World Journal of Management Science
ISSN: 2959-9628
DOI: 10.61784/wms3014

STOCK MARKET PREDICTION STRATEGY BASED ON REGULARIZED MULTIPLE ENSEMBLE LEARNING

HaoRan Mo[1,#], HongYe Qian[1,#], ZhiYuan Zhang[1,#], SiQi Zhang[1], ChenXu Zhu[1], JiHeng Hou[1,*], HaiLan Sun[1], XuLe Cheng[2], Shi Chen[3]
[1]*University of Liverpool, Liverpool L69 7ZX, Merceyside, United Kingdom.*
[2]*Macau University of Science and Technology, Taipa 999078, Macau, China.*
[3]*National University of Defense Technology, Changsha 410073, Hunan, China.*
[#]*HaoRan Mo, HongYe Qian and ZhiYuan Zhang contributed equally to this work and they are all first authors.*
Corresponding Author: JiHeng Hou, Email: sgjhou5@liverpool.ac.uk

Abstract: The capital market has always aimed to use human intelligence algorithms to predict stock trends. However, due to the stock market's complexity and variability, accurately predicting the stock market and enhancing profits remains challenging and crucial. Internal and external factors affect the stock market, making it difficult to forecast its movements with precision. To improve the prediction accuracy, this paper proposes a Boosting ensemble learning method with regularized weights, which combines support vector Machine (SVM), decision tree and ridge regression in a gradient boosting framework. The algorithm has been recognized for improving prediction performance by exploiting the strengths of a single model and mitigating its weaknesses. A new method of model weight adjustment has been proposed to speed up the training speed of Meta-learning. This study uses ensemble learning to capture complex patterns and trends in stock market data, aiming to build a robust prediction model and improve generalization ability. We evaluate our model by back-testing different stocks in the US. Our model achieves significant prediction accuracy improvement. Compared with the single model method, the MSE and MAE of the back-test data and the actual trend are significantly reduced, and the volatility is also significantly improved.

Keywords: Ensemble learning; Boosting; Regularized weights; SVM; Meta-learner; Ridge regression

1 INTRODUCTION

The unpredictability of financial markets and the large risks involved in investment decisions highlight the importance of accurately predicting the stock market. Traditional statistical methods such as time series forecasting methods and regression forecasting methods, including ARIMA [1], prophet [2] and Exponential Smoothing (ETS) [3], capture the trend change from the perspective of statistics. While linear regression, ridge regression [4] and lasso regression [5] assume the stock trend as a regression problem to predict, machine learning algorithms ensure higher accuracy by capturing complex patterns in historical data. In the latest results, deep learning proposes deep learning prediction methods based on LSTM [6] and CNN [7] to predict stock trends by capturing long-term dependencies in time series data [8].

However, a single classifier system is often inadequate when dealing with the multifaceted nature of the stock market. Ensemble learning addresses this gap by combining multiple classifiers to form more effective predictive entities. Specifically, ensemble methods like bagging, boosting, and stacking have been shown to outperform individual classifiers in various domains, including stock market prediction. Boosting, a sequential ensemble method that focuses on successively correcting the previous error, has been particularly successful. Gradient boosting is a boosting algorithm that iteratively improves the model by highlighting instances that were misclassified by previous iterations. By integrating support vector machine, decision tree and ridge Regression into Gradient Boosting Regression (GBR) [9], the complementary advantages of these algorithms are used to improve the prediction accuracy and reliability.

The study uses ensemble learning to overcome the limitations of a single algorithm and aims to provide a comprehensive solution for stock market forecasting. This paper proposes an ensemble learning prediction model based on Boosting method to regularize model weights. The model includes decision tree model, Ridge Regression model and Support Vector Regression model [10]. At the same time, we propose a model training method that can speed up the Meta-learner by regularizing the model weight parameters and integrate it into our ensemble learning Meta-Learner [11]. The experimental results show that our regularized Weighted Ensemble Learner achieves good results in stock prediction tasks. The MSE and MAE of the backtest data and the actual trend are significantly reduced, and the maximum retrace, and volatility perform well. Our method further makes progress on the stock market prediction problem.

Our contributions as follows,

- We designed an ensemble meta-learner based on decision tree model, ridge Regression model and Support Vector Regression model for stock prediction problem.
- We propose a method to regularize the parameter adjustment between Meta-learner baseline models, which effectively improves the iterative training time of ensemble learning.

2 RELATED WORK

2.1 Traditional Stock Forecasting Methods

Stock market prediction has long been a significant area of interest for researchers and practitioners due to its potential for substantial financial gains and the inherent challenges it presents. Traditional methods for stock market prediction, including technical analysis, fundamental analysis, time series analysis, and the random walk theory, have laid the foundation for understanding and forecasting stock price movements.

Technical analysis involves studying past market data, primarily price and volume, to identify patterns and make predictions about future price movements. The underlying assumption of technical analysis is that historical price movements tend to repeat themselves due to market psychology. Key tools and techniques in technical analysis include candlestick charts, moving averages, and the relative strength index (RSI) [12]. Candlestick charts display the open, close, high, and low prices for a given period, helping traders identify patterns and potential reversals. Moving averages, such as the simple moving average (SMA) and exponential moving average (EMA), help smooth out price data to identify trends and reversals by averaging a stock's price over a specific period. The RSI measures the speed and change of price movements to identify overbought or oversold conditions [13].

Fundamental analysis assesses a stock's intrinsic value by examining related economic, financial, and other qualitative and quantitative factors. The goal is to determine a company's actual worth and predict future stock price movements based on this assessment. Key components of fundamental analysis include financial statements, industry conditions, and macroeconomic indicators. Analysis of the balance sheet, income statement, and cash flow statement helps evaluate a company's financial health and profitability. Industry conditions involve studying market trends, competition, and the regulatory environment to understand a company's market position and potential. Macroeconomic indicators, such as GDP growth rates, inflation, and unemployment rates, provide insights into the broader economic environment that can impact a company's performance [14].

Time series analysis is a statistical method used to analyze time-ordered data points to identify patterns and predict future values. Common time series models include autoregressive (AR) models, moving average (MA) models, and autoregressive integrated moving average (ARIMA) models [1]. AR models predict future values based on past values, assuming that current values are a linear combination of past values. MA models predict future values based on past error terms, assuming that current values are a linear combination of past errors. ARIMA models combine AR and MA models and are suitable for non-stationary time series data, transforming non-stationary data into stationary data through differencing [15].

2.2 Machine Learning Methods in Stock Forecasting

Machine learning has revolutionized the field of stock forecasting by providing sophisticated tools to model and predict complex market behaviors. This section reviews some of the most widely used machine learning methods in stock forecasting, including traditional algorithms, deep learning techniques, and ensemble methods.

2.2.1 Linear regression

Linear regression is one of the simplest and most interpretable models for predicting stock prices. It models the relationship between a dependent variable (stock price) and one or more independent variables (predictors) by fitting a linear equation to the observed data. However, its assumption of linearity often limits its performance in capturing the complex, nonlinear patterns in stock data [16].

2.2.2 Support Vector Machines (SVM)

SVMs are powerful for classification and regression tasks. They work by finding the hyperplane that best separates different classes in the feature space [17]. In stock forecasting, SVMs are used to predict stock price trends or classify stock movements (upward or downward). Their ability to handle high-dimensional data and prevent overfitting makes them popular in this domain.

2.2.3 Decision Trees and Random Forests

Decision trees split the data into branches to make predictions based on the values of the input features [18]. Random forests, an ensemble of decision trees, improve the prediction accuracy by averaging the predictions of multiple trees [19]. They are robust to overfitting and can handle large datasets with many features, making them suitable for stock forecasting.

2.2.4 Recurrent Neural Networks (RNN)

RNNs are specifically designed for sequential data, making them ideal for time series forecasting such as stock prices. They maintain a hidden state that captures information from previous time steps, enabling them to model temporal dependencies. Long Short-Term Memory (LSTM) networks, a type of RNN, are particularly effective in handling long-term dependencies and mitigating the vanishing gradient problem [20].

2.3 Ensemble Learning Method

In the field of machine learning, ensemble learning methods are widely used due to their ability to improve model performance and stability. Traditional time series analysis methods (e.g., ARIMA and GARCH) and statistical models (e.g., multiple regression analysis and support vector machines) often struggle with capturing complex market dynamics and nonlinear relationships [21] [22]. With advancements in machine learning, particularly deep learning models such as LSTM and CNN, researchers have achieved significant results in stock prediction [6] [23]. However, these methods still face challenges such as overfitting and poor adaptability to market fluctuations.

Ensemble learning methods can effectively overcome the limitations of a single model by building multiple models and combining their prediction results. Common ensemble learning methods include Bagging and Boosting. Bagging, such as random forests, improves the accuracy and stability of models by building multiple decision trees and combining their predictions. Boosting methods such as AdaBoost [24], GBM [25], XGBoost [26], LightGBM [27], and CatBoost [28] optimize the overall model performance by gradually adding weak classifiers, especially when dealing with large-scale data and classification features.

In addition, stacking generalization and Voting are also commonly used ensemble methods. Stacked generalization improves prediction accuracy by training multiple base models and using a meta-model to combine the predictions of these base models [11]. The voting method uses hard voting or soft voting to carry out majority voting or weighted average of the prediction results of multiple models [29]. Blending is similar to Stacking, but typically only one validation set is used to train the meta-model without cross-validation [30]. Bagged Boosting combines the advantages of Bagging and Boosting and improves the robustness of the model by resampling the data and training with Boosting [31]. Bayesian Model Averaging (BMA) uses Bayesian method to weighted average multiple models, which makes the prediction results of the model more reliable [32].

3 METHODOLOGY

In this section, we will first introduce three baseline models: Decision Tree Regressor, Ridge Regressor, and Support Vector Regressor. Then we will introduce the Regularization Weighted Meta Learner used to train the Meta-learner.

3.1 Decision Tree Regressor

Decision Tree Regressors (DTR) are non-parametric, supervised learning algorithms used for regression tasks that partition the data into subsets based on feature value conditions, forming a tree-like structure. The methodology for training a Decision Tree Regressor involves several key steps:

3.1.1 Tree construction

Decision Tree Regressors (DTR) are non-parametric, supervised learning algorithms used for regression tasks that partition the data into subsets based on feature value conditions, forming a tree-like structure. The methodology for training a Decision Tree Regressor involves several key steps:

$$MSE_m = \frac{1}{N_m}\sum_{i\in N_m}(y_i - \widehat{y_m})^2$$

where $\mathrm{N_m}$ is the number of observations in node $\mathrm{N_m}$, $\mathrm{y_i}$ is the actual value of the i-th observation, and $\widehat{\mathrm{y_m}}$ is the predicted value for the observations in node m, typically taken as the mean of the target values within the node.

3.1.2 Splitting criterion

The decision on how to split the data at each node is based on the Reduction in Impurity (RI), which is calculated as the difference in the impurity measure before and after the split. For a potential split s that partitions the data into left and right nodes, L and R , the Reduction in Impurity is given by:

$$\mathrm{RI}(s) = \mathrm{I_{parent}} - \left(\frac{N_L}{N}\times \mathrm{I_L} + \frac{N_R}{N}\times \mathrm{I_R}\right)$$

where I is denoted the impurity, N is the total number of observations at the parent node, N_L and N_R are the number of observations in the left and right nodes respectively, and Impurity I_L and Impurity I_R are the impurity measures of the left and right nodes, calculated similar to $\mathrm{MSE_m}$ for each node.

where I is denoted the impurity, N is the total number of observations at the parent node, N_L and N_R are the number of observations in the left and right nodes respectively, and Impurity I_L and Impurity I_R are the impurity measures of the left and right nodes, calculated similar to $\mathrm{MSE_m}$ for each node.

3.1.2 Pruning:

To prevent overfitting, the tree may be pruned by removing branches that have little power to classify instances. This can be done using cost-complexity pruning, which introduces a complexity parameter, α, that penalizes the tree size. The goal is to find the subtree that minimizes the following cost-complexity function:

$$R_{\alpha}(T) = R(T) + \alpha \times |Leaves|$$

where $R(T)$ is the total misclassification rate of the tree T, | Leaves | is the number of terminal nodes in the tree, and $\alpha \geq 0$ is the complexity parameter.

3.2 Ridge Regressor

Ridge Regression is a supervised learning algorithm specifically for regression that aims to solve multicollinearity problems and reduce the risk of model overfitting by introducing a regularization term. The methodology of ridge regression is described in detail in this section.

Ridge regression improves the fit of the model to the data by adding an L2 regularization term to the ordinary least squares method. This regularization term penalizes the size of the model coefficients, preventing them from becoming too large during fitting. The objective function for ridge regression can be expressed as follows.

$$RSS_{\text{ridge}}(w) = \sum_{i=1}^{N} (y_i - w^T x_i)^2 + \lambda \sum_{j=1}^{p} w_j^2$$

Where N is the number of samples, y_i is the target variable for the i-th observation, x is the feature vector for the i-th observation, w is the vector of model coefficients, p is the number of features, and n is the number of features. λ is the regularization strength parameter.

The regularization parameter λ controls the influence of the regularization term. When $\lambda = 0$, ridge regression is equivalent to ordinary least squares regression. As the value of λ increases, the penalty on the coefficients becomes stronger and the complexity of the model decreases, reducing the possibility of overfitting.

The training process of ridge regression models involves minimizing a loss function with a regularization term. This can be done by solving the following optimization problem: $\min_{w}\{RSS_{\text{ridge}}(w)\}$ By differentiating the loss function and setting it equal to zero, we can find an optimal set of coefficients w to achieve the best fit on the training data.

The closed-form solution of ridge regression can be obtained directly by matrix operations with the following formula.

$$\boldsymbol{w} = (X^T X + \lambda I)^{-1} X^T y$$

$\boldsymbol{X}$ is the design matrix, I is the identity matrix, and y is the target variable vector.

3.3 Support Vector Regressor

Support Vector Regressor (SVR) is a regression analysis method based on Support Vector Machine (SVM). SVR looks for a hyperplane in a high dimensional space to maximize the edges of this hyperplane while keeping the error of all data points within a certain threshold. The methodology of SVR is detailed in this section. The basic idea of SVR is to find a linear function in the feature space such that the gap between the predicted value and the actual value is as small as possible, while ensuring that the complexity of the model is under control. Specifically, this optimization problem can be formulated as follows,

$$\min_{\boldsymbol{w},b,\boldsymbol{\xi},\xi^*} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i^*)$$

$$\text{subject to} \begin{cases} y_i - \left(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b\right) \le \epsilon + \xi_i, \\ \left(\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_i) + b\right) - y_i \le \epsilon + \xi_i^*, \\ \xi_i, \xi_i^* \ge 0, i = 1, 2, \ldots, N \end{cases}$$

Where w is the normal vector of the hyperplane, b is the bias term, C is the regularization parameter, and ϵ is the threshold of the insensitive loss function. ξ and ξ* are loose variables, and ϕ is a function that maps to a higher dimensional space.

In SVR, kernel functions are usually used to deal with nonlinear problems, and kernel functions can map the input data into a high-dimensional feature space. Commonly used kernel functions include linear kernels, polynomial kernels, and Radial Basis Function (RBF) kernels. RBF kernel is a common kernel function, which is defined as follows.

$$K(x_i, x_j) = \exp\left(-\gamma |x_i - x_j|^2\right)$$

Where γ is the parameter of the kernel function, and x_i and x_j are the feature vectors.

3.4 Regularization Weighted Meta Learner

In this part, we want to propose the Regularization Weighted Meta Learner based on the ensemble learning Boosting method. We will propose Gradient Boosting based on weight regularization as our enhanced parameter update method.

$$\hat{y}^{(0)}(x) = \arg\min_{\gamma} \sum_{i=1}^{N} L(y_i, \gamma)$$

Among them, $\widehat{y^{(0)}}(x)$ is the prediction of the initial model, $L(y_i, \gamma)$ is the loss function, commonly used is the mean squared error (MSE), y_i is the true value of the i-th sample in the training data, N is the number of training samples, γ is the parameter of the initial model, such as the mean. For each step of Boosting $t = 1,2, \ldots, T$, we compute the residual of the model on the currently learned training data:

$$r_{it} = -\left[\frac{\partial L\big(y_i, \hat{y}(x_i)\big)}{\partial \hat{y}(x_i)}\right]_{\hat{y}(x_i)=\hat{y}^{(t-1)}(x_i)}$$

r_{it} is the residual at step t for the i sample, $\hat{y}(x_i)$ is the model prediction at the i sample, and $\frac{\partial L(y_i, \hat{y}(x_i))}{\partial \hat{y}(x_i)}$ is the partial derivative of the loss function with respect to the prediction. Next, we use these residuals to train the next regressor $h_t(x)$:

$$h_t(x) = \arg\min_{h\in\mathcal{H}} \sum_{i=1}^{N} L\big(r_{it}, h(x_i)\big)$$

The contribution of each regressor is adjusted by the learning rate ν, and the model complexity is controlled by a regularization term $\Omega(h)$:

$$\hat{y}^{(t)}(x) = \hat{y}^{(t-1)}(x) + \nu \cdot h_t(x) - \lambda \cdot \Omega(h_t)$$

$\widehat{y^{(t)}}(x)$ is the predicted value of the ensemble model at step t, ν is the learning rate, which controls the update rate at each step, λ is the regularization parameter, which is used to balance model complexity and fit. $\Omega(h_t)$ is the regularization term for the t-th regresses.

The regularization term $\Omega(h)$ can be L1 or $L2$ regularized, depending on the regularization strategy we want to use. For ridge regression, for example, regularization item usually L2 regularization: $\Omega(h_t) = |w_t|^2$, for the support vector regression machine, regularization item may be more complex, depending on the kernel function and other parameters. In this way, we prevent the model from becoming too complex during augmentation and help improve the generalization ability of the final model.

After T rounds, we get the final ensemble model:

$$\hat{y}^{(T)}(x) = \hat{y}^{(0)}(x) + \sum_{t=1}^{T} \left(\nu \cdot h_t(x) - \lambda \cdot \Omega(h_t)\right)$$

$\widehat{y^{(T)}}(x)$ is the final model prediction after T rounds of Boosting, where T is the total number of Boosting iterations. In this process, we will choose the λ and ν that make the whole model perform best on the validation set.

4 EXPERIMENTS

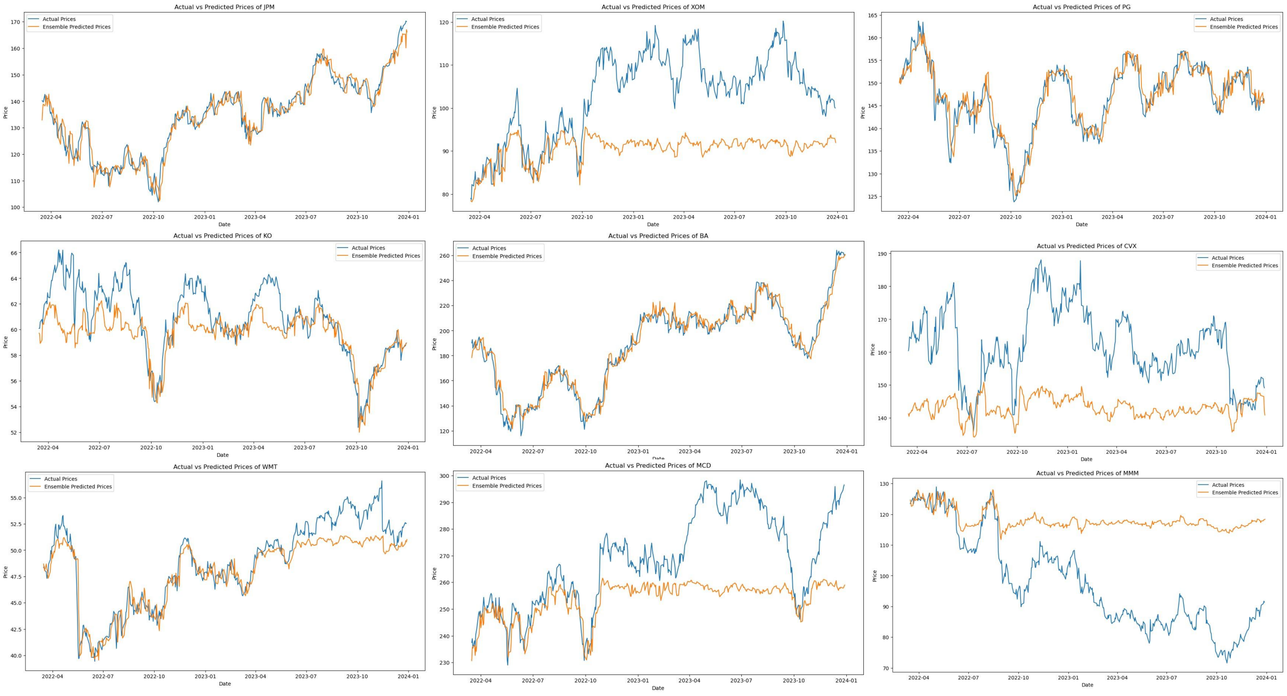

Figure 1 Single Model Prediction Results

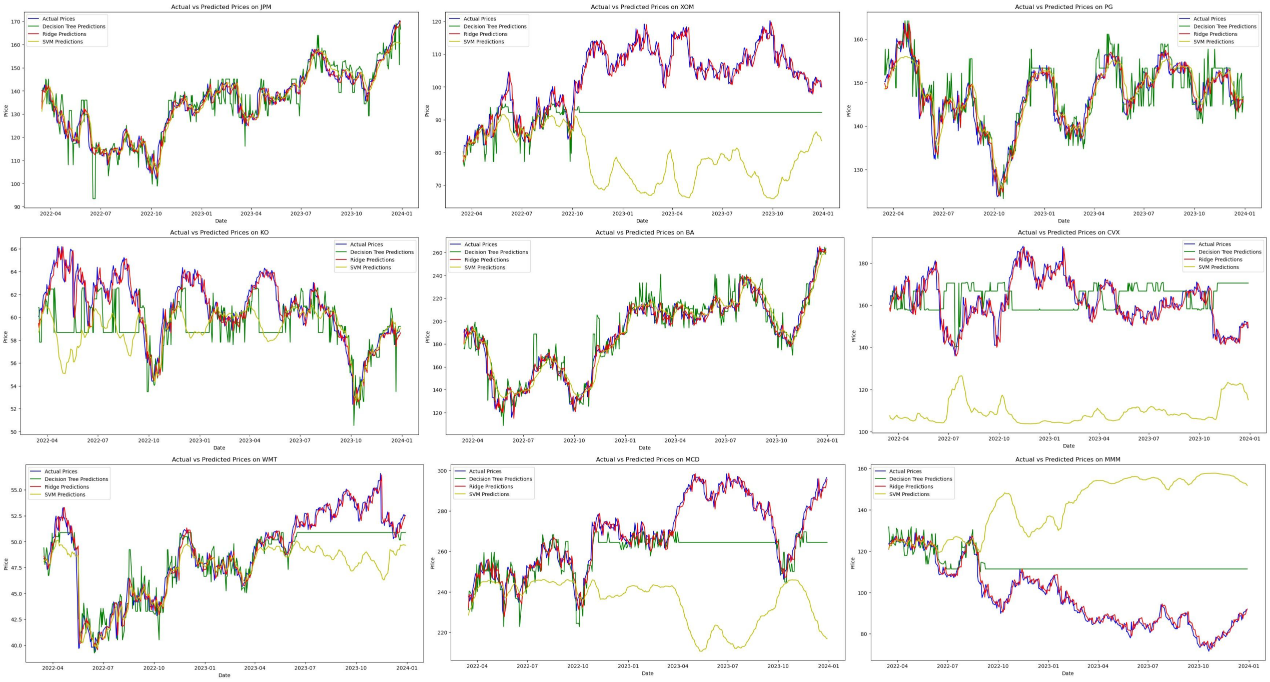

Figure 2 Ensemble Models Prediction Results

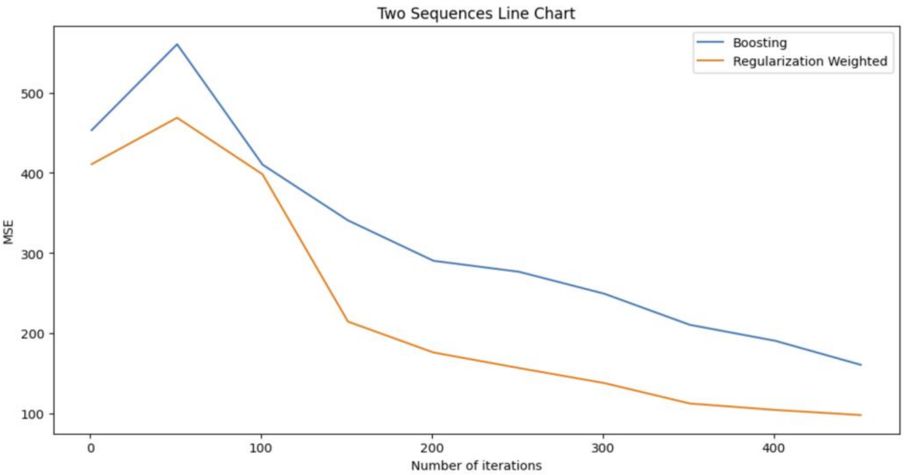

Figure 3 Performance comparison of the number of iterations between strategies

4.1 Prediction Comparison

In this part, we selected the representative blue-chip stocks of the US stock market to carry out a ten-year stock trend backtest. Here, we selected the trading data of several big blue-chip stocks from 2014-01-01 to 2024-01-01, including 'APPL', 'GOOGL', 'DOW',' AMZN ', 'MSFT ', 'META', 'NKE', 'GS'. We set back the step length measurement is 10, the random number seed to 42, and GradientBoostingRegressor as a integrated study of the learner, the number of the estimators is set to 100, the vector is set to 0.1, Max_depth set to 3. The result is shown in the figure 1,2 below, and in the table 1 below.

Table 1 The Stock Market Predictions

Stock	Model	MSE	MAE	DrawDown	Volatility
JPM	Decision Tree	26.18	3.88	-0.36	0.04
	Ridge	8.10	2.18	-0.28	0.02
	SVM	13.99	2.92	0.26	0.01
	Ensemble	9.87	2.65	-0.11	0.02
XOM	Decision Tree	204.55	12.20	-0.19	0.02
	Ridge	7.21	2.13	-0.20	0.02
	SVM	876.07	29.60	-0.28	0.01

	Ensemble	229.44	12.86	-0.15	0.01
PG	Decision Tree	12.97	2.82	-0.25	0.03
	Ridge	5.77	1.80	-0.24	0.01
	SVM	10.40	2.42	-0.19	0.00
	Ensemble	7.08	2.08	0.26	0.01
KO	Decision Tree	8.32	2.16	-0.20	0.02
	Ridge	0.81	0.66	-0.21	0.01
	SVM	11.10	2.33	-0.14	0.00
	Ensemble	6.15	2.48	0.16	0.01
BA	Decision Tree	94.35	7.26	-0.45	0.06
	Ridge	31.66	4.42	-0.41	0.03
	SVM	70.55	6.49	-0.31	0.01
	Ensemble	35.95	4.68	0.31	0.02
CVX	Decision Tree	225.49	12.70	-0.18	0.02
	Ridge	16.69	3.12	-0.25	0.02
	SVM	307.64	52.77	-0.18	0.01
	Ensemble	115.46	8.76	0.16	0.01
WMT	Decision Tree	3.18	1.38	-0.23	0.02
	Ridge	0.94	0.64	-0.26	0.01
	SVM	8.17	2.02	-0.21	0.01
	Ensemble	4.17	1.46	0.23	0.02
MCD	Decision Tree	228.25	11.15	-0.17	0.02
	Ridge	15.21	3.04	-0.16	0.01
	SVM	181.90	33.45	-0.14	0.00
	Ensemble	223.77	10.85	0.14	0.01
MMM	Decision Tree	410.62	16.97	-0.22	0.02
	Ridge	6.13	1.84	-0.24	0.02
	SVM	277.19	44.59	-0.14	0.00
	Ensemble	436.31	17.09	0.17	0.00

4.2 Ablation Study

In this part of ablation experiment, we will conduct ablation experiments on our Meta-learner and Regularization Weighted Method, and the effectiveness of these two strategies has been determined.

4.2.1 Effectiveness Experiments on Meta-learner

We designed a set of experiments on the number of iterations and the prediction effect of Gradient Boosting Regression in estimation. Here, the contrast of Boosting method and Regularization Weighted Ensemble learning method are shown in Figure 3, respectively.

From the experimental results shown in Figure 3, our regularized Weighted Ensemble Learning indeed Narrows the number of iterations required by our Gradient Boosting Regression to some extent. Thus, the effectiveness of our regularized weight adjustment strategy is verified. Bagging, Boosting and Reg_Weighted Results can be seen in table 2.

Table 2 Bagging, Boosting and Reg_Weighted Results

Model Type	Stock	MSE	MAE	DrawDown	Volatility
Bagging	DOW	1.305	0.901	-0.33	0.016
	GOOGL	15.561	3.121	-0.41	0.018
	AMZN	30.702	4.162	-0.52	0.023
	MSFT	361.975	13.218	-0.35	0.015
Boosting	DOW	1.352	0.923	-0.27	0.015
	GOOGL	19.989	3.534	-0.41	0.020
	AMZN	48.763	5.089	-0.48	0.025
	MSFT	214.401	9.798	-0.35	0.017
Reg-Weighted Boosting	DOW	1.01	0.92	0.22	0.01
	GOOGL	9.89	1.53	0.45	0.01
	AMZN	9.07	1.29	0.60	0.01
	MSFT	38.14	3.80	0.35	0.01

4.2.2 Effectiveness experiments on Regularization Weighted Method

We designed a set of experiments on the number of iterations and the prediction effect of Gradient Boosting Regression in estimation. Here, the contrast of Boosting method and Regularization Weighted Ensemble learning method are shown in Figure 3, respectively.

From the experimental results shown in Figure 3, our regularized Weighted Ensemble Learning indeed Narrows the number of iterations required by our Gradient Boosting Regression to some extent. Thus, the effectiveness of our regularized weight adjustment strategy is verified.

5 CONCLUSIONS

By combining Support Vector Machine (SVM), decision tree and ridge regression models, this study proposes a regularized weighted Boosting ensemble learning method aimed at improving the accuracy of stock market forecasting. Our empirical study shows that the method significantly reduces the mean square error (MSE) and mean absolute error (MAE) for both backtest data and actual trends, and significantly improves volatility. Our model provides a comprehensive solution to solve the stock market prediction problem, provides new ideas and methods for future stock market prediction research, and is expected to have a positive impact in practice.

COMPETING INTERESTS

The authors have no relevant financial or non-financial interests to disclose.

REFERENCES

[1] Box GE, Jenkins GM, Reinsel GC, Ljung GM. Time series analysis: Forecasting and control. John Wiley & Sons , 2015.

[2] Taylor SJ, Letham B. Forecasting at scale. The American Statistician, 2018,72 (1): 37–45.

[3] Hyndman RJ, Khandakar Y. State space models for time series forecasting in r. Journal of Statistical Software, 2002, 18 (1): 1–16.

[4] Hoerl AE, Kennard RW. Ridge regression: Biased estimation for nonorthogonal problems. Technometrics, 1970, 12 (1): 55–67.

[5] Tibshirani R. Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological), 1996, 58 (1): 267–288.

[6] Hochreiter S, Schmidhuber J. Long short-term memory. Neural computation, 1997, 9 (8): 1735–1780.

[7] Lawrence S, Giles CL, Tsoi AC, Back A. Face recognition: A convolutional neural-network approach. IEEE transactions on neuralnetworks, 1997: 8 (1): 98–113.

[8] Nti IK, Adekoya AF, Weyori BA. A comprehensive evaluation of ensemble learning for stock-market prediction. Journal of Big Data, 2020, 7 (1): 20.

[9] Friedman JH, Greedy function approximation: A gradient boosting machine, Annals of statistics, 2001, 1189–1232.

[10] Drucker H, Burges CJ, Kaufman L, Smola A, Vapnik V, Support vector regression machines. Advances in neural information processing systems, 1997, 9: 155–161.

[11] Wolpert DH. Stacked generalization. Neural networks, 1992, 5 (2): 241–259.

[12] Murphy J. Technical analysis on the financial markets. new york institute of finance. New York institute of finance, 1999.

[13] Edwards RD, Magee J, Bassetti WC. Technical analysis of stock trends. CRC press, 2018.

[14] Penman SH. Financial statement analysis and security valuation. McGraw-hill, 2013.

[15] Granger CWJ, Hatanaka M, Spectral analysis of economic time series. Princeton university press, 2015.

[16] Miller CN, Rol Rl, Taylor W, et al.. Efficient capital markets: A review of theory and empirical work. The journal of Finance, 1970, 25 (2): 383–417.

[17] Kim K-j, Financial time series forecasting using support vector machines. Neurocomputing, 2003, 55 (1-2): 307–319.

[18] Breiman L. Bagging predictors machine learning, 1996, 24 (2): 123-140.

[19] Breiman L. Random forests machine learning, 2001, 45 (1): 5-32.

[20] Sherstinsky A. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. Physica D: Nonlinear Phenomena, 2020, 404.

[21] Nerlove M. Time series analysis, forecasting, and control, 1971.

[22] Bollerslev T. Generalized autoregressive conditional heteroskedasticity. Journal of econometrics, 1986, 31 (3): 307–327.

[23] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature, 2015,521 (7553): 436–444.

[24] Freund Y, Schapire RE. A decision-theoretic generalization of online learning and an application to boosting. Journal of Computer and System Sciences, 1997, 55 (1): 119–139.

[25] Friedman JH. Greedy function approximation: A gradient boosting machine0 Annals of Statistics, 2001, 29 (5): 1189–1232.

[26] Chen T, Guestrin C. Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2016, 785–794.

[27] Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu TY. Lightgbm: A highly efficient gradient boosting decision tree. in: Advances in Neural Information Processing Systems, 2017, 30: 3146–3154.

[28] Prokhorenkova L, Gusev G, Vorobev A, Dorogush AV, Gulin A. Catboost: Unbiased boosting with categorical features. in: Advances in Neural Information Processing Systems, 2018, 31: 6638–6648.

[29] Polikar R. Ensemble based systems in decision making, IEEE Circuits and Systems Magazine, 2006,6 (3): 21–45.

[30] Sill J, Takacs G, Mackey LW, Lin D. Feature-weighted linear stacking. arXiv preprint arXiv, 2009, 0911.0460.

[31] Dietterich TG. Ensemble methods in machine learning, in: International workshop on multiple classifier systems. Springer, 2000: 1–15.

[32] Hoeting JA, Madigan D, Raftery AE, Volinsky CT. Bayesian model averaging: A tutorial. Statistical science, 1999, 14 (4): 382–401.