijforecast.2022.11.005.

Wolpert, David H. 1992. "Stacked Generalization." *Neural Networks* 5 (2): 241–59. doi:10.1016/S0893-6080(05)80023-1.

Wolpert, David H. 1996. "The Lack of A Priori Distinctions between Learning Algorithms." *Neural Computation* 8 (7): 1341–90. doi:10.1162/neco.1996.8.7.1341.

Wolpert, David H. 2002. "The Supervised Learning No-Free-Lunch Theorems." In *Soft Computing and Industry*, edited by Rajkumar Roy, Mario Köppen, Seppo Ovaska, Takeshi Furuhashi, and Frank Hoffmann. London: Springer. doi:10.1007/978-1-4471-0123-9_3.

Yazdani, Alireza. 2020. "Machine Learning Prediction of Recessions: An Imbalanced Classification Approach." *Journal of Financial Data Science* 2 (4): 21–32. doi:10.3905/jfds.2020.1.040.