The Journal of Alternative Investments

Forecasting ETFs with Machine Learning Algorithms

Jim Kyung-Soo Liew and Boris Mayster

JAI 2017, 20 (3) 58-78
doi: https://doi.org/10.3905/jai.2018.20.3.058
http://jai.iijournals.com/content/20/3/58

Institutional Investor Journals
1120 Avenue of the Americas, 6th floor,
New York, NY 10036, Phone: +1 212-224-3589

Forecasting ETFs with Machine Learning Algorithms

Jim Kyung-Soo Liew and Boris Mayster

Jim Kyung-Soo Liew
is an assistant professor in finance at the Johns Hopkins Carey Business School in Baltimore, MD, and is the founder of www.SoKat.com.
kliew1@jhu.edu

Boris Mayster
is a quantitative analyst at EPFC in Kaliningrad, Russia.
boris.mayster@jhu.edu

Machine learning and artificial intelligence (AI) algorithms have come home to roost. These algorithms, whether we like it or not, will continue to permeate our daily lives. Nowhere is this more evident than in their current uses in self-driving cars, spam filters, movie recommendation systems, credit fraud detection, geo-fencing marketing campaigns, and so forth. The usage of these algorithms will only expand and deepen going forward. Recently, Stephen Hawking issued a forewarning: "The automation of factories has already decimated jobs in traditional manufacturing, and the rise of AI is likely to extend this job destruction deep into the middle classes" (Price [2016]). Whether we agree or disagree with the virtues of automation, the only way to better utilize its potentials and evade its dangers is to gain a deeper knowledge and appreciation of these algorithms. Moreover, quite nearly upon us is the next big wave called the *Internet of Things* (IoT), whereby increasingly more devices and common household items will be interconnected and stream terabytes of data. As our society is deluged with data, the critical question that emerges is whether machine learning algorithms contribute a net benefit to or extract a net cost from society.

While the future looms large for machine learning and AI, one pocket of their development appears to have been deliberately left behind—namely, in finance and more so in hedge funds that attempt to predict asset prices to generate alpha for their clients. The reason is clear: One trader's gain in applying a well-traded learning algorithm is another's loss. This edge becomes a closely guarded secret, in many cases, defining the hedge fund's secret sauce. In this work, we investigate the benefits of applying machine learning algorithms to this corner of the financial industry, which academic researchers have left unexamined.

We are essentially interested in understanding whether machine learning algorithms can be applied to predicting financial assets. More specifically, the goal is to program an AI to throw off profits by learning and besting other traders and, possibly, other machines. This goal is rumored to have already been achieved by a hedge fund, Renaissance Technology's Medallion fund. The Medallion fund, cloaked in mystery to all but a few insiders, has generated amazing performance over an extended time period. Renaissance Technology clearly has access to enough brain power to be on the cutting edge of any machine learning implementation, and as much as others have tried to replicate their success, Medallion's secret sauce recipe has yet to be cracked. In this article, we attempt to unravel several potential research paths in an attempt to shed light on how machine learning algorithms could be employed to trade financial market assets. We employ machine learning algorithms to build and test models of prediction of asset price direction for several well-known and highly liquid exchange-traded funds (ETFs).

Markets are efficient or, at a minimum, at least semi-strong form efficient. All publicly available information is reflected in the stock prices, and the pricing mechanism is extremely quick and efficient in processing new information sets. Attempting to gain an edge is nearly impossible, especially when one tries to process widely accessible public information. Investors are therefore better off holding a well-diversified portfolio of stocks. Clearly, Fama would side with those who have little faith in the ability of these machine learning algorithms to process known publicly available information and, with such information, gain an edge by trading.

Many researchers have documented evidence that asset prices are predictable. Jegadeesh and Titman [1993] and Rouwenhorst [1998] showed that past prices help predict returns. Fama and French [1992, 1993, 1995] showed that the fundamental factors of book-to-market and size affect returns. Liew and Vassalou [2000] documented the predictability of fundamental factors and linked these factors to future real gross domestic product growth. Keim [1983] documented seasonality in returns, with more pronounced performance in January. For more recent evidence on international predictability, see Asness, Moskowitz, and Pedersen [2013]. Whether the predictability stems from suboptimal behavior along the reasoning of Lakonishok, Shleifer, and Vishny [1994]; limits to arbitrage by Shleifer and Vishny [1997]; or some unidentified risk-based explanation by Fama and French [1992, 1993], it nonetheless appears that predictability exists in markets.

We employ the most advanced machine learning algorithms, namely, deep neural networks (DNNs), random forest (RF), and support vector machines (SVMs). Our results are generally similar across the algorithms employed, with a slight advantage for RF and SVMs over DNNs. We report results for the three distinct algorithms and are interested in predicting price changes in 10 ETFs. These ETFs were chosen for their popularity as well as their liquidity, and their historical data were sourced from Yahoo Finance. Because we are interested in predicting the change in prices over varying future periods, we employ daily data. The horizons that we attempt to predict range from trading days to weeks and months.

We test several information sets to determine which sets are most important in predicting across differing horizons. Our information sets are based on (A) prior prices, (B) prior volume, (C) dummies for days of the week and months of the year, and (ABC) all our information sets combined. We find that (B) volume is very important for predicting across the 20- to 60-day horizon. Additionally, we document that each feature has very low predictability, so we recommend that model builders use a wide range of features guided by financial experiences and intuition. Our methodology was constructed to be robust and allow for easy switching and testing of different information set specifications and securities.

The next section describes the procedures we employed and the assumptions made in this work, with a focus on applying best practices when applicable. Afterward, we discuss the machine algorithms employed. We then move into the details of our methodology and present our main results. Finally, we present our thoughts on implementation, weaknesses in our approach, implications, and conclusions.

PROCEDURE WITH EMBEDDED BEST PRACTICES

Machine learning algorithms are extremely powerful, and most can easily overfit any dataset. In machine learning parlance, overfitting is known as *high variance.* Fitting an overly complex model on the training set does not perform well out of sample or in the test set. Prior to the introduction of cross-validation, researchers would rely on their best judgment as to whether a model was overfit. Currently, the best practices for building a machine learning predictive model are based on the holdout cross-validation procedure.

We therefore employ the holdout cross-validation procedure in our article. We split the data into three components: the training set, the validation set, and test set. Best practices state that we should construct the model on the training set and validation set only and use the test set once. Repeatedly using the training set and validation set on that part of the data that does not contain the test set is known as holdout cross-validation. Although cross-validation is readily employed in many other fields, some criticize its use in financial time-series data. Nonetheless, we believe our results are still interesting because we are investigating the foundational question of whether using machine learning algorithms works in modeling changes in prices.

The Irksome Stability Assumption

Arguably the most famous cross-sectional relationship is the capital asset pricing model (CAPM), which states that the expected return on any security is equal to the risk-free rate of returns plus a risk premium. The CAPM's risk premium is defined as the security's beta multiplied by the excess return on the market. Market observability has been changed by Roll's [1976] critique; however, generally speaking, the theoretical CAPM has become a mainstay in academics as well as in practice. To estimate beta, students are taught to run a time-series linear regression of the excess return of a given security on the excess return on the market. The covariance of a security and the market provides for the fundamental building block on which the CAPM has been constructed.

In this work, however, breaking from some finance tradition, we view predictability disregarding the time-series structure. We make the irksome stability assumption that there is a stable relationship between predicting price changes and the many features employed across our information sets. That is, like modeling credit card fraud and email spam, we assume that the relationship between the response and features is independent of time. For example, we allow our algorithms to capture the relationship that maps the feature input matrix (X) to the output responses (y). With that said, the features are always known prior to the change in prices.

To sum, we incorporate the current best practices of balancing the overfitting (or high-variance) problem with the underfitting (or high-bias) problem. This is accomplished by separating the training sample and performing k-fold cross-validation on the training sample and employing the test sample only once. In this work, we adhere to this best practice when applicable.

We attempt to use machine learning algorithms to answer the following questions:

1. What is the optimal prediction horizon for ETFs?
2. What are the best information sets for such prediction horizons?

Because we have a dependent variable (y) as future price movements, either up or down, in this work we are dealing with a supervised learning problem. The true value of the dependent variable is known a priori. We can also test the accuracy of our forecasts. Accuracy is measured by the percentage of times the model predicts correctly over the total number of predictions. Although we could choose from a vast number of algorithms, we restrict our analysis to the following three powerful and popular algorithms: DNNs, RFs, and SVMs.

DNNs

DNNs are defined by neural networks with more than one hidden layer. Neural networks are composed of *perceptrons*, first introduced by Rosenblatt [1957], who built on the prior work of McCulloch and Pitts [1943]. McCulloch and Pitts introduced the first concept of a simplified brain cell: the McCulloch–Pitts neuron. Widrow and Hoff [1960] improved upon Rosenblatt's [1957] work by introducing a linear activation function, thus allowing the solutions to be cast in the minimization of the *cost function*. The cost function is defined as the sum of squared errors, with *errors* in the context of a supervised machine learning algorithm defined as the predicted or hypothesized value minus the true value. The advantage of this setting allows for the change of only the activation function to yield different techniques. Setting the activation function to either the logistic or hyperbolic tangent allows us to arrive at the multilayered neural network. If the networks have more than one hidden layer, we arrive at our deep artificial neural network (see Raschka [2015]).

The parameters or weights in our DNN setting are determined by gradient descent. The process consists of initializing the weights across the neural network to small random numbers and then forward propagating the weights throughout the network. At each node, the weights and input data are multiplied and aggregated, then sent through the prespecified activation function. Prior layers are employed as input into the next layer and repeated. Once the errors have been forward propagated throughout the network, backward propagation is employed to adjust the weights. Weights are adjusted until some maximum number of iterations has been met or some minimum limit of error has been achieved. It should be noted that the reemergence of neural networks can be attributed to backward propagation contribution, which allowed for a much quicker convergence to the optimal parameters. Without backward propagation, this technique would have taken too long for convergence and would remain much less popular.

In our analysis, we employ a DNN algorithm (i.e., a neural network with more than one hidden layer). Between the input and output layers are the hidden layers. We employ two- and three-hidden-layer neural networks and thus a DNN in this work. Recently, a deep learning neural network beat the best human champion in the game Go, showing that this algorithm can be employed in surprising ways.

RFs

RFs, introduced by Breiman [2001], have become extremely popular as a machine learning algorithm. Much of this popularity stems from their quick speed and ease of use. Unlike the DNN and SVM, the RF classifier does not require any standardization or normalization of input features in the preprocessing stage. By taking the raw data of features and responses and specifying the number of trees in the forest, RF will return a model quickly and often outperforms even the most sophisticated algorithms.

Decision trees can easily overfit the data, a problem that many have tried to overcome by making the decision tree more robust. Limiting the depth of the tree and number of leaves in the terminal nodes are some methods that have been employed in an attempt to reduce the high variance problem. RFs take a very different approach to gaining robustness in resultant predictions. Given that decision trees can easily overfit the data, RFs attempt to reduce such overfitting along two dimensions. The first is bootstrapping with replacement of the row samples used in a given decision tree. The second is a subset of features that are randomly sampled without replacement at each node split, with the objective of maximizing the information gain for this subsample of features. Parent and child nodes are examined and features are chosen that provide for the lowest impurity of the child node. The more homogeneous the elements within the child split, the better the branch is at separating the data.

Many statisticians were irked by RFs when they were initially introduced because they only provided a limited number of features. At that time, the model-building intuition was to employ as much data as possible and to avoid limiting the feature space. By limiting the feature space, each tree has slightly different variations, and thus the average across the many trees, also known as *bagging* the trees within the forest, provides for a very robust prediction that easily incorporates the complexity in the data. RFs will continue to gain even more appeal with the added benefit of allowing researchers to see the features that are most important for a given RF prediction model. We will present a list of the most important feature per ETFs later in this work, and our results show the complexity of predicting across our ETF asset classes.

SVMs

SVMs, by Vapnik [1995], attempt to separate the data by finding supporting vectors that provide for the largest separation between groups, or maximize the margin. *Margin* is defined as the distance between the supporting hyperplanes.

One of the main advantages of this approach is that SVMs generate separations that are less influenced by outliers and potentially more robust vis-a-vis alternative classifiers. Additionally, SVMs allow for the option to apply the radial basis function, which allows for nonlinear separation by leveraging the kernel trick. The kernel trick casts the data into a higher dimension. In this higher dimension, linear separation occurs when projecting the data back down into the original dimensional space.

METHODOLOGY

As mentioned earlier, we have chosen widely used and liquid ETFs from various asset classes. The cross-section of ETFs allows us to include cross-asset correlation to boost predictive power. Presumably, investors make their decisions depending on their risk preferences as well as the ability to hold a well-diversified portfolio of assets. Although our list of ETFs is not exhaustive, it does represent well-known ETFs with which most practitioners and registered investment advisors should be well acquainted. The list of ETFs is as follows.

ETF Opportunity Set

- SPY—SPDR S&P 500;
  U.S. equities large cap
- IWM—iShares Russell 2000;
  U.S. equities small cap
- EEM—iShares MSCI Emerging Markets;
  Global emerging markets equities

- TLT—iShares 20+ Years;
  U.S. Treasury bonds
- LQD—iShares iBoxx $ Invst Grade Crp Bond;
  U.S. liquid investment-grade corporate bonds
- TIP—iShares TIPS Bond;
  U.S. Treasury inflation-protected securities
- IYR—iShares U.S. Real Estate
  Real estate
- GLD—SPDR Gold Shares;
  Gold
- OIH—VanEck Vectors Oil Services ETF;
  Oil
- FXE—CurrencyShares Euro ETF;
  Euro

We test the predictability of ETF returns on a set of varying horizons. Although it is common knowledge that stock prices and thus ETF prices follow a random walk on the shorter horizons, thus making shorter-term predictability very difficult, longer horizons may be driven by asset class linkages and attention. Asset classes ebb and flow in and out of investors' favor. With this intuition, we attempt to predict the direction of price moves, not the magnitude. Thus, we cast our research into a supervised classification problem. The returns are calculated by employing adjusted closing prices (adjusted for stock splits and dividends) for the given time periods as measured in trading days (1, 2, 3, 5, 10, 20, 40, 60, 120, and 250 days), using the following formula:

$$r_{t-n,t}^{ETF} = P_t^{ETF}/P_{t-n}^{ETF} - 1$$

For each horizon of n days and each ETF, we examine four dataset combinations as explanatory information sets. We employ the term *information set* as the set of features based on the following explicit definitions. Note that, for any given asset's change in price, we allow for its own information as well as the other ETFs' information to influence the sign of the price change over the given horizon. We define our four information sets A, B, C, and ABC as follows:

- Information set A: previous n days return and j lagged n days return, where j is equivalent to the previous horizon (i.e., for a 20-day horizon, the number of lagged returns will be 10) for all ETFs:

$$X_A = \{r_{t-n,t}, r_{t-n-1,t-1}, \ldots, r_{t-n-j,t-j}\}$$

- Information set B: average volume for n days and j lagged average volume for n days, where j is equivalent to the previous horizon for all ETFs:

$$X_B = \{v_{t-n+1,t}, v_{t-n,t-1}, \ldots, v_{t-n-j+1,t-j}\}$$

- Information set C: day of the week and month dummy variables.
- Information set ABC: A, B, and C combined.

We concentrate our presentation of results on information set ABC, but the other information sets provide insight on the drivers of the predictions across our three algorithms. A priori, we believe that past returns will be the most beneficial in terms of future return predictions, and volume will be useful to boost the results. Many have shown volume to capture the notion of investors' attention. Higher volumes are typically associated with more trading activity. If trading releases information, then those ETFs with a higher volume of trading should be adjusting more quickly to their true values. Note that we implicitly assume the dollar volume of trading is approximately equal across ETFs and concern our study with share volume. Clearly, the ETF prices are not equal at any given time. However, the intuition is clear: More relative trading volume is an important feature for predictability across ETFs. We would suspect that volume and prior returns should work in tandem. However, we find that volume in isolation works very well; that is, B works well even without A—a rather surprising result.

Dummy variables are assumed to boost the performance of algorithms on shorter time periods and to be insignificant on longer horizons. It is important to note that A and ABC datasets are equivalent in number of observations, whereas B and C are not equivalent to A and have one less observation. This occurs because we need $n + 1$ days of adjusted closing prices to compute returns. We only need n days, however, to compute average volume. We are employing the daily volume for that day.

The dependent variable is defined as 1 if n days' return is equal to or greater than 0 and 0 otherwise:

$$Y = \{1, \text{if } r_{t,t+n} \geq 0; 0 \textit{ otherwise}\}$$

Next, we employ cross-validation. We divide datasets and corresponding dependent variables into training and test sets. Division is done randomly in the following proportion: 70% training set and 30% test set. The training set will be used to train the model and the test set to estimate the predictive power of the algorithms. Following best-practice procedures, we use our training set and perform holdout cross-validation. A validation set is obtained from the training set. Once the *k*-fold cross-validation has been performed and the optimal hyperparameters have been selected, we employ this model only once on our test set. Many textbooks recommend 10-fold cross-validation in the training set; however, we used only threefold cross-validation, given that larger cross-validation tests would take an even longer time to generate results. We exhaustively searched for the best hyperparameters for each of our algorithms.[1] The possible values for hyperparameters for each algorithm are as follows.

Hyperparameter Search Space.[2]

- DNN
  - alpha (L2 regularization term)—{*0.0001, 0.001, 0.01, 0.1, 1.0, 10.0, 100.0, 1000.0*}
  - activation function—{*rectified unit linear function, logistic sigmoid function, hyperbolic tan function*}[3]
  - solver for weight optimization—*stochastic gradient descent*
  - hidden layers—{(*100, 100*), (*100, 100, 100*)}
- RF
  - number of decision trees—{*100, 200, 300*}
  - function to measure quality of a split—{*Gini impurity, information gain*}
- SVM
  - C (penalty parameter of the error term)—{*0.0001, 0.001, 0.01, 0.1, 1.0, 10.0, 100.0, 1000.0*}
  - kernel—{*linear, radial basis function*}

The best estimator is determined by the accuracy of all possible combinations of hyperparameters values listed. After the best estimator is found, we use test set data to see how the algorithm performs. Scoring for test performance is based on accuracy, where accuracy is defined by how well the predicted outcome of the model compares to the actual outcome. To estimate the performance of each algorithm, we introduce our gain criteria. These criteria show whether the explanatory dataset explains the dependent variable better than randomly generated noise.

Introduce Gain Criteria

The gain criteria are computed as the difference between the accuracy of the model given the input information set and the accuracy of the model given noise data. We define *noise* as creating random data from a uniform random distribution bounded by 0 and 1 and replacing the original input data with these simulated noise data in the modeling process. We replaced this noise directly into the input feature data space, which preserves the shape of the actual data. We compute the gains by rerunning the same code used previously to obtain accuracy scores for the original data and subtracting the results from the scores obtained by testing best estimators with the actual data. Formally,

$$Gain_n^{ETF} = Actual\ Data\ Test\ Score_n^{ETF} - Noise\ Test\ Score_n^{ETF}$$

Using this score, it is easy to compare the performance of each algorithm and choose the one with the highest explanatory power.

RESULTS

First, we compare the test scoring accuracy (Exhibit 1) and prediction gain criteria accuracy (Exhibit 2) of each algorithm on different horizons for the ABC dataset. In Exhibit 1, test score accuracy increases as the prediction horizon increases. We would expect such a pattern given that some of our ETFs have had a natural positive drift over the sample period examined. The gain criteria would adjust for such positive drifts. Notice that the gain criterion ranges from 0% to 35%, compared to the scoring accuracy, which ranges from 0% to 100%.

RF and SVM show close results for all horizons in terms of test set accuracy and gain criterion, whereas DNN converges to the other algorithms at 40 days or more. This may be due to the insufficient number of hidden layers or default values of other hyperparameters of the DNN classifier. Overall, we see that, even though test set accuracy on average increases with horizon length, the gain criterion peaks at 40 days and steadily falls thereafter. This finding is because for some ETFs,

Exhibit 1

Overall Algorithm Performance

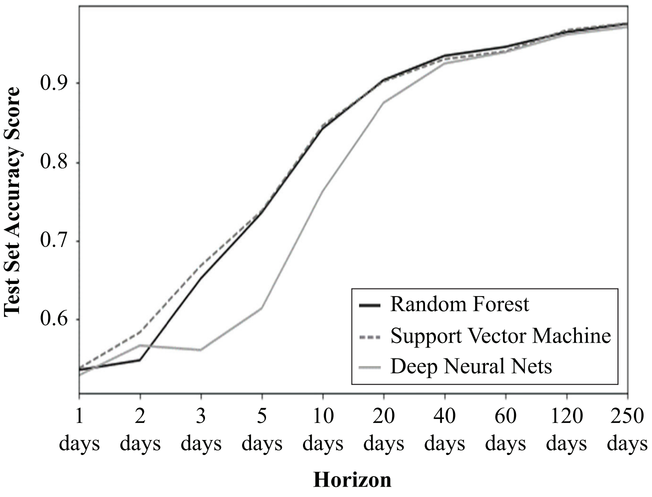


Exhibit 2

Overall Algorithm Gain

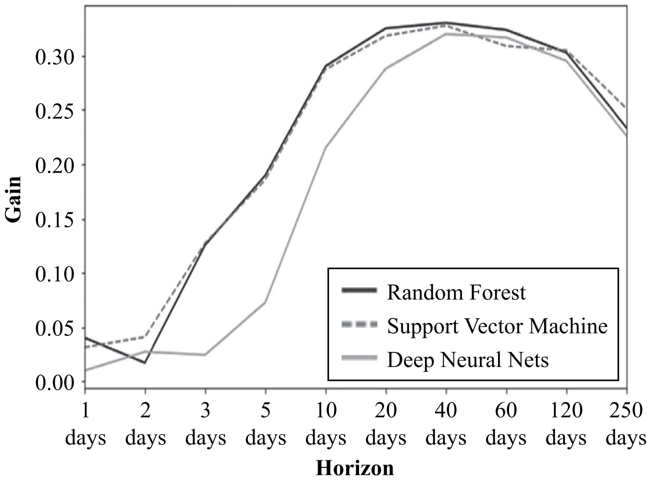
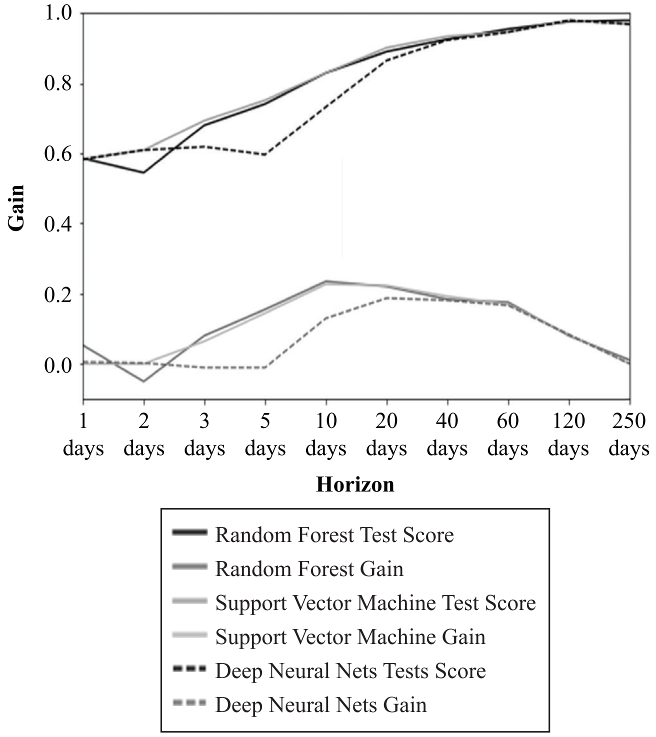


Exhibit 3

SPY Algorithm Gain



such as SPY, IWM, IYR, and GLD, we see a strong trend toward increases or decreases in price changes at longer horizons (see the chart for deep neural nets TLT in Exhibit A6 in the Appendix), which increases the test set accuracy score but lowers the gain criterion score.

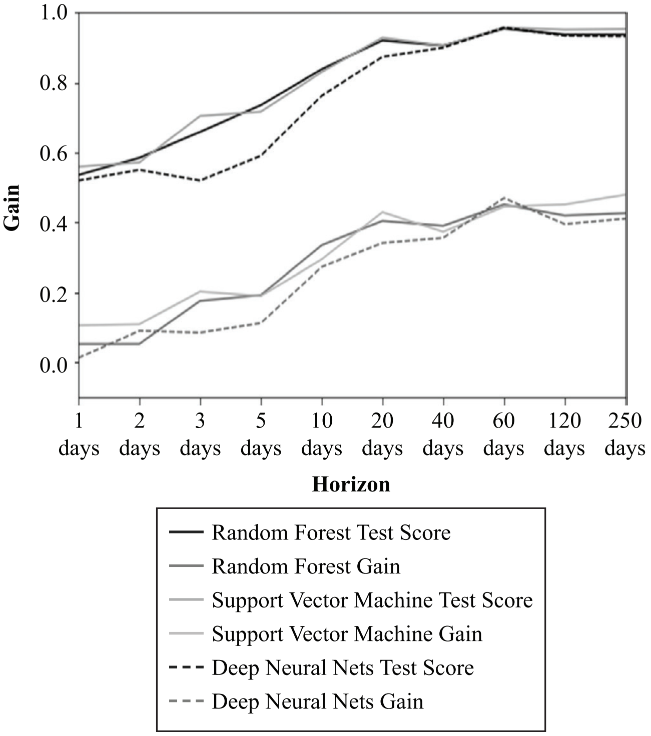
Not surprisingly, the predictive power of algorithms increases with the forecast horizon. The result for the gain criterion was anticipated because, from preprocessing the data, examination suggested that noise level will increase with horizon, thus decreasing the gain. However, gain in predictive power for 120 and 250 days is still high, which raises the question of why such long-term predictions using only technical analysis still have good results. To understand this phenomenon, more analysis is required with the introduction of fundamental data, which is outside the scope of this article.

The next step in our analysis is to see how the algorithms performed in more detail. Using the gain criterion, we compare the performance of algorithms for each ETF from our list (see the Appendix). Generally, we see two patterns in gain behavior. The first is a peak at the 10- to 40-days level and a fall at consequent horizons, and the second is an increase for up to 20 to 60 days and a plateau with fluctuations or a slight rising trend to 250 days. We believe the reasons for such behaviors are the same as previously described.

The examples of SPY and EEM, respectively, shown in Exhibits 3 and 4.

Because overall DNN performance was lower than that of the other two algorithms on 3- to 20-day horizons, it is not surprising that it shows the same pattern at the single-ETF level. Nonetheless, for some ETFs (i.e., IWM and TLT), the DNN algorithm was able to

Exhibit 4
EEM Algorithm Gain

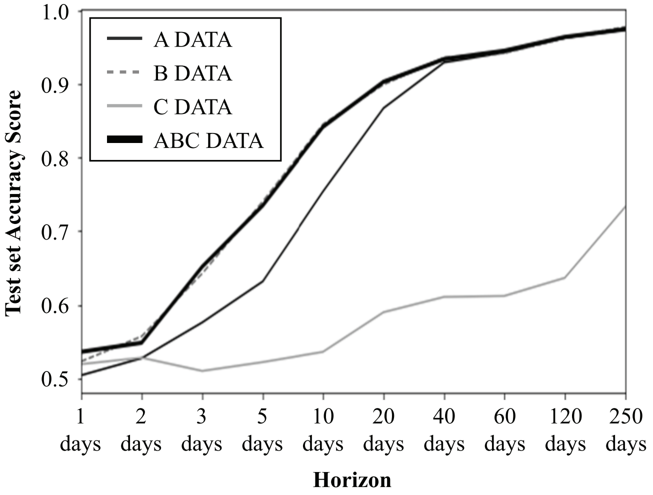


catch up to the rest at 20 days. SVM and RF show close results for all of the ETFs, but the latter seems to produce less volatile results with respect to horizon. We find horizons of 10 to 60 days to be most interesting across all ETFs. Although for some instances 3-, 5-, 120-, and 250-day periods also draw attention and deserve more rigorous analysis, our goal is to compare the algorithms' performances and to estimate the possibility and feasibility of meaningful predictions rather than to investigate the specifics of prediction of individual ETFs.

Next, we develop a deeper understanding of the explanatory variables and their significance in terms of predictive power. For that purpose, we examine the average test scores and gain criterion for the A, B, and C datasets for each algorithm and compare them to the ABC results.

Let's start with RF, displayed in Exhibits 5 and 6. Volume (dataset B) effectively explains all the results obtained in previous sections, which is a surprising and unexpected result. Returns (dataset A) have decent predictive power. However, strong at the horizon of 40 days and longer, returns show the same performance as volume and combined datasets. Calendar dummies (dataset C), however, seem to explain a small portion of daily returns and monthly (20 days) returns. We assume this is because dummies are for day of the week and month. Nonetheless, the predictive power of this set is negligible, and a clear contribution can only be seen at a one-day horizon.

Exhibit 5
RF Dataset Test Scores



SVM results (Exhibits 7 and 8) have the same pattern as RF. However, the overall performance and gain for the returns dataset is closer to those of the combined dataset in comparison with RF. What is more interesting, the combined dataset seems to outperform individual datasets on one- to three-day horizons. Furthermore, calendar dummy variables seem to yield better results but are still not large enough to be significant, and they do not add any predictive power to the combined dataset.

As mentioned earlier, DNNs (Exhibits 9 and 10) struggle to show competitive results on horizons less than 20 to 40 days. One- to five-day horizon predictions have effectively no predictive power. Apart from other algorithms, DNN benefits from a combination of datasets. However, the gradation of datasets with respect to gain is the same as for the previous algorithms, as are patterns of change in gains and test scores with varying horizons.

The results of DNN on 10- to 60-day horizons suggest that there is a possibility of improvement in algorithm predictions with combinations of datasets, which is not the case for other algorithms. Overall, we see poor ability to predict short-term returns for all

Exhibit 6

RF Dataset Gain

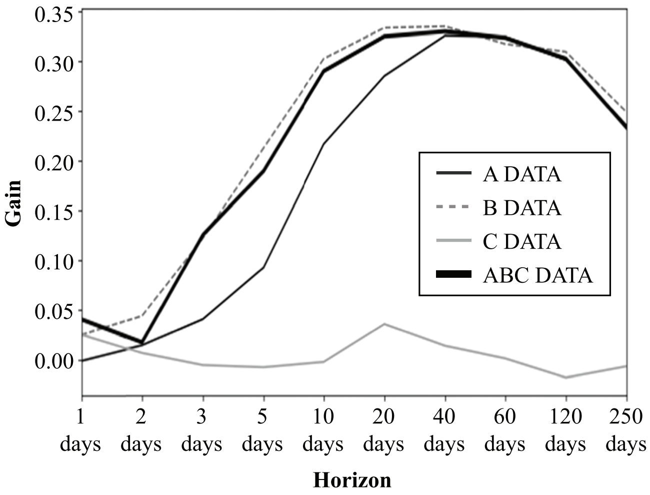


Exhibit 7

SVM Test Set Scores

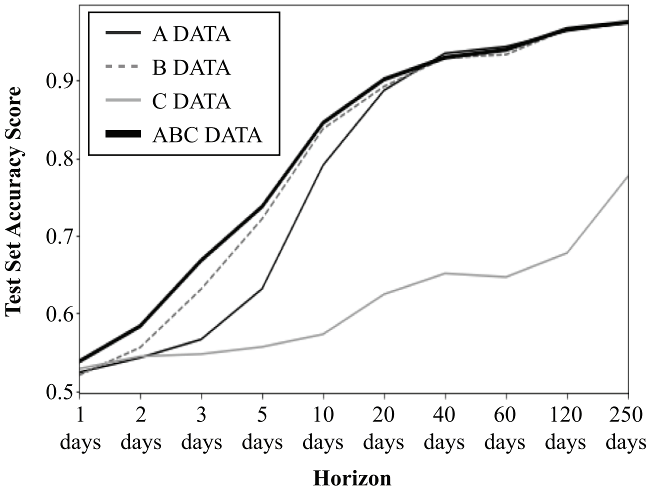


Exhibit 8

SVM Dataset Gain

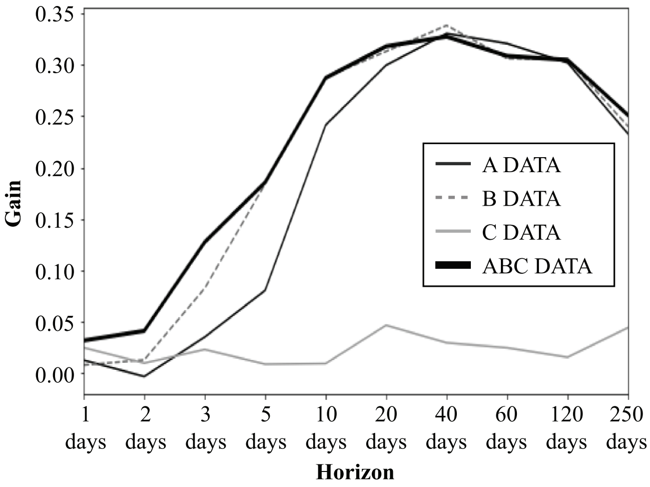
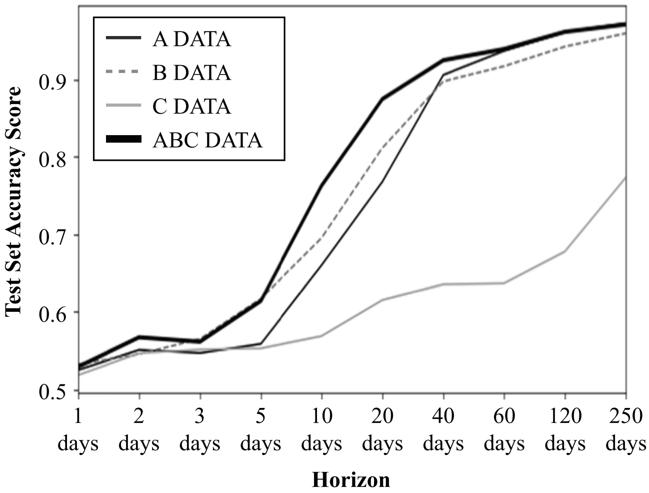


Exhibit 9

DNN Dataset Test Scores



algorithms. The solution to boost results might be an ensemble of algorithms, but such an analysis is beyond the scope of this article.

As mentioned earlier, we find horizons from 10 through 60 days to be most interesting in terms of predictive power. Thus, we will examine the performance of the algorithms in these time periods in more detail using the receiver operator characteristic (ROC), which will allow us to compare algorithms from a different angle. Based on ROC, we can compute another measure for algorithm comparison, the ROC area under the curve (AUC). We generated ROC curves for horizons of 10 to 60 days (see the Appendix). The results follow the same pattern as in all previous sections. For example, see the ROC graphs for EEM (Exhibits 11, 12, and 13). We also calculated the AUC for each of the selected horizons, ETFs, and algorithms (Exhibits A2 through A4 in the Appendix).

Longer-horizon ROC curves have an almost ideal form and AUCs close to 1, which suggest high predictive ability with high accuracy. Altogether, we can conclude that predictions for these ETFs are possible.

Exhibit 10

DNN Dataset Gain

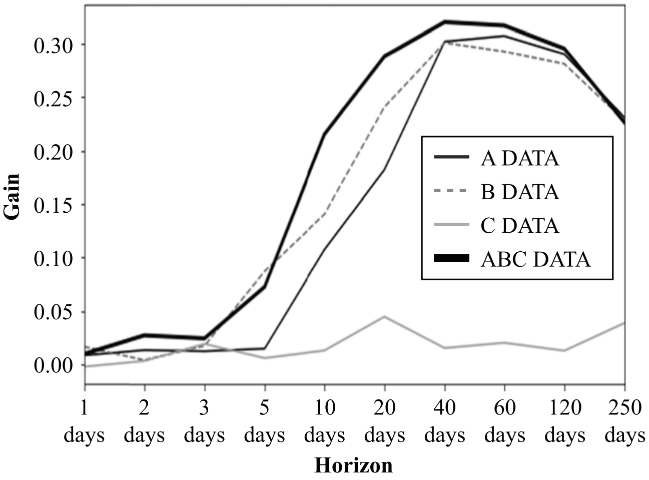


Exhibit 11

RF EEM

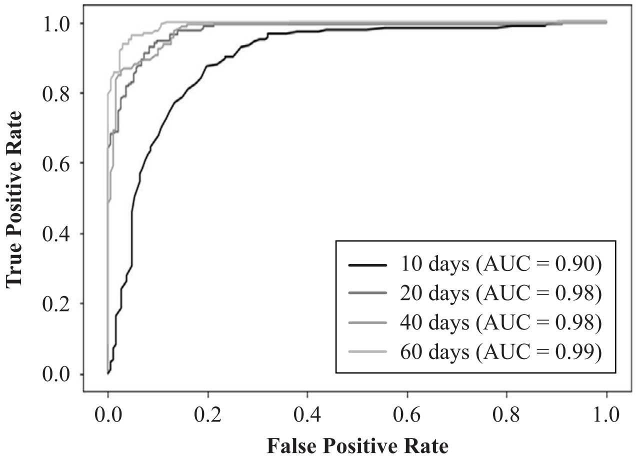


Exhibit 12

SVM EEM

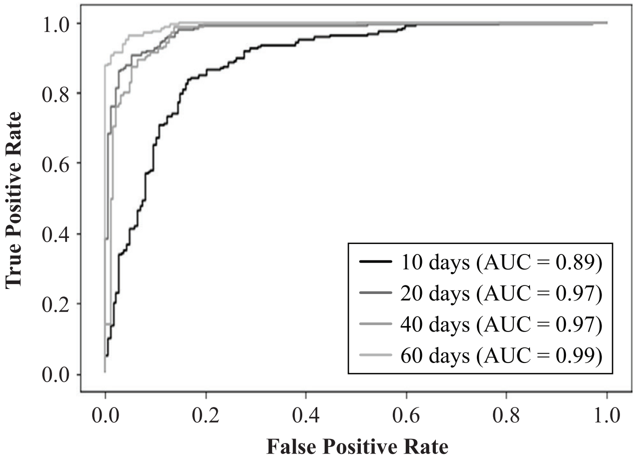
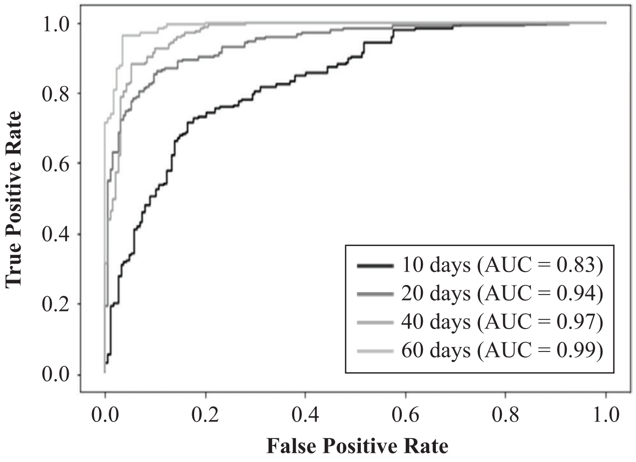


Exhibit 13

DNNs EEM

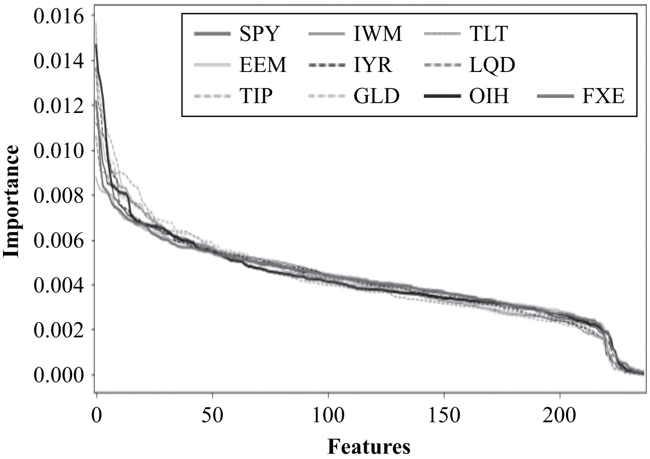


Feature Importance with RF

We also try to shed some light on which data drive the performance of algorithms by assessing feature importance with RFs. As previously discussed, volume is a good predictor by itself but is more powerful in combination with returns. However, it is unclear which features are actually driving the performance of the algorithms. In the case of SVM, it is only possible to interpret weights of each feature if the kernel is linear. For DNN, it is hard to explain and grasp what relationships exist within the hidden layers. For RF, however, we can compute and interpret the importance of each feature in an easy way.

We decided to examine the importance of 20-day horizon RF features for ETFs (Exhibits 14 and 15). The results show that there is no single feature that would explain most of the returns. Note that on the graph, features are sorted in descending order for each ETF. As one can see, the pattern is the same for all ETFs, meaning that almost all information in the dataset is contributing and is useful for prediction. With the features' importance structured this way, we see that there is not a single factor that contributes more than 1.6%. However, the dataset also contains lagged variables. The questions that immediately arise are whether a group or groups of features (i.e., volume of SPY and returns on EEM) are more beneficial, and whether we can drop them.

Exhibit 14
20 Days Feature Importance



For that purpose, we grouped returns and volumes for each ETF and summed up importance within each group (Exhibit 16). Volume is more important than returns, which confirms the difference in results for the A and B datasets. One of the reasons for such behavior might be a relationship between volume and returns; we assume that might be the result of a relationship obtained by Chen, Hong, and Stein [2001] and Chordia and Swaminathan [2000] in the sense that past volume is a good predictor of future returns' skewness and patterns. Calendar dummies show little to no influence on predictions, as expected from dataset results.

CONCLUSIONS

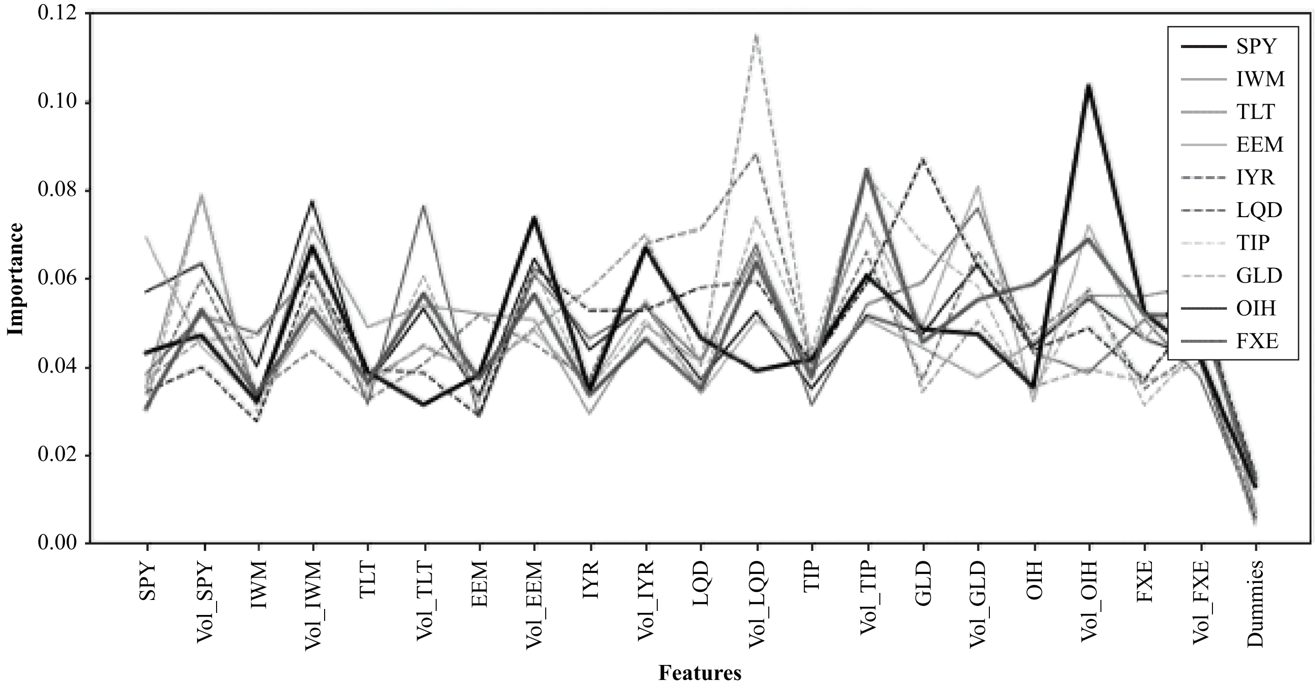
In this work, we examined the ability of three popular machine learning algorithms to predict ETF returns. Although we restricted our initial analysis to only the direction of the future price movements, we still

Exhibit 15
10 Features with Highest Importance by ETF

	Equity			Fixed Income			Commodities and Real Estate			
	SPY	IWM	EEM	TLT	TLP	LQD	IYR	GLD	OIH	FXE
1	Lag10_Vol_ GLD	Vol_TLT	Lag1_Vol_ OIH	Lag10_Vol_ SPY	Lag6_Vol_ LQD	Lag10_Vol_ TIP	Lag3_GLD	Lag10_Vol_ SPY	Vol_OIH	Lag1_Vol_ TIP
2	Lag4_Vol_ IWM	Lag10_Vol_ GLD	Lag1_SPY	Lag9_Vol_ SPY	Lag7_Vol_ LQD	Lag10_Vol_ GLD	GLD	Lag1_Vol_ TIP	Lag2_Vol_ OIH	Vol_TIP
3	Lag9_Vol_ EEM	Lag10_Vol_ EEMSPY	SPY	Lag7_Vol_ SPY	Lag5_Vol_ LQD	Lag4_Vol_ LQD	Lag2_GLD	Lag3_Vol_ LQD	Lag3_Vol_ OIH	Lag1_OIH
4	SPY	Lag9_Vol_ EEM	Vol_OIH	Lag8_Vol_ SPY	Lag10_Vol_ LQD	Lag7_Vol_ LQD	Lag9_Vol_ IWM	Lag4_Vol_ TIP	Lag1_Vol_ OIH	Lag2_Vol_ TIP
5	Lag3_Vol_ IWM	Lag1_Vol_ TLT	Lag9_Vol_ GLD	Lag6_Vol_ IWM	Lag8_Vol_ LQD	Lag5_Vol_ LQD	Lag1_GLD	Lag2_Vol_ TIP	Lag10_FXE	Lag5_Vol_ OIH
6	Lag10_Vol_ OIH	Lag8_Vol_ LQD	Lag7_Vol_ GLD	Lag4_Vol_ SPY	Lag9_Vol_ LQD	Lag7_LQD	Lag10_Vol_ IYR	Vol_TIP	Lag5_Vol_ OIH	Lag5_Vol_ TIP
7	Lag10_Vol_ EEM	Lag1_Vol_ IWM	Lag5_Vol_ TIP	Lag1_Vol_ FXE	Lag7_Vol_ IYR	Vol_LQD	Lag7_LQD	Lag8_Vol_ SPY	Lag9_Vol_ EEM	Lag1_Vol_ LQD
8	Vol_SPY	Lag8_Vol_ GLD	Lag10_Vol_ GLD	Lag6_Vol_ SPY	Lag3_Vol_ TIP	Lag8_Vol_ LQD	Lag9_Vol_ LQD	GLD	Lag4_Vol_ OIH	Vol_LQD
9	Lag6_Vol_ EEM	Lag9_Vol_ TIP	Lag6_Vol_ TIP	Lag5_Vol_ SPY	Lag3_Vol_ LQD	Lag4_LQD	Lag10_Vol_ LQD	Lag10_GLD	Lag6_Vol_ OIH	Lag8_Vol_ GLD
10	Vol_TLT	Lag9_Vol_ GLD	Lag8_Vol_ GLD	Lag5_Vol_ IWM	Vol_LQD	Lag1_Vol_ LQD	Lag10_Vol_ GLD	Lag5_Vol_ OIH	Vol_EEM	Lag2_OIH

EXHIBIT 16

20 Days Group Feature Importance



procured valuable results. First, machine learning algorithms do a good job of predicting price changes at the 10- to 60-day horizon. Not surprisingly, these algorithms fail to predict returns on short-term horizons of five days or less. We introduce our gain measure to help assess efficacy across algorithms and horizons. We also segmented our input feature variables into different information sets so as to cast our research in the framework of the efficient markets hypothesis. We find that the volume information set (B) works extremely well across our three algorithms. Moreover, we find that the most important predictive features vary depending on the ETFs that are being predicted. Financial intuition helps us to understand the prediction variables with complex relationships embedded within the prediction of the S&P 500, as proxied by SPY, requiring a more diverse set of features compared to the complexity of the top feature set needed to explain GLD or OIH.

In practice, the information set could be vastly extended to include other important features, such as social media, along the lines of Liew and Budavari [2017], who identified the social media factor. Additionally, the forecasting time horizons could have been extended even further beyond one trading year or shortened to examine intraday performance. However, we leave this more ambitious research agenda to future work.

One interesting application is to use several different horizon models launched at staggered times within a day, thereby gaining slight diversification benefits for the resultant portfolio of strategies.

In sum, we hope that our application of machine learning algorithm motivates others to move this body of knowledge forward. These algorithms possess great potential in their applications to the many problems in finance.

Appendix

Exhibit A1

Proportion of Upward Price Movements in Test Set

	1 Day	2 Days	3 Days	5 Days	10 Days	20 Days	40 Days	60 Days	120 Days	250 Days
SPY	0.584	0.609	0.629	0.607	0.603	0.678	0.741	0.779	0.896	0.969
IWM	0.562	0.612	0.581	0.575	0.568	0.617	0.670	0.688	0.764	0.932
TLT	0.504	0.559	0.536	0.532	0.557	0.579	0.589	0.618	0.573	0.661
EEM	0.507	0.516	0.544	0.529	0.495	0.468	0.457	0.488	0.479	0.490
IYR	0.520	0.593	0.624	0.634	0.578	0.617	0.557	0.618	0.747	0.917
LQD	0.541	0.604	0.568	0.586	0.624	0.631	0.693	0.733	0.684	0.745
TIP	0.469	0.559	0.525	0.516	0.551	0.579	0.598	0.585	0.524	0.349
GLD	0.493	0.524	0.512	0.505	0.514	0.424	0.382	0.370	0.205	0.047
OIH	0.496	0.532	0.525	0.503	0.514	0.474	0.529	0.533	0.556	0.484
FXE	0.488	0.484	0.501	0.503	0.481	0.457	0.437	0.406	0.403	0.411
Mean	**0.516**	**0.559**	**0.555**	**0.549**	**0.548**	**0.552**	**0.565**	**0.582**	**0.583**	**0.601**

Notes: For each horizon and ETF, we compute the proportion of upward movements in price in our test set. One can see strong upward trend with increasing horizon for SPY, IWM, and IYR and a downward trend for GLD.

Exhibit A2

RF ROC AUC

	1 Days	2 Days	3 Days	5 Days	10 Days	20 Days	40 Days	60 Days	120 Days	250 Days
SPY	0.579	0.563	0.704	0.791	0.878	0.951	0.967	0.980	0.999	0.996
IWM	0.516	0.553	0.727	0.804	0.919	0.953	0.967	0.977	0.998	0.986
TLT	0.516	0.514	0.682	0.804	0.906	0.974	0.986	0.992	0.992	0.999
EEM	0.506	0.618	0.745	0.801	0.896	0.975	0.980	0.993	0.991	0.990
IYR	0.518	0.582	0.697	0.819	0.934	0.951	0.992	0.986	1.000	0.981
LQD	0.539	0.504	0.672	0.843	0.935	0.981	0.980	0.992	0.995	0.993
TIP	0.496	0.563	0.721	0.807	0.923	0.962	0.983	0.980	0.991	0.999
GLD	0.536	0.586	0.708	0.792	0.934	0.945	0.982	0.995	0.974	0.997
OIH	0.482	0.560	0.691	0.793	0.907	0.972	0.996	0.987	0.988	1.000
FXE	0.571	0.553	0.662	0.804	0.928	0.962	0.980	0.994	0.996	1.000

Exhibit A3

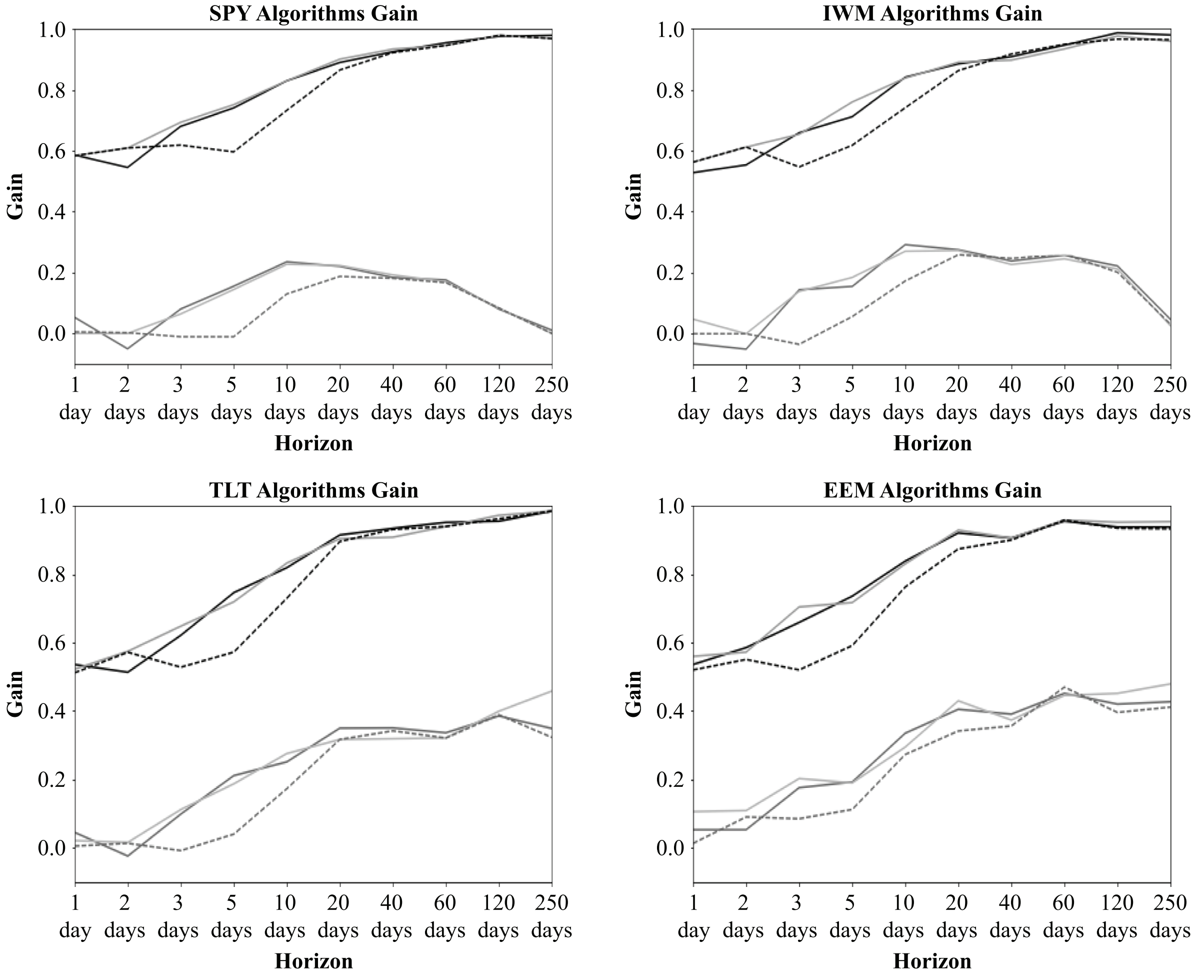
SVM ROC AUC

	1 Days	2 Days	3 Days	5 Days	10 Days	20 Days	40 Days	60 Days	120 Days	250 Days
SPY	0.521	0.440	0.755	0.817	0.895	0.959	0.966	0.986	0.997	0.987
IWM	0.542	0.548	0.722	0.825	0.918	0.944	0.957	0.982	0.998	0.953
TLT	0.547	0.587	0.726	0.794	0.907	0.977	0.970	0.975	0.988	0.999
EEM	0.558	0.600	0.775	0.786	0.889	0.974	0.970	0.993	0.985	0.990
IYR	0.478	0.618	0.702	0.816	0.927	0.953	0.993	0.987	0.999	0.967
LQD	0.527	0.497	0.668	0.819	0.937	0.971	0.974	0.993	0.998	0.978
TIP	0.532	0.600	0.709	0.820	0.933	0.955	0.971	0.978	0.983	1
GLD	0.542	0.621	0.752	0.822	0.927	0.936	0.986	0.992	0.977	0.987
OIH	0.506	0.578	0.737	0.801	0.903	0.966	0.998	0.973	0.988	1
FXE	0.515	0.619	0.695	0.827	0.921	0.959	0.971	0.991	0.991	0.996

EXHIBIT A4
DNNs ROC AUC

	1 Days	2 Days	3 Days	5 Days	10 Days	20 Days	40 Days	60 Days	120 Days	250 Days
SPY	0.469	0.524	0.509	0.615	0.782	0.922	0.965	0.982	0.994	0.996
IWM	0.462	0.524	0.527	0.649	0.812	0.927	0.960	0.979	0.995	0.987
TLT	0.545	0.589	0.547	0.602	0.803	0.959	0.981	0.988	0.990	0.999
EEM	0.563	0.585	0.573	0.611	0.835	0.940	0.969	0.989	0.987	0.985
IYR	0.441	0.506	0.588	0.716	0.843	0.937	0.980	0.984	0.999	0.974
LQD	0.510	0.493	0.538	0.642	0.875	0.956	0.968	0.992	0.996	0.981
TIP	0.565	0.532	0.520	0.658	0.839	0.945	0.975	0.978	0.988	0.999
GLD	0.450	0.611	0.619	0.646	0.856	0.929	0.978	0.991	0.972	1
OIH	0.496	0.523	0.544	0.645	0.844	0.932	0.992	0.980	0.990	1
FXE	0.530	0.560	0.582	0.658	0.843	0.925	0.967	0.992	0.993	0.996

EXHIBIT A5
ETF Algorithms Gain



(continued)

Exhibit A5 *(continued)*

ETF Algorithms Gain

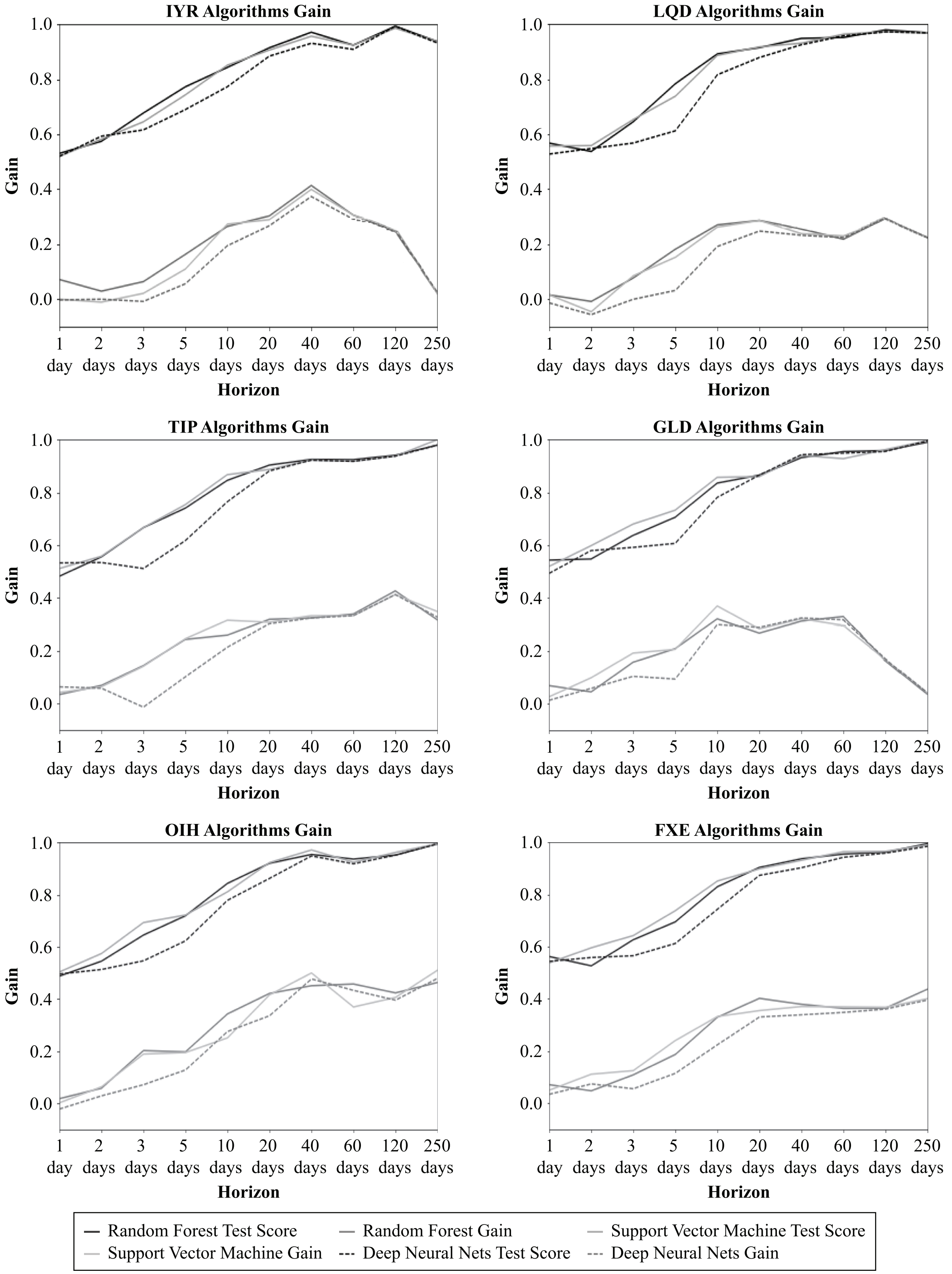
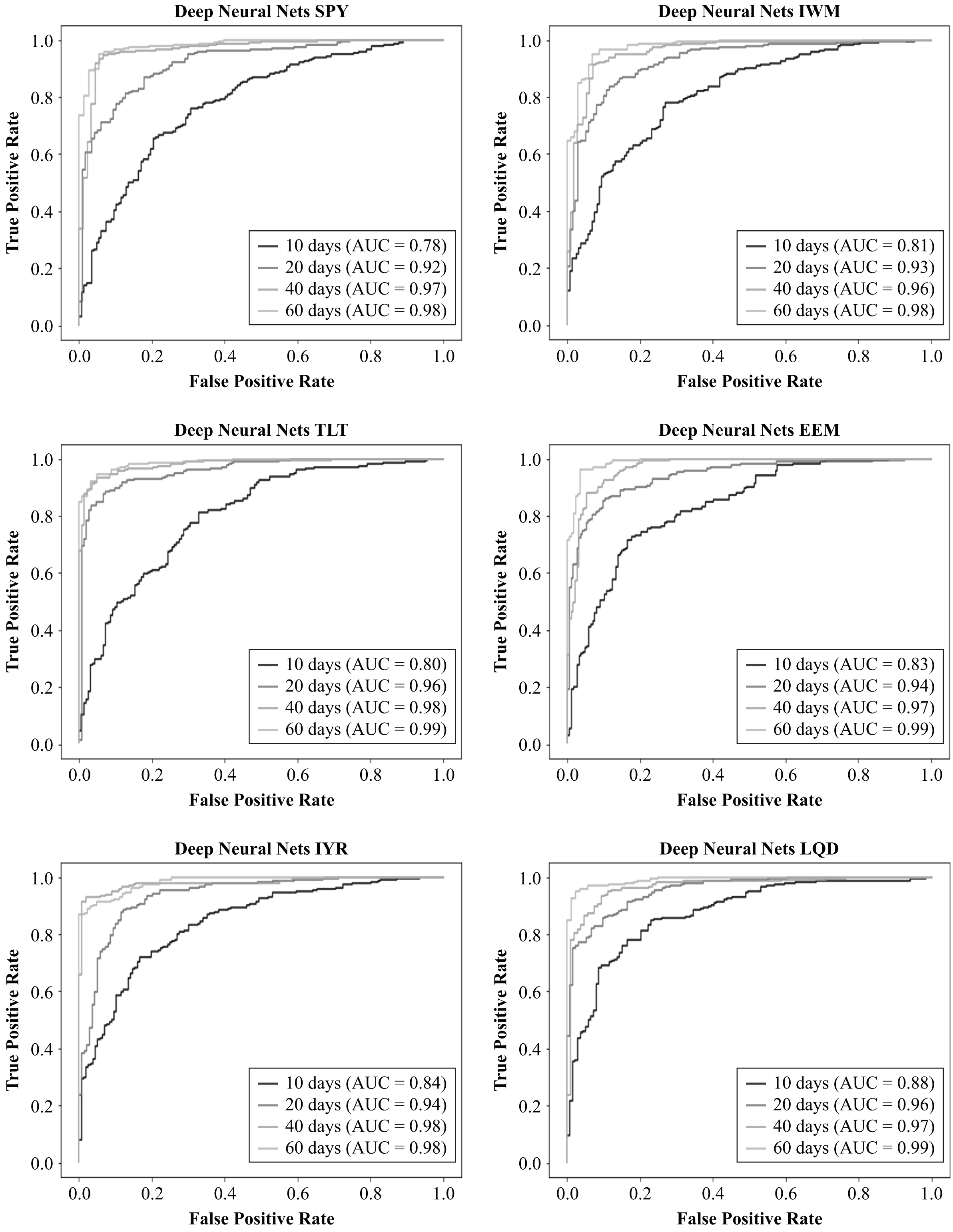


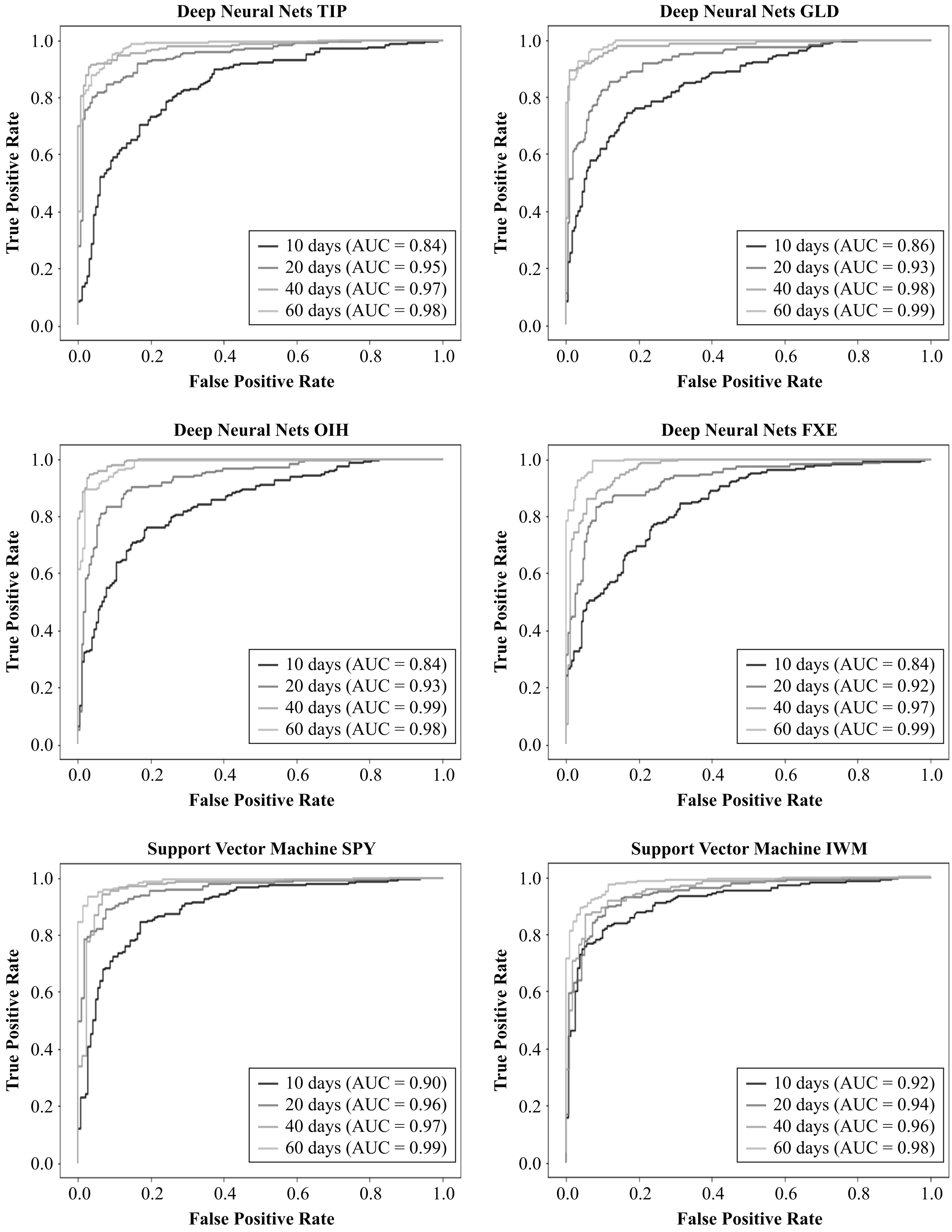
Exhibit A6
ETF ROC



(continued)

EXHIBIT A6 *(continued)*

ETF ROC



Deep Neural Nets TIP

Deep Neural Nets GLD

Deep Neural Nets OIH

Deep Neural Nets FXE

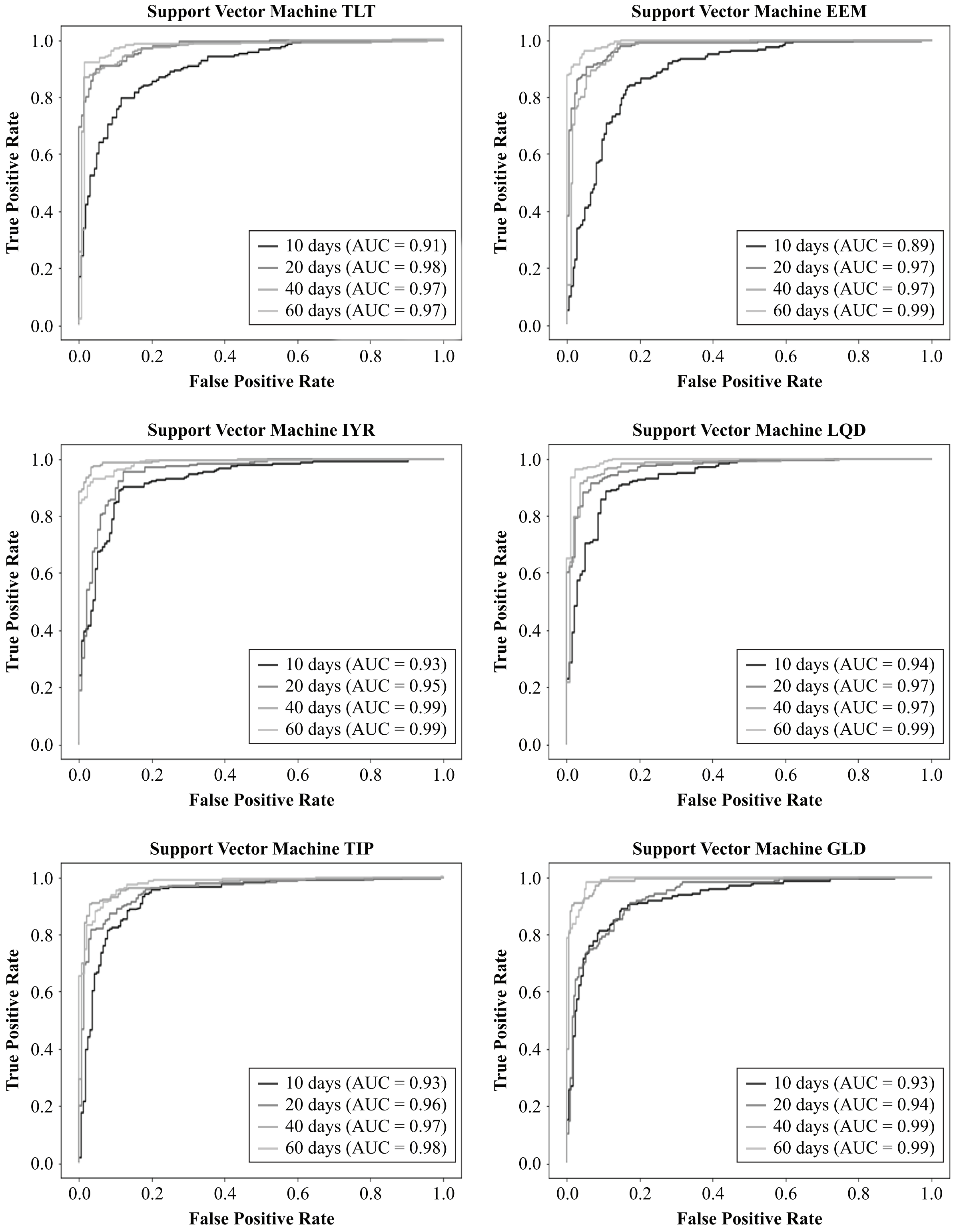
Support Vector Machine SPY

Support Vector Machine IWM

(continued)

Exhibit A6 *(continued)*

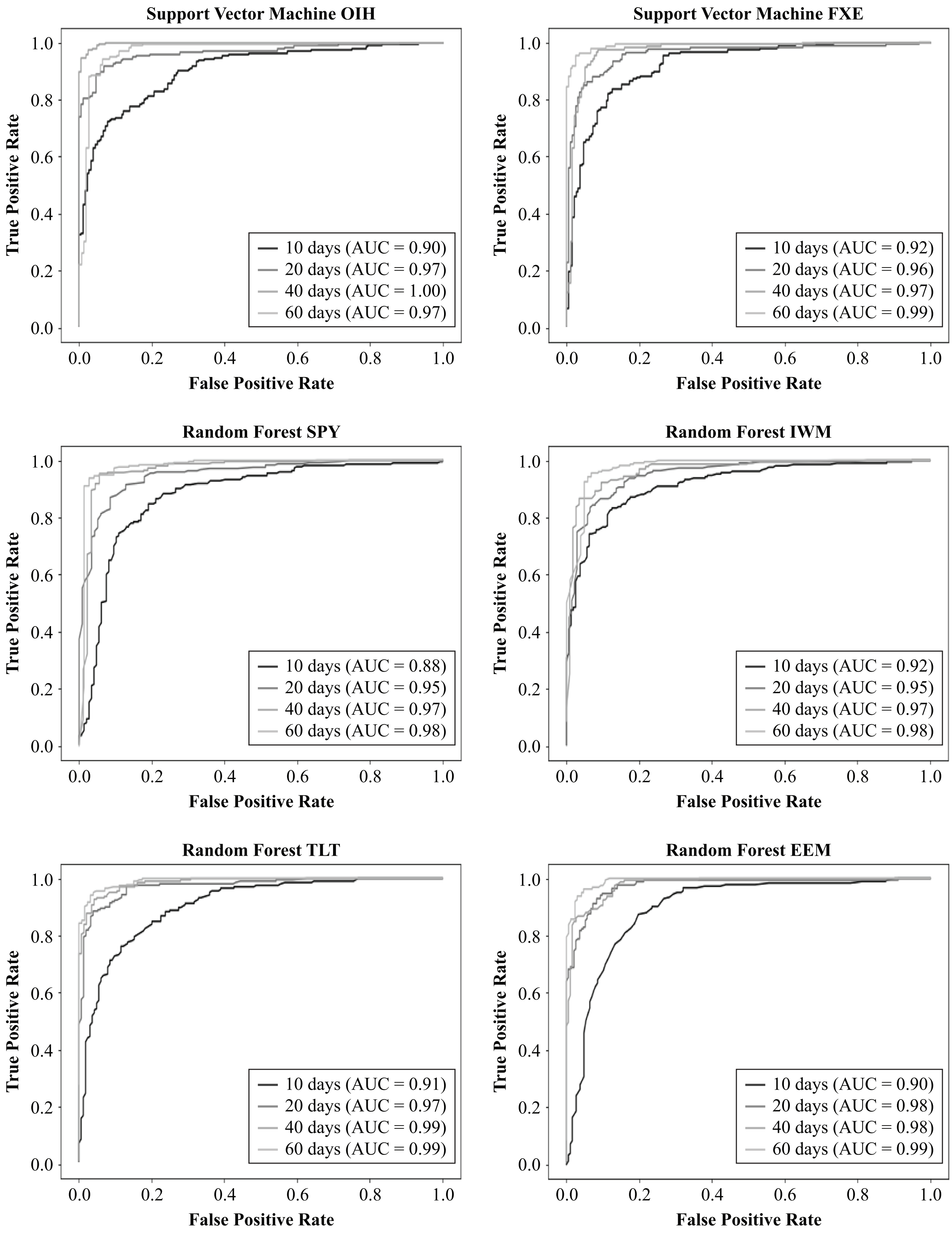
ETF ROC



(continued)

Exhibit A6 *(continued)*

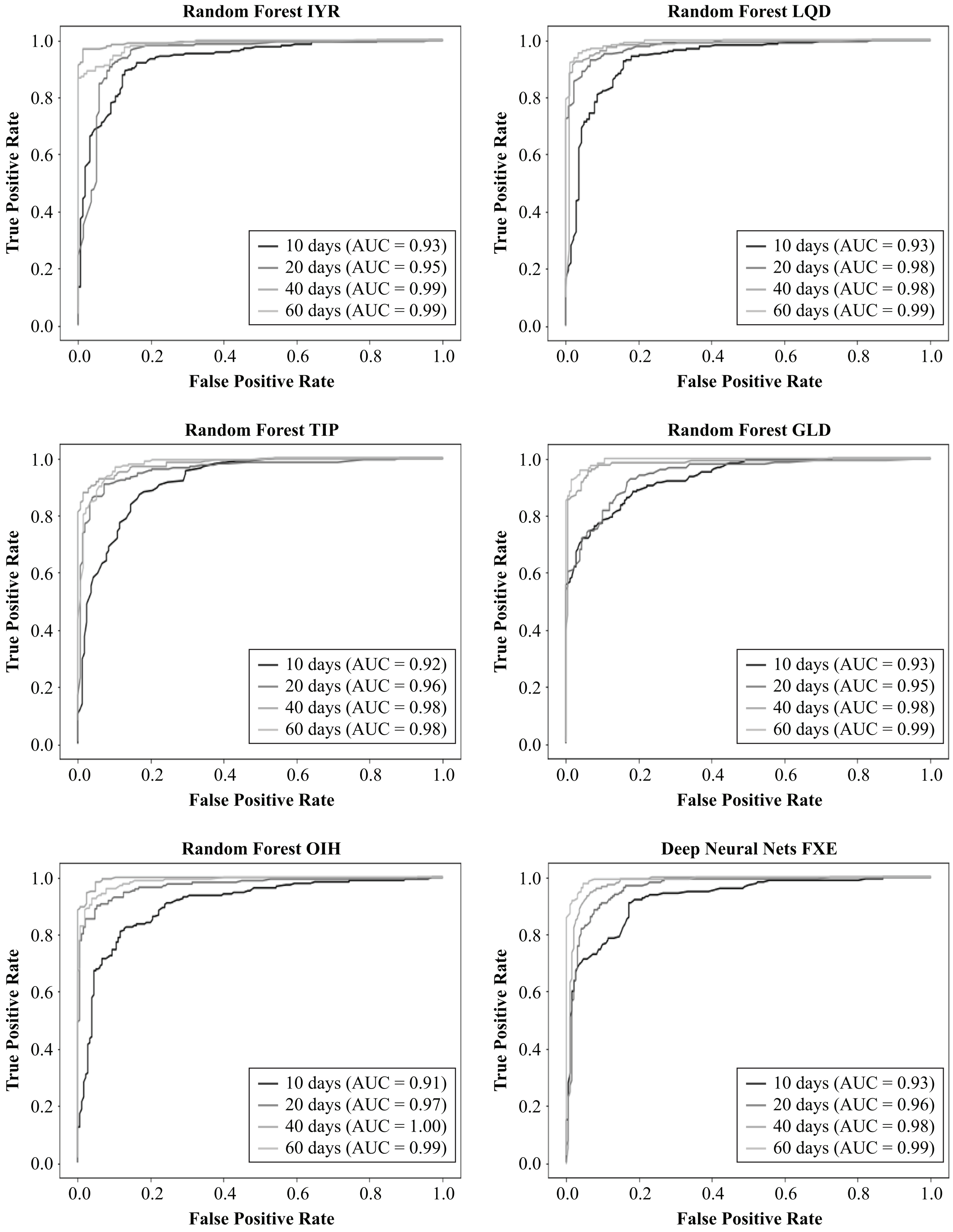
ETF ROC



(continued)

EXHIBIT A6 *(continued)*

ETF ROC



Random Forest IYR

Random Forest LQD

Random Forest TIP

Random Forest GLD

Random Forest OIH

Deep Neural Nets FXE

ENDNOTES

[1] For the DNN and SVM approaches, we also standardize the data using the training set prior to training and testing estimators. Standardization computes the ratio of the demeaned feature divided by the standard deviation of that feature. Thus, in the training set, each feature has a mean of zero and a standard deviation of one; however, in the test set, the features' mean and standard deviation varies from zero and one.

[2] All other parameters have default values. For more information, see http://scikit-learn.org/.

[3] $f(x) = \max(0, x)$, $f(x) = 1/(1 + \exp(-x))$ and $f(x) = \tanh(x)$, respectively.

REFERENCES

Asness, C., T. Moskowitz, and L. Pedersen. "Value and Momentum Everywhere." *The Journal of Finance*, Vol. 68, No. 3 (2013), pp. 929-985.

Breiman, L. "Random Forest." *Machine Learning*, Vol. 45, No. 1 (2001), pp. 5-32.

Chen, J., H. Hong, and J.C. Stein. "Forecasting Crashes: Trading Volume, Past Returns, and Conditional Skewness in Stock Prices." *Journal of Financial Economics*, Vol. 61, No. 3 (2001), pp. 345-381.

Chordia, T., and B. Swaminathan. "Trading Volume and Cross-Autocorrelations in Stock Returns." *The Journal of Finance*, Vol. 55, No. 2 (2000), pp. 913-935.

Fama, E., and K. French. "The Cross-Section of Expected Stock Returns." *The Journal of Finance*, 47 (1992), pp. 427-465.

——. "Common Risk Factors in the Returns on Stocks and Bonds." *The Journal of Financial Economics*, 33 (1993), pp. 3-56.

——. "Size and Book-to-Market Factors in Earnings and Returns." *The Journal of Finance*, 50 (1995), pp. 131-155.

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

Keim, D. "Size-Related Anomalies and Stock Return Seasonality: Further Empirical Evidence." *Journal of Financial Economics*, Vol. 12, No. 1 (1983), pp. 13-32.

Lakonishok, J., A. Shleifer, and R. Vishny. "Contrarian Investment, Extrapolation, and Risk." *The Journal of Finance*, Vol. 49, No. 5 (1994), pp. 1541-1578.

Liew, J., and T. Budavari. "The 'Sixth' Factor—Social Media Factor Derived Directly from Tweet Sentiments." *The Journal of Portfolio Management*, Vol. 43, No. 3 (2017), pp. 102-111.

Liew, J., and V. Vassalou. "Can Book-to-Market, Size, and Momentum Be Risk Factors that Predict Economic Growth?" *Journal of Financial Economics*, 57 (2000), pp. 221-245.

McCulloch, W.S., and W. Pitts. "A Logical Calculus of the Ideas Immanent in Nervous Activity." *The Bulletin of Mathematical Biophysics*, Vol. 5, No. 4 (1943), pp. 115-133.

Price, R. "Stephen Hawking: Automation and AI Is Going to Decimate Middle Class Jobs." *Business Insider*—Tech Insider, December 2, 2016. http://www.businessinsider.com/stephen-hawking-ai-automation-middle-class-jobs-most-dangerous-moment-humanity-2016-12.

Raschka, S. *Python Machine Learning*. Birmingham, UK: PACKT Publishing, 2015.

Roll, R. "A Critique of the Asset Pricing Theory's Tests." *Journal of Financial Economics*, 4 (1976), pp. 129-176.

Rosenblatt, F. "The Perceptron, a Perceiving and Recognizing Automaton." Cornell Aeronautical Laboratory, 1957.

Rouwenhorst, K. "International Momentum Strategies." *The Journal of Finance*, 53 (1998), pp. 267-284.

Shleifer, A., and R. Vishny. "The Limits of Arbitrage." *The Journal of Finance*, Vol. 52, No. 1 (1997), pp. 35-55.

Vapnik, V. *The Nature of Statistical Learning Theory*. New York: Springer-Verlag, 1995.

Widrow, B., and T. Hoff. "Adaptive 'Adaline' Neuron Using Chemical 'Memistors'." Number technical report 1553-2, Stanford Electron Labs. Stanford, CA, October 1960.

To order reprints of this article, please contact David Rowe at drowe@iijournals.com or 212-224-3045.