AFM 423 -Topics in Financial Econometrics

GPR #2

Optimizing Momentum Through Ensemble Machine Learning

By: Mikael Gascho

Winter 2025

Abstract

The emergence of disparate explanations for the cause of extreme crashes within momentum-based portfolios raises the question of how to best adjust for momentum risk within an ensemble machine learning environment. Traditional Fama-French factors, along with competing risk-adjusted momentum factors were aggregated, cleaned, and transformed to enable effective machine learning. An ensemble model approach was utilized, combining an elastic net, decision tree, random forest, boosted decision tree, and neural network. These models were trained on Fama French factors in addition to the differing momentum factors, to predict future stock returns, and create a resultant portfolio. Finally, these models are compared through examining portfolio return distributions, and their performance during the Covid market crash. Risk-adjusted momentum factors increase the portfolio Sharpe Ratio while mitigating crash risk, with the strongest factor being WML* which adjusts for the time-varying nature of momentum risk.

Introduction

Ever since Jegadeesh and Titman (1993) first introduced the strategy of buying historical winners and selling losers, momentum has become one of the most studied factors within finance. This financial anomaly was demonstrated to occur consistently throughout different asset classes and financial markets, providing excess returns even after controlling for other common risk factors (Asness et al., 2013). Despite consistent returns with a high Sharpe Ratio, deeper analysis of momentum-based portfolios has found that they often experience extreme crashes during turbulent markets. Numerous potential explanations have been proposed for this behavior, demonstrating improvements to the momentum factor by accounting for this risk. Barrosa and Santa Clara (2015) suggest that this crash risk is momentum-specific and is caused by time-varying volatility. An entirely different rationale is based on cross-sectional stock volatility rather than any risk inherent to the momentum strategy, addressed by Rachev et al. (2007) using the return to variance ratio. Fan et al. (2022) take this a step further by adjusting the risk calculation based on market conditions rather than using a static risk penalization. Although accounting for crash risk provides clear benefits to a momentum strategy, it remains to be seen what factor best captures this risk and how these findings extend to an ensemble machine learning approach.

Within factor investing, models predict future stock returns based on characteristics that represent an exposure to underlying risk factors. Fama, E., & French, K. (1992) introduced one of the most well-known asset pricing models basing stock returns on the stock's linear exposure to three factors; the market risk, how small firms outperform large firms, and how high-value firms overperform low-value ones. Machine learning extends traditional models like the Fama-French three-factor model by removing the linearity assumption and recognizing that firm factors and their impact on stock prices are time-varying. This leads to numerous algorithmic approaches such as a decision tree, random forest, or neural network. Although they generally outperform traditional linear models, it can be difficult to know what the best machine learning algorithm is, or how to avoid issues caused by overfitting or randomized outcomes. This leads to ensemble learning which aggregates the outputs of multiple different machine learning models to make a final prediction. As shown by Mcwera, et al. (2023), an ensemble approach combines the strengths of the constituent models, improving predictive power and overall model stability.

This paper will combine the above research by utilizing an ensemble approach to machine learning to test different risk-adjusted momentum factors.

Research Questions

This report will address two key questions:

1. Which risk-adjusted momentum factor performs best within our ensemble factor investing approach?
2. How do these different portfolios capture the crash risk within the Covid market crash?

Variables and Measures

Key features and portfolios utilized include:

1. Returns (RET). This is computed by the Center for Research in Security Prices (CRSP) and is the return of the stock over the past month including adjustments for stock splits and dividends.
2. Future return (FRET). This is our response variable and is simply the stock return (RET) for the next month.
3. Excess return (XRET). This is computed as the stock return (RET) minus the risk-free rate (RF).
4. Market capitalization (MCAP). This is simply the number of shares of a stock (SHROUT) multiplied by the price of the stock (PRC).

5. Book-to-market ratio (BMKT). This is the book value of a stock as measured by the common equity value (CEQ) divided by the market capitalization (MCAP).
6. Volatility (VOL). This is the daily volatility of the stock over the past year (assuming 252 trading days in a year) and is computed as $\sqrt{\frac{21}{252}\sum_{j=0}^{251}(r_{d_{t-1-j}})^2}$ where r_d is the asset return (RET) on day d as per Fan et al. (2022).
7. Momentum (XRET11). This is computed as the average excess stock return (XRET) over the past 12 months, excluding the most recent month to prevent short term reversals as per Daniel and Moskowitz (2016).
8. Momentum Decile Portfolio (WML). This is a zero-cost portfolio from the Fama-French database that longs stocks with high momentum and shorts those with low momentum.
9. Excess return of market (XRM). This is also a portfolio from Fama-French that measures the returns of the market as a whole, minus the risk-free rate.

Risk adjusted momentum factors tested:

1. Return-variance ratio (RV). This method is proposed by Rachev et al. (2007) as the ratio of momentum returns to variance, penalizing risk further than the traditional Sharpe Ratio. In our case, this is computed as $\frac{XRET11}{VOL^2}$.
2. Generalized risk-adjusted momentum (GRJMOM). This method is proposed by Fan et al. (2022) as $\frac{XRET11}{VOL^N}$. N is a tuning parameter between 0 and 4 (with intervals of 0.1) that adjusts the risk according to market conditions. In each month, a portfolio is created for every possible value of N. This zero-cost portfolio longs the top decile of stocks based on GRJMOM and shorts the bottom decile. The N selected in practice for a given month is the value used by the portfolio with the highest Sharpe Ratio over an expanding window including all previous periods.

3. Risk-managed momentum (WML*). This is a strategy that invests with dynamic weightings in the WML portfolio proposed by Barroso and Santa-Clara (2015), scaling the amount invested by a factor of $\frac{\delta_{target}}{\widehat{\delta_t}}$. δ_{target} is a target volatility (chosen as 12%), and $\widehat{\delta_t}$ is the daily realized volatility of the WML portfolio over the past half year (assumed to be 126 trading days). This is computed as $\sqrt{\frac{21}{126}\sum_{j=0}^{125}(r_{WML,d_{t-1-j}})^2}$ where r_{WML,d_t} is the daily return of WML on day t.

Application of the Machine Learning Approach to FI

Ensemble Approach

An ensemble approach to machine learning is utilized to predict portfolio returns and create investment strategies. This approach was selected to combine the predictive power of the constituent models and reduce overfitting. For a given set of predictive factors, five different models are created; an elastic net, binary decision tree, random forest, boosted decision tree, and neural network. Then, two linear combinations of these models are used; one that is equally weighted, and one that is optimized on the training data to reduce the squared error. The optimal return method has the added constraint that all model weights are positive and add to 1, preventing shorting poorly performing models. This methodology was proposed by Breiman, L. (1996) as solving the following constraint satisfaction problem for the weights (**w**) on the covariance matrix of errors (Σ).

$$\boldsymbol{w}^* = \underset{\boldsymbol{w}}{argmin}(\mathbf{w}'\Sigma\mathbf{w})\ such\ that\ \begin{Bmatrix} \mathbf{1}'\boldsymbol{w} = \mathbf{1} \\ w_i \geq 0\ \forall w_i \end{Bmatrix}$$

Elastic Net

A standard linear regression of $y = \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$ explains returns (y) based on a linear combination of predictive factors (x_i). A model is fitted by finding the set of regression coefficients (β_i), that best explain the linear relationship between returns and the predictor by minimizing the sum of squares of the error.

Ridge regression looks to shrink these coefficient values towards zero, reducing the risk of overfitting and allowing the model to perform better on out-of-sample data. Lasso regression looks to solve the same problem by dropping factors that underperform. An elastic net combines the two approaches, penalizing coefficients and eventually dropping those that underperform based on a mixing parameter ($\propto$) that measures how much of each method to use.

Binary Decision Tree

A decision tree repeatedly splits the data into homogenous clusters based on shared characteristics. Each split is based on a binary comparison to one feature within the dataset, selecting the feature that minimizes the variance within resultant clusters. Predictions are created for a new piece of data by finding the cluster it belongs to through moving down the tree and returning the average value for data points in that cluster. To prevent overfitting, there is generally penalization on each additional leaf.

Random Forest

A random forest uses an ensemble approach, training multiple decision trees and averaging their results. Each tree is trained on a subset of the training data and features in a process known as bagging. This sampling approach better replicates the underlying data, creating a diverse set of trees that when aggregated produce accurate predictions.

Boosted Decision Tree

This is another model that uses an ensemble of decision trees, but these are trained iteratively where each new tree is trained on the error of the previous iteration, gradually reducing errors over time. The algorithm used is XGBoost which minimizes both the training loss function, and an additional penalty based on the complexity of each tree. To prevent overfitting, the number of iterative steps needs to be limited.

Neural Network

The core of a neural network is a perceptron which is simply a linear model of weights along with an activation function that transforms the perceptron’s output, introducing non-linearity. A neural network is then composed of numerous perceptrons that feed their outputs into each other. A simple feed-forward neural network consists of a directed acyclic graph of perceptrons organized in layers.

These are trained by initializing random weights and gradually updating them by using a backpropagation of the model’s error on the training data. This iterative approach is called gradient descent, and tends towards local error minima, at a rate know as the learning rate.

All training done uses the sigmoid activation function which is $f(x) = \frac{1}{1+e^{-x}}$.

Experimental Methodology

Data Source

All data for the experiments was sourced from the Wharton Research Data Services (WRDS). Datasets used include CRSP Monthly Stock, CRSP Daily Stock, Compustat Fundamentals Annual with CRSP Monthly Stock, 5 Factors Plus Momentum - Monthly Frequency, and 5 Factors Plus Momentum - Daily Frequency. A subset of 1159 stocks was selected from the CRSP monthly data from 1995 to 2024, using only those that had complete data over the period. Our final dataset only uses data between January 2004 and December 2023, but this extended period of data was sourced since some factors require rolling calculations. The unique stock id used was CRSP's PERMNO which remains constant across CRSP data, even through changes in the stock's ticker.

Independent Datasets

1. Based on the set of all monthly CRSP stock data, a list of PERMNOs is created for stocks that have full data over the entire period.
2. This PERMNO list is then used to download the remaining CRSP and Compustat data over the period.
3. Monthly CRSP data is used to compute stock returns (RET), price (PRC), and shares outstanding (SHROUT).
4. Daily CRSP data is used to compute the rolling 252-day volatility for each stock (VOL), as per Fan et al. (2022).
5. CRSP-Compustat data is used to find the common equity value for each stock (CEQ). This is quarterly data, so results are rolled upwards, providing the most recent data available for each month (rolling down would cause look-ahead bias).
6. Fama-French 5-factor monthly data is used to find the monthly portfolio returns for the momentum (WML), and beta (XRM) portfolios, in addition to the risk-free rate (RF).
7. Fama-French 5-factor daily data is used to calculate the 126-day rolling volatility for the momentum portfolio (WML_VOL).
8. Daily datasets are changed to monthly by selecting the data on the last day of each month or the last available day if this is missing.
9. All datasets are modified to use the end-of-month convention, and any rows with missing data are removed.

Combining Datasets

1. All datasets are combined and formatted into monthly stock data using the unique row keys of PERMNO and DATE.
2. Basic features are calculated including excess return (XRET), future return (FRET), market capitalization (MCAP), book-to-market ratio (BMKT), rolling 11-month return (RET11), momentum (XRET11), and the return-variance ratio (RV), as defined earlier.
3. The GRJMOM factor for each stock is computed as suggested by Fan et al. (2022).
4. The WML* portfolio returns are calculated as suggested by Barroso and Santa-Clara (2015).
5. Select the data from the full period: January 2004 to December 2023.
6. The factors XRET11, BMKT, MCAP, RV, and GRJMOM are all normalized between 0 and 1 using the empirical cumulative distribution function (ECDF) of the factor values for each month.

Ensemble Machine Learning

1. The dataset is split into training data from January 2004 to December 2013, and testing data from January 2014 to December 2023.
2. First, all five of the machine learning models were trained separately on the training data. Hyperparameters were hard coded based on training time, rather than performance.
   a. The elastic net is trained with a hard coded alpha but uses cross-validation to find the optimal lambda parameter.
   b. The decision tree and random forest are trained.
   c. The boosted decision tree is trained only on extreme values (in the first and fifth quantiles), again reducing training time.

d. The neural net is trained with layers containing 64, 16, 4, and 1 node respectively, all using the sigmoid activation function (except the output layer).

3. Each individual model is tested on the training data, finding model errors.
4. A solution is found to the constraint satisfaction problem defined earlier, determining the optimal linear combination of the five models.
5. The ensemble model is used to predict returns over the testing data as the weighted sum of predicted returns of the five trained models. Both the optimal weighting method is used, in addition to an equal weighted approach.
6. A zero-cost ensemble portfolio is created over the testing data in each month by longing the stocks in the top decile of predicted returns and shorting those in the bottom decile.
7. The returns of each of these portfolios are analyzed using summary statistics and graphing their distribution over time.

Results and Discussions

Summary Statistics

The table below contains summary statistics for each portfolio over the test data. Note that the mean return, standard deviation, and Sharpe Ratio are annualized. As stated earlier, each portfolio tested uses the three Fama-French factors (XRM, MCAP, BMKT), in addition to a momentum factor (XRET11, RV, GRJMOM, WML*, or all four factors - ALL). For each set of factors, two ensemble portfolios were tested using the optimal weighting (OP), and equal weighting (EQ).

	Mean Return	Standard Deviation	Sharpe Ratio	Skew	Kurtosis
XRET11 OP	0.0884	0.1509	0.5859	1.5259	7.2322
XRET11 EQ	0.1086	0.1872	0.5802	1.3261	4.5167
RV OP	0.1156	0.1400	0.8251	1.2818	4.1740
RV EQ	0.1294	0.1586	0.8161	1.0374	1.9137
GRJMOM OP	0.1055	0.1372	0.7691	1.4255	4.3905
GRJMOM EQ	0.0967	0.1559	0.6202	1.5153	5.0246
WML* OP	0.1291	0.1448	0.8918	1.0522	3.4384
WML* EQ	0.1420	0.1634	0.8687	1.5607	4.7303
All OP	0.1263	0.1554	0.8132	1.2690	2.8187
All EQ	0.1461	0.1739	0.8400	2.0903	8.0979

We can see an improvement in model performance using risk-adjusted momentum factors, with the Sharpe Ratio increasing from around 0.6 to nearly 0.9. There also is a decline in return kurtosis, with the one of the base momentum portfolios having a kurtosis of 7.2322, while other models get as low as 1.9137. This suggests there is less data in the tails of the distributions, meaning there are less extreme return values and therefore crashes. These results are consistent with the source papers for these factors, demonstrating that risk-adjusted momentum better indicates future returns. The discrepancy observed here is far smaller than the original papers which generally doubled the Sharpe Ratio through their risk adjustment. This is explained by the inclusion of more factors and the robust machine learning algorithms used, improving profitability of even the base momentum factor.

The best model was WML* using the optimal weighting approach, resulting in an average annual return of 12.91%, and a Sharpe Ratio of 0.8918. This suggests that the time-varying volatility of the momentum portfolio is a key component of momentum risk. By scaling exposure to this risk, the portfolio’s variance can be normalized across time while still retaining high returns. This portfolio even outperforms the one that included all the momentum factors. This is a somewhat surprising result, since intuitively more factors should increase machine learning accuracy. Instead, including poorly performing factors drags down performance by overfitting to the training data on patterns that don’t generalize.

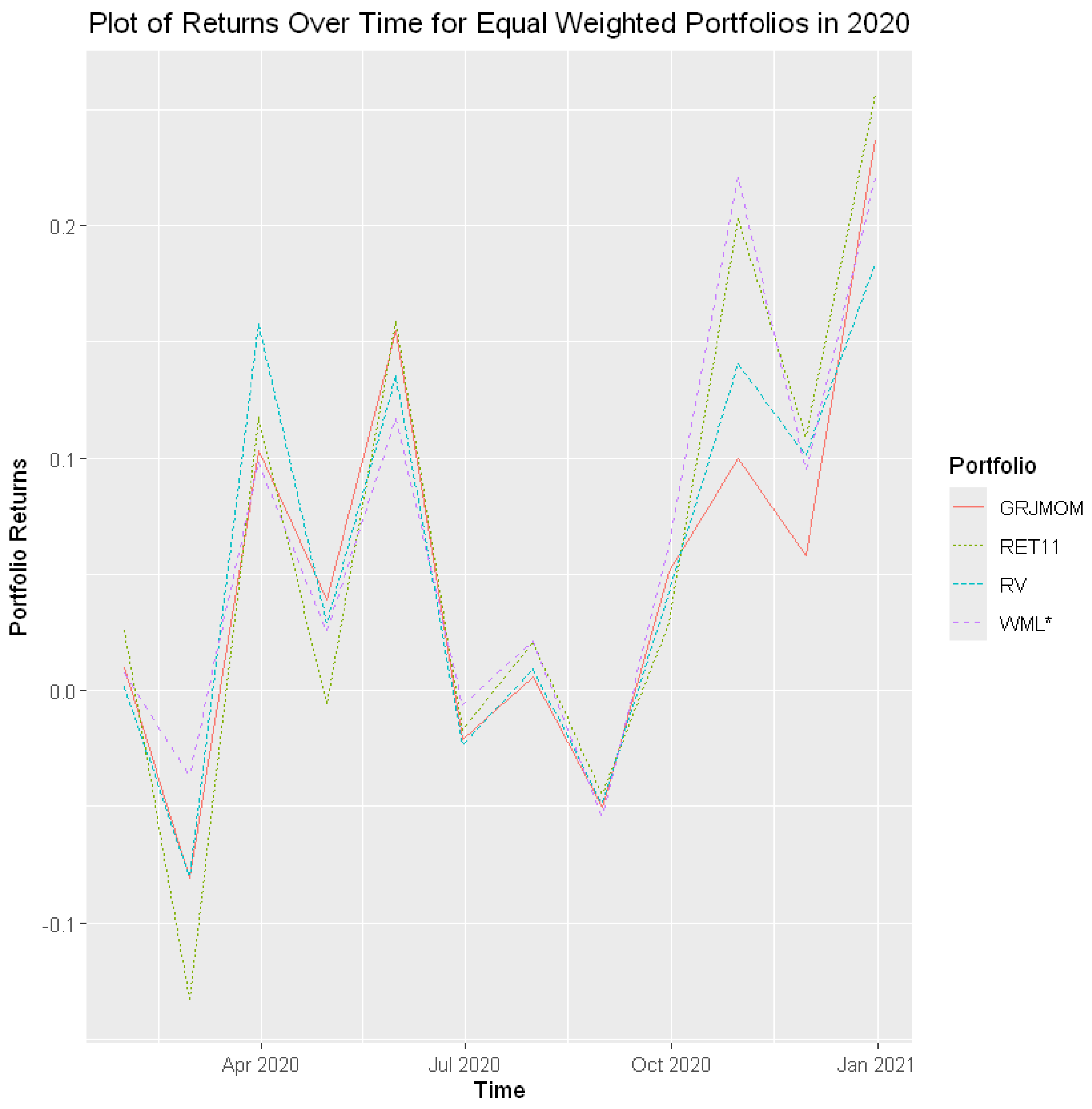
Another unexpected observation is that RV’s static risk penalization factor is more effective than the dynamic metric GRJMOM. The optimal N parameters based on the past data appear to under-penalize risk, resulting in disappointing results. The optimal parameters calculated differ immensely from the source literature which over most periods has a far larger penalization factor for equity (generally roughly 2.5 versus 0.5) that would perform better over our testing data. This large discrepancy is likely caused by the expanding window approach where our estimates of the optimal N values are based on a far shorter timeframe.

Unsurprisingly, the optimally weighted approach generally outperforms the equally weighted one. This makes sense since both the neural network and basic decision tree had disappointing performance on the testing data, dragging down the accuracy of the estimates.

Another fascinating pattern is that most models have a positive skew of around 1, as opposed to the negative skew that is found with traditional momentum-based portfolios analyzed in other literature. This is likely caused by adding factors like market capitalization that capture additional underlying risk factors, as opposed to single-factor models. Additionally, more complex models such as neural networks can better capture underlying data relationships due to the lack of imposed restrictions like linearity. The specific training period used between 2004 and 2013 likely also played a key role, with the 2008 financial crisis composing a large portion of the training set.

Crash Performance

Next, we examine the returns of the four main equally weighted portfolios in 2020 to determine how they performed during the Covid market crash.



We can see that all the portfolios are heavily correlated, following very similar return patterns which is understandable due to the similarity in factors. The unadjusted momentum (RET11) fails to adequately capture momentum crash risks, facing immense losses of around 13% in

February 2020. The risk-adjusted factors perform far better during this crash, sustaining far lower losses while still participating in the subsequent market recovery. The best factor identified earlier, WML*, faces the lowest losses of only around 4%. At least in this instance, momentum crash risk can be mitigated through various risk adjustment methods. This corroborates the findings of the underlying literature, demonstrating that their results are applicable even within an ensemble machine learning framework.

Related Work

Projects 1 and 2 are highly related, therefore the reader is referred to the GPR #1

report for related work.

Conclusions and Recommendations

The strong performance of applying risk-mitigation scaling to momentum factors holds even within an ensemble machine learning approach, effectively predicting future stock returns. The best risk adjustment factor tested is WML*, suggesting that crash risk is largely defined by the time-varying volatility of momentum portfolios. Accounting for this effectively captures crash risk and improves performance during volatile markets.

The limitations of the testing done suggest a clear path for further research. Firstly, the scope of the data used was small, using only 1159 stocks and the testing and training datasets contained only 10 years each. Additionally, hyperparameters were selected to optimize runtime rather than accuracy, reducing the performance of the machine learning models. Finally, the inclusion of more explanatory factors would better model the underlying risk structure of stock returns, improving predictive accuracy. Overall, more robust results could be achieved through further research using a larger dataset, stronger hyperparameters, and more factors.

Works Cited

Asness, C. S., et al. 2013. Value and Momentum Everywhere. The Journal of Finance (New York), 68(3), 929–985. https://doi.org/10.1111/jofi.12021.

Barroso, P., & Santa-Clara, P. 2015. Momentum has its moments. Journal of Financial Economics, 116(1), 111–120. https://doi.org/10.1016/j.jfineco.2014.11.010.

Breiman, L. 1996. “Stacked Regressions.” *Machine Learning*, vol. 24, no. 1, 1996, pp. 49–64, https://doi.org/10.1007/bf00117832.

Carhart, M. 1997. On Persistence in Mutual Fund Performance. The Journal of Finance (New York), 52(1), 57–82. https://doi.org/10.1111/j.1540-6261.1997.tb03808.x.

Daniel, K., & Moskowitz, T. J. 2016. Momentum crashes. Journal of Financial Economics, 122(2), 221–247. https://doi.org/10.1016/j.jfineco.2015.12.002.

Fama, E., & French, K. 1992. The Cross-Section of Expected Stock Returns. The Journal of Finance (New York), 47(2), 427–465. https://doi.org/10.1111/j.1540-6261.1992.tb04398.x.

Fan, M., et al. 2022. Momentum and the Cross-section of Stock Volatility. Journal of Economic Dynamics & Control, 144, 104524-. https://doi.org/10.1016/j.jedc.2022.104524.

Jegadeesh, N., & Titman, S. 1993. Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency. The Journal of Finance (New York), 48(1), 65–91. https://www.jstor.org/stable/2328882.

Mcwera, et al. 2023. “Predicting Stock Market Direction in South African Banking Sector Using Ensemble Machine Learning Techniques.” *Data Science in Finance and Economics*, vol. 3, no. 4, 2023, pp. 401–26, https://doi.org/10.3934/DSFE.2023023.

Rachev, et al. 2007. “Momentum Strategies Based on Reward–Risk Stock Selection Criteria.” *Journal of Banking & Finance*, vol. 31, no. 8, 2007, pp. 2325–46, https://doi.org/10.1016/j.jbankfin.2007.02.006.

Saifan, Ramzi, et al 2020. “Investigating Algorithmic Stock Market Trading Using Ensemble Machine Learning Methods.” Informatica (Ljubljana), vol. 44, no. 3, 2020, pp. 311–25, https://doi.org/10.31449/INF.V44I3.2904.