Research on Stock Price Prediction Model based on Clustering-Sliced LDA and Gaussian Process Regression

Kuan Jiang*

Navigation College, Dalian Maritime University, Dalian, China, 116026

*Corresponding author: agorgeous123@163.com

Abstract. Multi-factor prediction techniques based on machine learning models are crucial methods in the field of financial quantification. However, these methods often encounter challenges such as curse of dimensionality and unknown structural connections. To address these issues, this paper proposes an adaptive linear discriminant analysis based on clustering-sliced approach, which can be utilized for continuous response variables. It not only resolves the curse of dimensionality but also extracts local information of predictor variables regarding the response variables. For fitting the unknown structural connections between predictor variables and response variables, Gaussian process regression is employed due to the flexibility of kernel functions, enabling it to fit complex connection functions. Furthermore, Gaussian process regression can simultaneously consider factor information and stock price volatility information, and provide uncertainty interval estimation for predicted values. Simulation data analysis demonstrates that the proposed method exhibits smaller mean squared error, absolute error, and relative error compared to some classical quantitative stock price prediction models. Finally, this model is applied to the task of predicting stock price fluctuations, revealing its higher accuracy and robustness compared to the benchmark methods.

Keywords: Stock Price Prediction, High-Dimensional Data, K-means Clustering, Linear Discriminant Analysis, Gaussian Process Regression.

1. Introduction

The stock market's vast and complex datasets are characterized by significant volume and noise, presenting nonlinear and unstable attributes. With the evolution of computational power and big data analytics, an increasing cadre of financial institutions is turning to machine learning methodologies to predict stock price volatility. Predominantly, two principal machine learning paradigms are employed: time series forecasting models and multifactor prediction models. The former extrapolates based on the historical volatility inherent in stock price sequences. Should the stock prices manifest discernible patterns of volatility, the efficacy of these time series models is markedly enhanced. For instance, Li Xiaoning (2019) introduced a model predicated on daily Shanghai Stock Exchange data, which exhibited commendable predictive accuracy in practical deployments.[1] Wang Xiaohong, Wang Mengyao, and Hao Ting (2020) engineered a composite predictive framework amalgamating Autoregressive Integrated Moving Average (ARIMA) with Support Vector Machine (SVM).[2] Hu Yamei (2023) leveraged a GARCH model in conjunction with a suite of neural network architectures, including RNN, LSTM, and GRU, to forecast stock prices.[3] Nonetheless, the reliance solely on stock price data could engender a paucity of predictive insights, potentially undermining researchers' capacity to project future stock price trajectories. Consequently, this treatise delves into a multifactorial approach to stock price prognostication.

In multifactor prediction models, two key issues deserve particular attention: the curse of dimensionality and the unknown structural connections between predictor variables and response variables. Linear models based on regularization techniques can be used for prediction when there is a linear relationship between X and Y. For example, Li Bochao (2022) applied ridge regression to analyze financial indicators affecting the stock price of Luoyang Molybdenum Company, demonstrating the application value of ridge regression in analyzing stock price influencing factors.[4] Li, Liang, and Ma (2022) proposed and validated a new model called MS-MIDAS-LASSO, which combines multiple predictors. It has shown better performance than other models in predicting the realization of volatility in the S&P 500 under a variety of market conditions, including during the

COVID-19 pandemic.[5] However, the structural connections between predictor variables and response variables are often unknown, limiting the effectiveness of such models. Non-parametric statistical methods exhibit more apparent advantages when there is a nonlinear relationship between X and Y. These include kernel-based support vector machines, decision tree models, random forests, and neural networks. Madhu et al. (2021) compared the effect of support vector machine (SVM) and artificial neural network (ANN) in option price prediction, and the results showed that ANN performed better than SVM in option price prediction.[6] Subba Rao Polamuri (2019) attempted to track and predict stock market behavior using decision trees, random forest regressors, and extra tree regressors, showing effectiveness in predicting stock prices.[7] Elena Podasca (2021) investigated whether combining sentiment features extracted from financial news with historical price data could improve the predictive performance of the Swedish stock market index price trend.[8] These methods do not require explicit assumptions about the connection structure and can fit sufficiently complex connection functions. However, non-parametric statistical methods generally suffer from issues such as slow convergence with increasing dimensions, overfitting due to excessive parameters, and low computational efficiency.

In conclusion, this study opts for dimension reduction techniques to surmount the challenges posed by the curse of dimensionality. Specifically, a novel clustering-sliced linear discriminant dimension reduction approach is introduced. This technique, in contrast to conventional methods like Principal Component Analysis (PCA), boasts the merit of being a supervised reduction strategy, adept at preserving the local pertinent information of predictor variables with respect to response variables. The clustering-sliced method adeptly navigates interval partitioning and slice count determination issues, circumventing the loss of local information that typically accompanies the binary classification of stock price fluctuation variables. In terms of modeling the connection function, it is discerned that extant predictive models fall short in furnishing investors with a metric for future stock price risk-return. Conversely, the Gaussian Process Regression method stands out, amalgamating historical volatility and factor information to extrapolate future stock price trajectories with heightened precision and robustness. Empirical analyses underscore the proposed method's superior predictive accuracy and robustness compared to classical quantitative stock prediction models, demonstrating its efficacy in deducing future stock price volatility trends.

2. Theory and Methods

2.1. Clustering-Sliced Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a classic linear learning method. In the binary classification scenario, given a dataset, let, $D=\{(x_i, y_i)\}_{i=1}^{m}, y_i \in \{0, 1\}$. $X_i, \mu_i, \sum_i$ respectively denote the sets of examples for the $i \in \{0, 1\}$ Class, Mean Vector, Covariance Matrix. If we project the data onto line w. then the projections of the centers of the two classes on the line are $w^T\mu_0$and $w^T\mu_1$; If all sample points are projected onto the line, then the within-class scatter for the two classes is $w^T\sum_0 w$ and $w^T\sum_1 w$. Since the line is a one-dimensional space, all $w^T\mu_0, w^T\mu_1, w^T\sum_0 w$ and $w^T\sum_1 w$ are real numbers.

To ensure that the projected points of the same class are as close as possible, it is desirable to minimize the covariance of the projected points of the same class, aiming $w^T\sum_0 w + w^T\sum_1 w$ for minimizing it as much as possible. Conversely, to ensure that the projected points of different classes are as far apart as possible, it is desirable to maximize the distance between class centroids, aiming $||w^T\mu_0 - w^T\mu_1||_2^2$ for maximizing it as much as possible. By simultaneously considering both objectives, we arrive at the optimization goal we want to maximize.

$$J = \frac{||w^T\mu_0 - w^T\mu_1|}{w^T\sum_0 w + w^T\sum_1 w} = \frac{w^T(\mu_0-\mu_1)(\mu_0-\mu_1)^T w}{w^T(\sum_0+\sum_1)w} \tag{1}$$

Define the within-class scatter matrix $Sw = \sum_0 + \sum_1 = \sum_{x \in X_0} (x - \mu_0)\,(x - \mu_0)^{\mathrm{T}} + \sum_{x \in X_1} (x - \mu_1)\,(x - \mu_1)^{\mathrm{T}}$ and the between-class scatter matrix $S_b = (\mu_0 - \mu_1)(\mu_0 - \mu_1)^{\mathrm{T}}$. then equation (1) can be rewritten as:

$$J = \frac{w^{\mathrm{T}} S_b w}{w^T S_n w} \tag{2}$$

Noting that both the numerator and denominator consist of quadratic terms regarding w, the solution depends only on the direction of w, not its magnitude. Without loss of generality, let's assume $\mathrm{w}^{\mathrm{T}} \mathrm{S}_{\mathrm{w}} \mathrm{w} = 1$, then we have:

$$\begin{aligned} \min_{w} \quad & -w^{\mathrm{T}} S_b w \\ \mathrm{s.t.} \quad & w^{\mathrm{T}} S_w w = 1 \end{aligned} \tag{3}$$

Considering the stability of numerical solutions, it is common practice to perform singular value decomposition (SVD) on $\mathrm{S_w}$, that is $\mathrm{S_w} = \mathrm{U}\sum\mathrm{V}^{\mathrm{T}}$, where $\sum$ is a real diagonal matrix whose diagonal elements are the singular values of $\mathrm{S_w}$, and then obtain $\mathrm{S_w^{-1}}$ from $\mathrm{S_w^{-1}} = \mathrm{V}\sum^{-1}\mathrm{U}^{\mathrm{T}}$. Furthermore, we can extend LDA to multi-class tasks. Assuming there are N classes, and the number of examples in the ith class is $\mathrm{m_i}$, we first define the "total scatter matrix" as:

$$S_t = S_b + S_w = \sum_{i=1}^{m} (x_i - \mu)\,(x_i - \mu)^T \tag{4}$$

μ is the mean vector of all examples. We redefine the within-class scatter matrix $S_w$as the sum of scatter matrices for each class, i.e., $S_w = \sum_{i=1}^{N} S_{\mathrm{wi}}\,.S_{w_1} = \sum_{x \in X} (x - \mu_i)(x - \mu_i)^T.$ $S_b = S_t - S_w = \sum_{i=1}^{m} m_i (\mu_1 - \mu)(\mu_1 - \mu)^T$.Obviously, multi-class LDA can be implemented in various ways: using $S_b.S_w.S_{\mathrm{t}}$, or a combination of both. One common implementation is to optimize the objective:

$$\max_{w} \frac{tr(W^1 S_{\prime} W)}{tr(W^1 S_{\prime} W)} \tag{5}$$

Where $W \in R^{d \times (N-1)}, tr(\cdot)$ denotes the trace of a matrix. Equation (5) can be solved by the following generalized eigenvalue problem:

$$S_b W = \lambda S_{\mathrm{w}} W \tag{6}$$

The closed-form solution for W is a matrix composed of the eigenvectors corresponding to the d' largest non-zero generalized eigenvalues of $S_w^{-1} S_b$. In the aforementioned description, it can be observed that Y is a categorical variable. However, the stock price rise and fall studied in this paper are continuous variables. Therefore, we propose an improved method to transform the response variable into a discrete variable by slicing it. Specifically, we first use clustering to bin the response variable Y, obtaining category labels for the sliced Y, and use these labels as Y in LDA to perform linear discriminant analysis for multiple categorical variables. We call this clustering-sliced LDA (CLDA) method and define the predictor variables after dimension reduction by LDA as Z.

2.2. Gaussian Process Regression

Gaussian Process Regression (GPR) is a stochastic process, where each random variable, indexed by time or space, follows a multivariate normal distribution, meaning that any finite linear combination of these random variables is also normally distributed. The distribution of a Gaussian process is the joint probability distribution of all these (infinitely many) random variables, given by:

$$\mathrm{f} \sim \mathrm{GP}(\mu, \mathrm{k}) \tag{7}$$

Where $\mu(\mathrm{z})$ and $\mathrm{k}(\mathrm{z}.\mathrm{z}')$ 分 are the mean and covariance functions of the random variables, respectively. Thus, a Gaussian process is determined by its mean and covariance functions. In traditional regression models, we define $Y = f(\mathrm{z})$,as a deterministic function. In Gaussian Process

Regression, $f(z)$ is assumed to follow a Gaussian distribution. Typically, we assume the mean to be zero, i.e.

$$Y = f(z) \sim N(0, \sum) \tag{8}$$

Where $\sum$ is the covariance function $\sum_{ij} = K(z_i, z_j)$. By utilizing the kernel trick, we express the covariance function in terms of the kernel function K. For example, with the Radial Basis Function (RBF) kernel, $\sum_{ij} = \tau e^{\frac{-|z_i - z_i|^2}{\sigma^2}}$. With the polynomial kernel, $\sum_{ij} = \tau(1 + z_i z_j)^d$. Therefore, the covariance function $\sum$ can be decomposed as $\begin{pmatrix} K, K_* \\ K_*, K_{**} \end{pmatrix}$, the training kernel matrix, K, the training-test kernel matrix, K_*, and K_*^T, the test-training kernel matrix. The conditional probability distribution of the latent function f can be represented as:

$$f_* \cdot |(Y_1 = y_1, \cdots, Y_n = y_n, z_1, \cdots, z_n, z_i) \sim N(K_*^T K^{-1} y, K_{**} - K_*^T K^{-1} K_*.) \tag{9}$$

3. Data Analysis

3.1. Simulated Data Analysis

This study conducted a comprehensive analysis of simulated data using the Python programming language. The entire analysis pipeline was implemented, including the generation of simulated data using the `make_regression` function from the `sklearn` library. In the simulated data analysis:

- Experiment 1: Sample size was set to 200, with 20 features. This experiment was repeated 10 times.
- Experiment 2: Sample size was set to 150, with 20 features. This experiment was repeated 10 times.
- Experiment 3: Sample size was set to 200, with 15 features. This experiment was repeated 10 times.

To assess the performance of the model, calculated Mean Squared Error (MSE), Absolute Error (ABE), and Relative Error (REE) over 10 repetitions. The specific formulas for these metrics are as follows: Mean Squared Error (MSE): MSE is calculated as the mean of the squared differences between the true values and the predicted values. $\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\overline{y_i} - y_i)$. Absolute Error (ABE): ABE refers to the absolute difference between the measured value and the true value. $\mathrm{ABE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \overline{y_i})^2$. Relative Error (REE): REE is calculated as the ratio of the absolute error to the true value, multiplied by 100%.$\delta = \Delta/L \times 100\%$.

Employed LASSO[9][10], Ridge Regression[11], Elastic Net[12], and Bagging model[13] as comparative methods.

The results from ten repeated experiments are presented in Tables 1 through 9. The experimental results indicate that the proposed model demonstrates optimal performance. For instance, in scenarios where the sample size is 150 and the number of features is 20, the model achieves lower mean squared error values compared to four other comparative methods, specifically by 4.6818 , 3.7149, 6.4768, and 3.2234, respectively. The model also achieves lower absolute error values by 1.3293, 1.1733, 1.6517, and 1.0708, respectively, and lower relative error values by 0.0524, 0.0418, 0.0610, and 0.0329, respectively. The superior performance of the proposed method can be attributed to its initial step of performing dimension reduction on the data, which alleviates the curse of dimensionality and reduces collinearity among the data. Subsequently, it employs Gaussian Process Regression to capture some of the nonlinear structures between X and Y, a characteristic that other models lack. For example, both LASSO regression and ElasticNet are incapable of capturing nonlinear structures, while bagging models can capture them to a certain extent but fail to effectively reduce the dimensionality of the data, resulting in the presence of redundant data noise. In conclusion, the proposed method outperforms LASSO, Ridge Regression, ElasticNet, and Bagging_linear models that are based on ensemble learning.

Table 1: Mean squared error comparison results for sample size = 200, feature count = 20

Methods	MSE-mean	MSE-std
spline_LDA_Bagging	0.2396	0.1405
LASSO	4.5686	1.1034
Ridge	1.1902	0.3297
ElasticNet	3.5652	0.9824
bagging_linear	1.0028	0.2906

Table 2: Absolute error comparison results for sample size = 200, feature count = 20

Methods	ABE-mean	ABE-std
spline_LDA_Bagging	0.3831	0.1036
LASSO	1.7147	0.1991
Ridge	0.8688	0.1295
ElasticNet	1.5039	0.2230
bagging_linear	0.7975	0.1351

Table 3: Relative error comparison results for sample size = 200, feature count = 20

Methods	REE-mean	REE-std
spline_LDA_Bagging	0.0153	0.0095
LASSO	0.0460	0.0109
Ridge	0.0171	0.0060
ElasticNet	0.0295	0.0104
bagging_linear	0.0163	0.0074

Table 4: Mean squared error comparison results for sample size = 150, feature count = 20

Methods	MSE-mean	MSE-std
spline_LDA_Bagging	0.2804	0.1197
LASSO	4.9622	1.3418
Ridge	3.9953	0.7936
ElasticNet	6.7571	1.3383
bagging_linear	3.5038	0.6960

Table 5: Absolute error comparison results for sample size = 150, feature count = 20

Methods	ABE-mean	ABE-std
spline_LDA_Bagging	0.4194	0.0851
LASSO	1.7487	0.2389
Ridge	1.5928	0.1269
ElasticNet	2.0711	0.1643
bagging_linear	1.4903	0.1366

Table 6: Relative error comparison results for sample size = 150, feature count = 20

Methods	REE-mean	REE-std
spline_LDA_Bagging	0.0231	0.0123
LASSO	0.0754	0.0502
Ridge	0.0648	0.0466
ElasticNet	0.0841	0.0604
bagging_linear	0.0560	0.0420

Table 7: Mean squared error comparison results for $\text{sample size} = 200$, $\text{feature count} = 15$

Methods	MSE-mean	MSE-std
spline_LDA_Bagging	0.1350	0.0778
LASSO	5.4771	1.9943
Ridge	1.9835	0.6679
ElasticNet	5.9411	1.9926
bagging_linear	1.5216	0.5417

Table 8: Absolute error comparison results for $\text{sample size} = 200$, $\text{feature count} = 15$

Methods	ABE-mean	ABE-std
spline_LDA_Bagging	0.2785	0.0828
LASSO	1.7940	0.3750
Ridge	1.0807	0.1628
ElasticNet	1.8705	0.2808
bagging_linear	0.9470	0.1406

Table 9: Relative error comparison results for $\text{sample size} = 200$, $\text{feature count} = 15$

Methods	REE-mean	REE-std
spline_LDA_Bagging	0.0089	0.0059
LASSO	0.0453	0.0419
Ridge	0.0132	0.0049
ElasticNet	0.0228	0.0084
bagging_linear	0.0113	0.0033

3.2. Real Data Analysis

The data for this analysis was sourced from the tushare website (https://tushare.pro/). Daily stock price data was randomly selected for model testing. The specific training and testing period span from January 1, 2023, to December 29, 2023. The testing methodology employed a rolling window approach, where, for instance, the stock price and factor data for February, March, and April were used as the prediction base to forecast the price movement in May, and so forth. This process is illustrated in Figure 1.

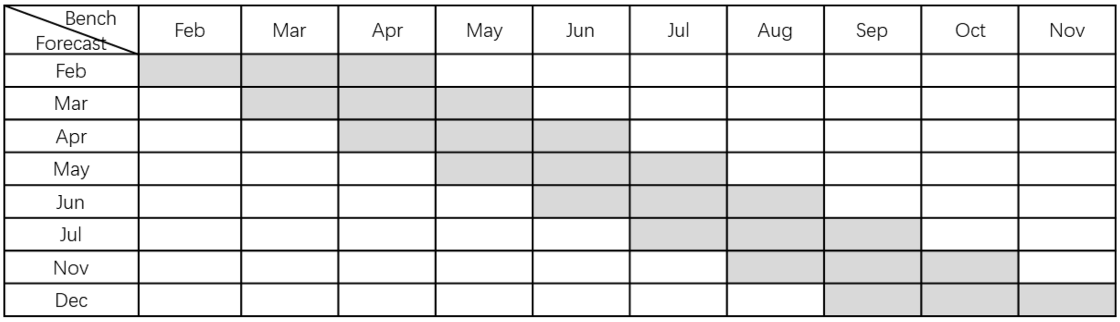

Figure 1: Rolling window training and testing approach

The experimental results presented in Tables 10-12 indicate that the proposed method achieves the lowest mean squared error, mean absolute error, and mean absolute percentage error, demonstrating its strong predictive ability and model generalization capability. This can be primarily attributed to the information compression ability of LDA and the non-parametric nature of Gaussian process regression, which enables the model to capture the inherent patterns in the data, particularly its advantages in handling complex data and simulating uncertainties. Additionally, analyzing the model's performance across different forecast periods reveals its good temporal stability. Even in long-term forecasts, the model's performance does not exhibit significant degradation, further validating its reliability and effectiveness in practical applications.

Table 10: Mean squared error comparison of forecast results for different months

Methods	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
spline_LDA_Bagging	0.3781	0.5177	0.3744	0.1753	0.4289	0.5590	0.4264	0.2425
LASSO	7.1602	7.1602	7.1602	7.1602	7.1602	7.1602	7.1602	7.1602
Ridge	1.1575	1.1575	1.1575	1.1575	1.1575	1.1575	1.1575	1.1575
ElasticNet	3.4688	3.4688	3.4688	3.4688	3.4688	3.4688	3.4688	3.4688
bagging_linear	1.0123	0.8443	1.0464	1.2184	1.1169	0.9882	0.8906	0.9246

Table 11: Absolute error comparison of forecast results for different months

Methods	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
spline_LDA_Bagging	0.4831	0.5714	0.4966	0.3269	0.5417	0.6101	0.5439	0.4108
LASSO	2.0546	2.0546	2.0546	2.0546	2.0546	2.0546	2.0546	2.0546
Ridge	0.8968	0.8968	0.8968	0.8968	0.8968	0.8968	0.8968	0.8968
ElasticNet	1.5525	1.5525	1.5525	1.5525	1.5525	1.5525	1.5525	1.5525
bagging_linear	0.8330	0.7633	0.8462	0.9194	0.8812	0.8372	0.7903	0.7936

Table 12: Relative error comparison of forecast results for different months

Methods	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
spline_LDA_Bagging	0.0096	0.0109	0.0124	0.0073	0.0108	0.0111	0.0103	0.0093
LASSO	0.0288	0.0288	0.0288	0.0288	0.0288	0.0288	0.0288	0.0288
Ridge	0.0097	0.0097	0.0097	0.0097	0.0097	0.0097	0.0097	0.0097
ElasticNet	0.0167	0.0167	0.0167	0.0167	0.0167	0.0167	0.0167	0.0167
bagging_linear	0.0094	0.0078	0.0091	0.0092	0.0101	0.0090	0.0086	0.0090

4. Conclusion and Perspectives

This paper proposes an innovative approach to tackle the challenges of dimensionality and unknown connection structure in multi-factor prediction models. It introduces a dimension reduction technique based on cluster slicing linear discriminant analysis (LDA) and employs Gaussian process regression (GPR) as the prediction model to fit the connection function. The primary innovations of this work lie in two aspects. Firstly, by utilizing clustering analysis for slicing, it not only overcomes the limitations of traditional equidistant and equi-frequency slicing but also allows for the determination of the number of slices through criteria such as the Akaike Information Criterion (AIC) or the Bayesian Information Criterion (BIC). Secondly, the application of Gaussian process regression enables the fitting of complex connection functions and provides a measure of uncertainty interval for quantitative trading strategies. The data analysis results demonstrate that, compared to classical benchmark models such as regularized linear regression and Bagging models based on ensemble learning, the proposed method exhibits smaller mean squared error, absolute error, and relative error. In summary, this method offers new insights and implementation approaches for stock price prediction.

For future research directions, several avenues can be explored. Firstly, it could be beneficial to explore more advanced dimension reduction techniques, such as sufficient dimension reduction methods, which preserve all relevant information about the response variable while reducing dimensionality. Secondly, the uncertainty measurement results from Gaussian process regression could be utilized in specific strategy formulation for quantitative investment, such as determining investment shares. Additionally, the selection of kernel functions in Gaussian process regression remains a challenge, and future work could consider employing a model averaging approach. For instance, by constructing Gaussian process regression with multiple kernel functions and using resampling methods to generate several sets of Gaussian process regression results, kernel function selection could be achieved based on Group Lasso regression.

References

[1] Li Xiaoning. Application of multivariate linear regression and time series models in stock prediction[J]. Science and Technology Entrepreneurship Monthly, 2019, 32(02): 153-155.

[2] Wang Xiaohong, Wang Mengyao, Hao Ting. Research on improved mixed prediction model of stock price with time-related sequence[J]. Technology Promotion and Development, 2020, 16(06): 672-678.

[3] Hu Yamei. Stock price prediction based on time series and neural network models[D]. Guangdong University of Finance, 2023.

[4] Li Bochao. Analysis of influencing factors of stock prices in non-ferrous metal industry based on ridge regression: a case study of Luoyang Molybdenum Industry[J]. China Business Review, 2022, 7(23): 110-112.

[5] Li, X., Liang, C., & Ma, F. (2022). Forecasting stock market volatility with a large number of predictors: New evidence from the MS-MIDAS-LASSO model. Annals of Operations Research, 18(1), 1-40.

[6] Madhu, B., Rahman, M. A., Mukherjee, A., et al. (2021). A comparative study of support vector machine and artificial neural network for option price prediction. Journal of Computer and Communications, 9(05), 78-91.

[7] Polamuri S R, Srinivas K, Mohan A K. Stock market prices prediction using random forest and extra tree regression[J]. Int. J. Recent Technol. Eng, 2019, 8(1): 1224-1228.

[8] Elena P. Predicting the movement direction of omxs30 stock index using xgboost and sentiment analysis[D]. Blekinge Institute of Technology, 2021.

[9] Liu Ying. Stock price prediction based on graph neural networks[D]. Southwest University of Finance and Economics, 2023.

[10] Yin Qiaoyi. Micro-analysis of Chinese industry stock prices based on Elastic Net method[D]. Anhui Normal University, 2018.

[11] Hu, W., & Zastawniak, T. (2020). Pricing high-dimensional American options by kernel ridge regression. Quantitative Finance, 20(5), 851-865.

[12] Sai, N., Smitha, K. G., & Lekha, A. C. (2023). Nifty stock prediction using Elasticnet and LSTM. IOSR Journal of Computer Engineering (IOSR-JCE), 15(4), 9-14.

[13] Niu Hongli, Zhao Yazhi. Predicting stock price index using Bagging algorithm and GRU model[J]. Computer Engineering and Applications, 2022, 58(12): 132-138.