# Comparative Analysis of ML models for Stock Price Prediction

Yaolong Xiang *

The King's School Canterbury, Kent, United Kingdom

* Corresponding Author Email: h18565101938@outlook.com

**Abstract.** Due to the unstable and non-stationary nature of financial market volatility, accurately predicting stock prices remains a challenging task. This study employs historical prices and technical indicators as features, comparing five machine learning models: linear regression, k-nearest neighbors, support vector regression, eXtreme Gradient Boosting, and random forests. The performance of these models is assessed based on mean squared error, $R^2$ score, and prediction accuracy within 1% and 2% thresholds. Experiment results show that eXtreme Gradient Boosting performed overall with the best accuracy under significant event impact and with a short-term dataset (approximately 2 years). The experiment also takes window rolling to make every model perform more stably. This study discusses the performance of the five models mentioned and uses the designed models to run through the same condition of the dataset to compare their differences. The limitations of the current work are taken into consideration, and directions for future research are identified.

**Keywords:** Stock Price Prediction, Machine Learning, Regression Model.

## 1. Introduction

Stock price predictions have long been a challenge for those who want to thrive in financial markets. Its unpredictability comes from multiple sources, including noise from the news, information gaps, and other external impacts. In addition, irrational investor behavior often leads to no logical decisions, causing the market to be mispriced and non-stationary. Significant events, such as the 2008 financial crisis and the 2020 COVID-19 pandemic, have further increased market volatility. Therefore, to be able to make precise predictions remains an extremely difficult task.

Despite the challenges, accurate and efficient stock predictions are important because they can improve investment decisions and risk management in practice. What is more, it can contribute to overall market efficiency by finding those mispriced assets [1]. For both individuals and groups of investors, even a small advantage in predicting stocks can make significant economic benefits or reduce losses. Thus, the development of more effective prediction methods remains both practically significant and academically relevant.

Traditionally, approaching stock prediction often includes both fundamental analysis and technical analysis. Fundamental analysis evaluates a company based on financial statements, industry conditions, and economic outlook [2]. Although it is logical, its corrections in the market are often slow, which makes it unavailable for short-term investment. Technical analysis relies on historical prices and volumes to identify trends. However, it is highly noise-sensitive and often provides only descriptive patterns without a clear trading signal, for example, the exact threshold for identifying the pattern is likely to be downwards or upwards.

To be able to solve these problems, more often researchers have turned to machine learning (ML). Unlike traditional methods, ML can combine different data sources, including fundamental ratios, historical prices, and even alternative data like news, to make predictions. However, the performance of ML models is not uniform. Linear models often have difficulties capturing the diversity of financial data, while more complicated ensembles and instance-based approaches introduce different biases of induction, making the comparative performance of these models remain an open question [3]. Also, many existing comparative studies suffer from methodological limitations. They often overlook rigorous hyperparameter optimization, sector-specific feature engineering, and practical evaluation metrics.

Therefore, this study conducts a meticulous comparative analysis of five ML models. Based on previous work, this study uses feature engineering, including relative price normalization, and builds a framework using automated hyperparameter optimization and cross-validation. Furthermore, it overcomes the limitations of traditional accuracy metrics by introducing a dual accuracy threshold mechanism, thus offering a more practical model performance evaluation approach for real-world trading.

Overall, this study offers some crucial contributions in the area of stock price prediction:

Comprehensive Model Comparison: This study evaluates five regression models, including LR, KNN, RF, SVR, and XGBoost, in a unified feature set and time-series cross-validation framework to assure fairness and reproducibility.

Feature Engineering: A feature set is built, containing lagged prices, rolling averages, and short-term volatility indicators. Relative price normalization is adopted to enhance comparability across different stocks, extending the framework of Cao et al. [4].

Rolling-Window Evaluation: A rolling window is designed to assess model performance based on different sub-periods rather than a single static train–test split, thereby improving model stability and fairness.

Expanded Performance Metrics: Beyond conventional measures like MSE, the study evaluates models using Mean Absolute Error (MAE), Mean Scaled Absolute Error (MSAE), Directional Accuracy (1% and 2% thresholds), R-square, and so on for the error metrics. Under multi-dimensional evaluation, it includes not only predictive accuracy but also trading-oriented performance.

Automated Hyperparameter Optimization: Model robustness is improved through automated hyperparameter tuning with GridSearchCV, in line with best practices in machine learning [5].

Trading-Oriented Visualization: Predicted results are visualized with gold–silver–bronze performance highlights and clear model separation between linear and non-linear, providing straightforward visualization into comparative strengths.

Empirical Validation: The analysis is conducted on real-world equity data obtained via AkShare, ensuring that findings are directly relevant to practical financial forecasting contexts.

## 2. Related Work

The field of financial forecasting has evolved from interpretable linear models to increasingly complex machine learning algorithms. Linear Regression provides a transparent and widely used benchmark [6], but its inability to capture nonlinear dynamics is a critical limitation. To overcome this, nonparametric methods were introduced. Random Forest demonstrated robustness in capturing nonlinear feature interactions, while K-Nearest Neighbors offered a simple framework for identifying local market structures [6, 7].

More advanced approaches have since been developed. Support Vector Regression leverages kernel methods to capture nonlinear relationships while maintaining robustness. Similarly, XGBoost extends the gradient to improve the framework with regularization and efficient optimization, and it has been widely adopted for financial applications due to its strong empirical performance [8, 9].

So forth, recent studies emphasize the critical role of feature engineering and evaluation methodology. Cao et al. highlighted the benefits of normalization techniques and pointed out sectoral characteristics, while Patel et al. demonstrated improvements from integrating technical and macroeconomic indicators. Lopez de Prado further advises the adoption of evaluation metrics with direct economic relevance, such as directional accuracy and risk-adjusted measures, rather than relying on statistical error terms only [10, 11, 12].

Beyond these advances, existing studies often assess models in isolation, apply limited hyperparameter tuning, or rely on static train–test splits. This creates a gap: a lack of controlled comparative analyses that merge modern preprocessing, rolling-window evaluation, automated optimization, and practical performance metrics. The present study addresses this gap by conducting a direct empirical comparison of five algorithms—LR, KNN, RF, SVR, and XGBoost—within a

consistent pipeline that incorporates sector-specific feature design, automated hyperparameter optimization, and multi-dimensional evaluation metrics.

## 3. Methodology

### 3.1. Data Collection

This research utilizes historical daily price and volume data for two distinct Chinese financial assets, selected to represent different risk-return profiles and market dynamics. One is sz.000001 (Ping An Bank) as a leading cyclical banking stock, representing the financial sector. The other is SH.600519 (Kweichow Moutai) as a leading defensive consumer staple stock, representing the high-end liquor and spirits sector. Data for each asset is programmatically retrieved using the akshare Python API, a reliable source for historical Chinese market data. The primary data fields collected for each trading day include opening price, high price, low price, closing price, and trading value. To ensure a robust out-of-sample test and simulate a realistic trading scenario, the dataset for each asset is split into a train set and a test set based on time. The programs are trained on the initial 80% of the chronological data and evaluated on the most recent 20%, strictly preserving the temporal order to prevent look-ahead bias.

A critical preprocessing step is applied during experimentation to mitigate the effects of absolute price levels and focus the models on learning relative price movements. All price-based features (open, high, low, close) are normalized relative to the first day closing price in the dataset using the formula:

$$\text{df[price\_cols]} = \frac{df[price\_cols]}{base\_price} \tag{1}$$

where $base\_price$ is the closing price on the first day of the respective asset data series. This transformation creates a stationary starting point for all models while preserving the intra-day and inter-day dynamics essential for prediction.

### 3.2. Feature Engineering

To capture historical price patterns, several feature sets are constructed based on fundamental financial theories. Lag features are created on the core theories of autocorrelation, which suggest that the most powerful predictive factor is the previous value. According to the Random Walk Hypothesis, the price of today deeply depends on the price of the day before or earlier. However, the markets are not entirely valid and are often varied by noise, which can lead to momentum and reversal effects. Lag features enhance the basic informational data to catch these short-term effects. For instance, if a model assigns a weight of the data in lag_1 close to 1, while the weights of other lag data are near 0, it effectively approximates a simple random walk model. This serves as a powerful and standard benchmark, providing the most straightforward memories for models to catch short-term effects and the sources of dependency on the data.

To reduce noise interference and identify underlying trends, this study further incorporates moving average features, including 5-day and 10-day moving averages. The calculation of the average price is the sum of the close prices in the wanted days divided by the number of days. When the price is below the average, it often indicates a downward trend, whereas a price above the average may indicate an upward trend. This transformation converts absolute price levels into relative price levels, serving as a gauge of trend strength, that is, the likelihood of prices rising or falling.

Volatility characteristics are also incorporated into feature engineering. By calculating a standardized metric of the percentage change in daily closing prices, the amount of price change within a specific time window can be effectively measured. This indicator captures the phenomenon of volatility clustering, where significant price swings are always recurring. It reflects market uncertainty, the degree of chaos, and trading risks. Eventually, high-volatility markets and low-volatility markets exhibit distinctly different characteristics. Models can use the information to assist

in forecasting—adopting a more conservative approach during high-volatility periods and a more aggressive one when volatility is low.

This study sets the prediction target as the closing price of the following day, aiming to enhance the robustness and interpretability of price forecasting.

### 3.3. Models

LR models treat the target variable as a linear mix of the input features, employing the Ordinary Least Squares method to minimize the error sum of squares [6]. It is selected for its strong performance as a baseline model in financial forecasting and its high interpretability, as its coefficients directly reveal insight into the relationship between each feature and the predicted outcome.

KNN is a nonparametric, instance-based learning algorithm. It operates on the principle that similar trends will eventually produce similar results that have happened before. For a given data point, the model forecasts the objective value by taking the average of the outcomes of the k most similar historical observations in the feature space, with similarity measured by a distance metric such as Euclidean distance [6]. This model is included for its capability to capture complicated, non-linear local patterns without assuming an underlying functional form of the data.

RF is an integrated learning method that involves building multiple decision trees during training. This method selects each decision tree from a guided sample of the data and trains it using a random subset of features, thereby introducing randomness. Finally, the forecasts are averaged across all individual trees to form the final prediction [7]. This approach significantly reduces the variance and overfitting issues commonly associated with single decision trees. The algorithm is chosen for its robust capability to capture high-dimensional non-linear relationships and interactions between features.

SVR employs the principles of Support Vector Machines to handle regression tasks. SVR attempts to find a function whose bias from the real objective values does not exceed the specified margin, while ensuring that the function is as flat as possible [8]. By using kernel functions, SVR can trap nonlinear correlations within the data without increasing the dimensionality of the feature space. This model is included for its capacity to handle complex, non-linear patterns and its robustness, making it a strong candidate for financial time series forecasting.

XGBoost is an ensemble learning method based on gradient boosting of decision trees. It progressively constructs models by having each new tree correct the errors of the previously integrated by minimizing a specified loss function [9]. XGBoost incorporates regularization terms to manage model complexity and resist overfitting. The algorithm is selected for its strong performance in predictive modeling competitions and its ability to capture complex, high-dimensional nonlinear relationships.

### 3.4. Evaluation Metrics

To fully assess the forecasting models, both error-based and accuracy-based metrics are employed. These primarily include the following:

Mean Squared Error (MSE) quantifies the mean range of the prediction errors by calculating the average of the squared differences between the predicted values $\hat{y}_i$ and the actual values $y_i$. Squaring the errors amplifies greater errors, making the metric especially responsive to outliers. A smaller MSE indicates superior model performance [13]. The formula for mathematics is as follows:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2}$$

Root Mean Squared Error (RMSE) is the square root of MSE, sharing the same unit as the objective variable, making it more interpretable:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3}$$

Mean Absolute Error (MAE) weighs the mean absolute value of errors without regard to their direction. It is more robust to outliers compared to MSE [14]:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{4}$$

Mean Absolute Percentage Error (MAPE) evaluates errors in percentage terms, providing scale-independent interpretability [15]:

$$MAPE = \frac{100}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{5}$$

The $R^2$ score gives the percentage of the variance in the dependent variable that can be forecasted by the independent variables. It offers a scale-free measure of goodness-of-fit. An $R^2$ value of 1 means a good fit, whereas a value of 0 suggests the model performs no better than simply predicting the average of the target variable. Higher values are preferred. The calculation formula is shown below [5]:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \tag{6}$$

where $\bar{y}_i$ is the mean of the actual values.

To assess the models from a trading perspective, two pragmatic accuracy metrics are introduced. Accuracy1% and Accuracy2% calculate the percentage of predictions where the relative absolute error falls within a 1% or 2% margin of the actual price, respectively. Predictions within 2% of the actual move are often accurate enough to be profitable in a trading strategy after accounting for transaction costs, making these metrics a direct indicator of practical utility. The formula for Accuracy X% is as follows [2]:

$$\text{AccuracyX\%} = \frac{100}{n}\sum_{i=1}^{n} 1\left(\frac{|y_i - \hat{y}_i|}{y_i} < \frac{X}{100}\right) \tag{7}$$

where $\sum_{i=1}^{n} 1$ is the directive function, which responds with 1 when the condition is correct and 0 otherwise.

Directional Accuracy (DA) measures the percentage of correct predictions in terms of direction (up or down), which is particularly important in finance:

$$DA = \frac{1}{n}\sum_{i=1}^{n} 1(sign(y_i - y_{i-1}) = sign(\hat{y}_i - \hat{y}_{i-1})) \tag{8}$$

where $1(\cdot)$: indicator function. This metric focuses on capturing market trends rather than exact price levels.

### 3.5. Experiment Setup

Daily stock price data from the Chinese market for the period 2019–2020 was used. The Shenzhen Component Index (000001) served as the primary subject. To provide a stability benchmark, individual stocks with relatively low volatility, such as Kweichow Moutai (600519), were included as control cases. All prices were adjusted for splits and normalized relative to the initial trading day to facilitate comparison.

The predictive features included lagged closing prices over the previous 1–5 days, 5-day and 10-day moving averages of the closing price, and 5-day rolling volatility based on daily returns. The target variable was defined as the closing price of the next day. Missing values resulting from lagging and rolling computations were removed.

To simulate a realistic forecasting scenario and prevent future data leakage, a rolling window approach was applied using the scheme detailed in Table 1. The window size is set for 60 days to test the stability. Historical data within the window was used to train the model, after which the model

predicted the closing price of the final day in the window. Features were standardized within each window to ensure consistent scaling across variables. This procedure was repeated by shifting the window forward by one trading day at a time, generating a sequence of out-of-sample predictions across the entire period.

**Table1**. Preventing Data Leakage

| Fold | Training Period | Validation Period |
|---|---|---|
| Fold1 | Train 0 → t1 | Validation t1 → t2 |
| Fold2 | Train 0 → t2 | Validation t2 → t3 |
| Fold3 | Train 0 → t3 | Validation t3 → t4 |
| Fold4 | Train 0 → t4 | Validation t4 → t5 |
| Fold5 | Train 0 → t5 | Validation t5 → t6 |

Model hyperparameters are optimized using Grid Search with the time series cross-validation strategy described above. This exhaustive search method evaluates all possible combinations of parameters within predefined ranges to identify the parameter set that minimizes the forecast error on the validation folds. The specific parameter grids tested for each model are described in Table 2.

**Table 2**. Hyperparameters of GridSearchCV.

| Models | Hyperparameters |
|---|---|
| Linear | “Fit_intercept” [True, False]<br>“positive” [True, False] |
| KNN | “n_netighbours” [50, 100]<br>“weights” [‘uniform’, ‘distance’] |
| Randomforest | “n_estimators” [200, 300]<br>“max_depth” [5, 10] |
| SVR | “C” [0.1, 1, 10]<br>‘epsilon’ [0.01, 0.1]<br>“kernel” [‘rbf’, ‘linear’] |
| XGboost | “n_estimators” [200, 300]<br>“max_depth” [3, 5]<br>“learning_rate” [0.05, 0.1] |

## 4. Experiment Results

The experiment data is taken from 2019.01.01 to 2020.12.31. This timeframe covers a range of market conditions, including relatively stable periods as well as heightened volatility associated with major events.

### 4.1. Model Prediction Performance Comparison

The predictive capabilities are assessed using a 60-day rolling window approach, with results summarized in Fig. 1 and Fig. 2. Across all error metrics, XGBoost consistently delivers the best performance among the five models. SVR, RF, and LG also demonstrate strong predictive accuracy, though they slightly trail XGBoost. In contrast, KNN performs significantly worse than the other models.


**Fig. 1** Model prediction performance for stock SZ.000001

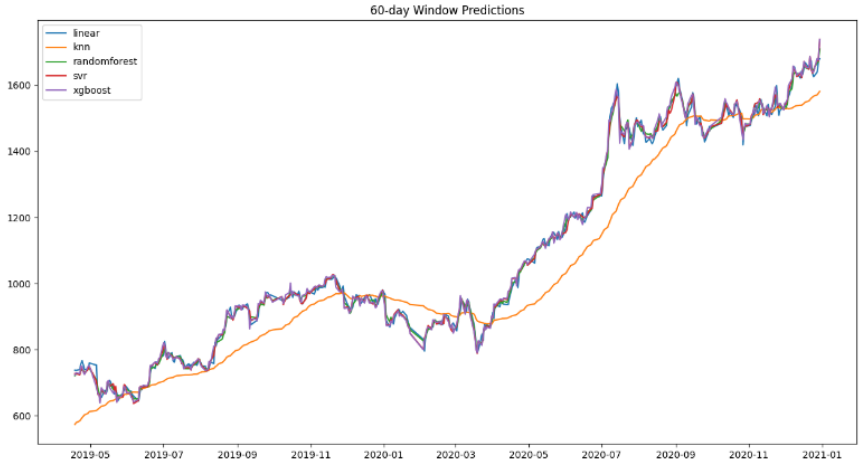


**Fig. 2** Model prediction performance for stocks sz.600519

### 4.2. Comparison of Different Metrics

From Fig. 3 to Fig. 6, the prediction performance under error metrics such as MSE, MAE, and R-square in stocks sz.000001 and sz.600519.

This pattern holds under a short-term dataset setting, where XGBoost continues to outperform all other models, reaffirming its robustness in time-series forecasting tasks. A detailed analysis of model capabilities based on the evaluation metrics reveals the following insights:

XGBoost exhibits best-in-class forecasting precision, achieving the lowest Mean Squared Error (MSE) of 0.0035 and Mean Absolute Error (MAE) of 1.7991%. It also ranks among the top models in directional accuracy metrics (Acc@1% and Acc@2%), demonstrating a strong ability to correctly predict price movement directions. Overall, XGBoost is the top-performing model, excelling in both numerical precision and directional forecasting.

SVR also delivers excellent performance, with MSE and MAE values of 0.0062 and 1.9209%, respectively, placing it just behind XGBoost in forecasting precision. Its directional accuracy is competitive, grouping it in the top-tier models. SVR is a powerful and reliable approach for this forecasting task.

RF shows good predictive capability, though its accuracy is lower than XGBoost and SVR, with an MSE of 0.0081 and an MAE of 2.0301%. It excels in directional prediction, consistently ranking among the top performers. RF can be considered an effective tool for identifying general patterns and directional trends, though it is not the most accurate for point forecasting.

LR displays relatively poor performance, with significantly higher MSE (0.0136) and MAE (2.4475%) values. It also fails to top tiers in accuracy, indicating its limitations in capturing

meaningful nonlinear patterns within financial data. As a result, the LR is deemed inadequate for such a complex forecasting task.

KNN fails to produce valid results, exhibiting abnormally high MSE (0.0273) and MAE (6.7613%) values. It also shows it performs poorly in directional forecasting. These results indicate that KNN is unsuitable for financial time-series prediction due to its inability to handle noise and complex time dependencies.

Based on comprehensive experimental evaluation, several key observations can be drawn regarding model performance: (1) Non-linear models demonstrate consistent superiority over linear approaches in capturing complex financial time series patterns; (2) XGBoost emerges as the optimal model by achieving the best balance between numerical precision and directional accuracy; (3) forecasting magnitude presents greater challenges than predicting direction, with XGBoost showing particular strength in numerical precision; (4) simpler models exhibit significant limitations in handling the complexity of financial forecasting tasks. These findings are further reinforced by the consistent performance hierarchy observed across evaluation metrics.

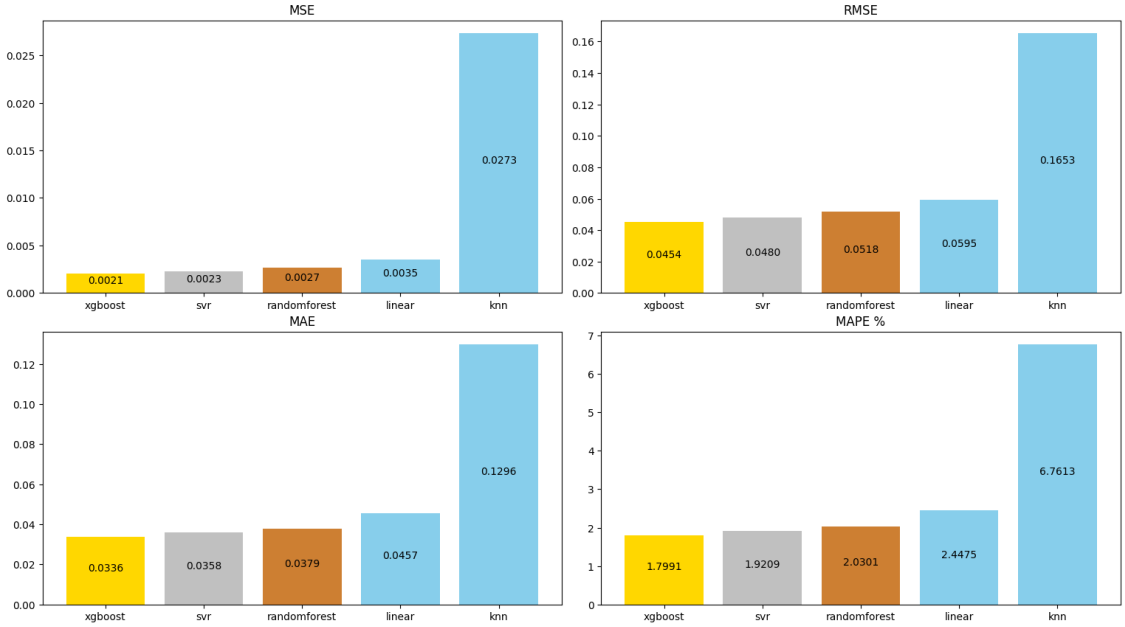


**Fig. 3** Comparison of Different Indicators for stocks sz.000001

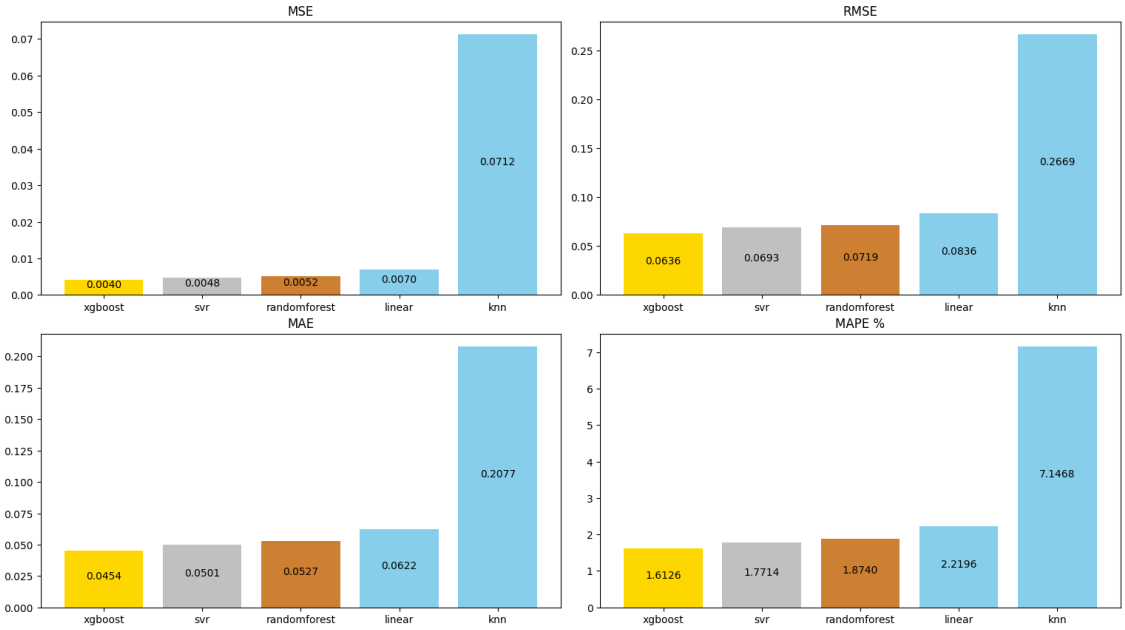


**Fig. 4** Comparison of Different Indicators for stocks sz.600519

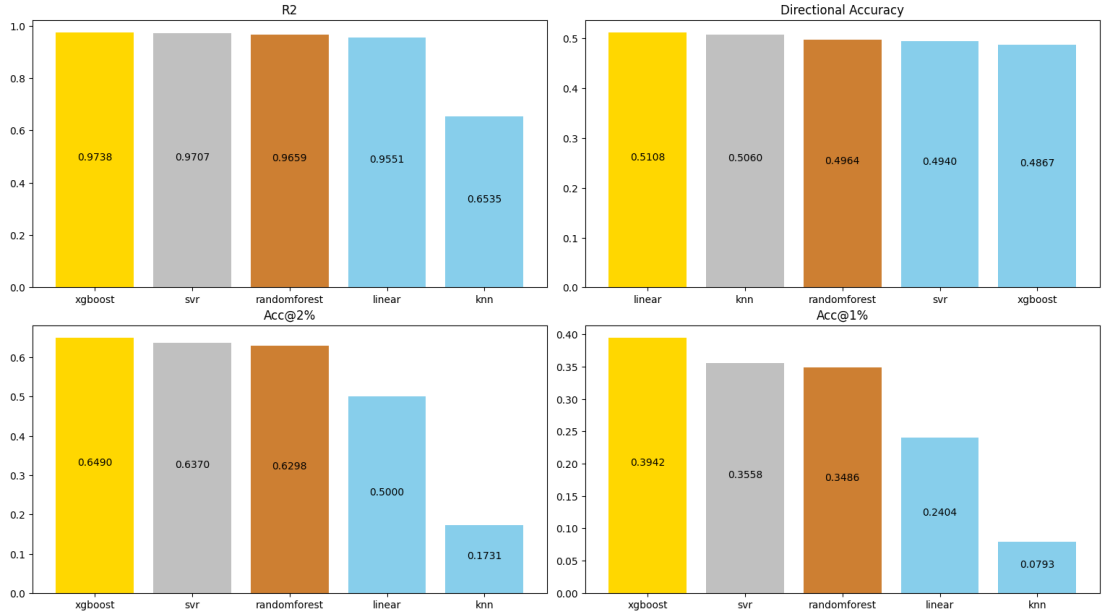


**Fig. 5** Comparison of Different Indicators for stocks sz.000001

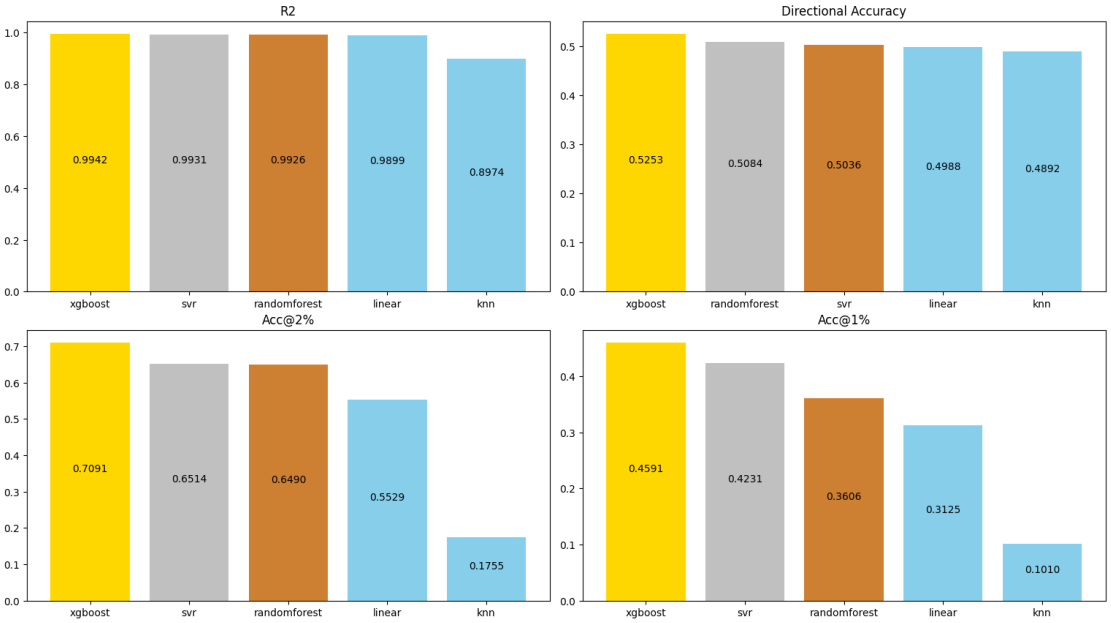


**Fig. 6** Comparison of Different Indicators for stocks sz.600519

# 5. Discussion

## 5.1. Superior Performance of XGBoost

A comprehensive empirical analysis across multiple assets and rolling window sizes reveals that XGBoost consistently outperforms the other models (LR, KNN, RF, and SVR) in next-day price prediction, particularly for longer-term windows. While this may initially seem counterintuitive given the apparent simplicity of linear relationships in short-term stock prices, the superior performance of XGBoost can be explained through several interconnected factors:

First, XGBoost possesses the capacity to capture nonlinear and complex dependencies. Financial time series, though partially linear in the very short term, often contain subtle nonlinear patterns due to market microstructure, volatility clustering, and delayed responses to information. XGBoost, as a gradient-boosted tree ensemble, can adaptively identify and exploit these patterns without prior specification. In contrast, LR is restricted to linear combinations of past prices and cannot leverage

these higher-order relationships, while KNN and RF may either underfit or overfit depending on the window size.

Second, XGBoost enhances robustness to noise through ensemble learning. Daily stock price data are notoriously noisy. XGBoost's gradient boosting approach sequentially reduces residual errors while incorporating regularization (tree depth, learning rate), which allows it to focus on genuine signals rather than idiosyncratic noise. RF and KNN, although powerful, lack this gradient-guided refinement and are more prone to overfitting in rolling window predictions, particularly in volatile periods.

Finally, XGBoost effectively utilizes lagged features and rolling statistics. The engineered features (lags, moving averages, volatility) provide both linear and weakly nonlinear information about past price behavior. XGBoost efficiently identifies which features contribute most to the prediction, weighing them through its tree-based structure. LR can only make fixed linear weights, and simpler models cannot capture communication between features, limiting their ability to predict.

The success of XGBoost shows that, for short-term forecasting, leveraging ensemble tree methods with gradient boosting can provide both flexibility and robustness. The model effectively balances capturing nonlinear structures. Prevent overfitting to noise, yielding superior predictive performance across all window sizes.

This finding highlights the importance of model selection based on data characteristics rather than assuming simpler models will always work as well as non-linear model does. While simplism is valuable, modern ensemble methods like XGBoost can exploit subtle signals that linear models cannot, making them highly suitable for short-term stock price forecasting when robust performance is critical.

### 5.2. Limitations of the Current Approach

Current approaches reveal some key limitations. Their feature relies on technical indicators, ignoring potentially valuable fundamental factors such as earnings reports, macroeconomics, and sentiment data from news or social media. Secondly, these approaches do not employ time-series-specific models, which may be better suited to capture temporal dependencies. Furthermore, the model also employs fixed hyperparameters, though the market conditions constantly change. Lastly, the evaluation framework excludes risk-adjusted performance metrics like the Sharpe ratio to evaluate actual trading value.

### 5.3. Future Work

Future work will advance along several key aspects to enhance the predictive performance. By breaking conventional frameworks, customized forecasting models will be constructed through stock classification (by industry or volatility characteristics), with feature and parameter optimization tailored to each category to achieve model specialization. Cross-asset learning mechanisms will be introduced through a transfer learning framework to capture inter-stock dependencies and cross-industry spillover effects.

## 6. Conclusion

This research conducts a comparative analysis of the five regression algorithms— LR, KNN, RF, SVR, and XGBoost—for stock price prediction. The results indicate that LR, while serving as a transparent and interpretable baseline, underperformed significantly compared to non-linear models in this dataset. Its role here is not competitive in accuracy, but important as a benchmark for evaluating more advanced methods. Among the non-linear models, RF shows moderate predictive capability but limited generalization, while KNN fails to handle noise and time dependencies, making it unsuitable for financial prediction. SVR improves in handling non-linear dynamics, while XGBoost shows strong predictive power through its regularization and ensemble boosting framework, though at the cost of reduced interpretability.

Overall, while advanced non-linear methods substantially improve predictive accuracy, linear regression retains value only as a transparent and interpretable benchmark, rather than a viable forecasting tool.

## References


[1] Bodie, Z., Kane, A., Marcus, A. J. Investments (12th ed.). McGraw-Hill Education, 2021.

[2] Graham, B., Dodd, D. L. Security analysis (6th ed.). McGraw-Hill, 2009.

[3] Fischer, T., Krauss, C. Deep learning with long short-term memory networks for financial market predictions. European Journal of Operational Research, 2018, 270 (2), 654–669.

[4] Cao, Y., He, H., Wang, J. Feature-based forecast model performance prediction. Information Sciences, 2021, 576, 1–13.

[5] James, G., Witten, D., Hastie, T., Tibshirani, R. An introduction to statistical learning. Springer, 2013.

[6] Altman, N. S. An introduction to kernel and nearest-neighbor nonparametric regression. The American Statistician, 1992, 46 (3), 175–185.

[7] Breiman, L. Random forests. Machine Learning, 2001, 45 (1), 5–32.

[8] Drucker, H., Burges, C. J., Kaufman, L., Smola, A., Vapnik, V. Support vector regression machines. Advances in Neural Information Processing Systems, 1997, 9, 155–161.

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

[10] Hastie, T., Tibshirani, R., Friedman, J. H. The elements of statistical learning: Data mining, inference, and prediction (2nd ed.). Springer, 2009.

[11] Patel, S., Shah, D., Patel, S. Forecasting stock market trends using machine learning algorithms. Procedia Computer Science, 2015, 50, 106–111.

[12] Lopez de Prado, M. Advances in financial machine learning. John Wiley & Sons, 2018.

[13] Hyndman, R. J., Koehler, A. B. Another look at measures of forecast accuracy. International Journal of Forecasting, 2006, 22 (4), 679–688.

[14] Willmott, C. J., Matsuura, K. Advantages of the mean absolute error. Climate Research, 2005, 30 (1), 79–82.

[15] De Myttenaere, A., Golden, B., Le Grand, B., & Rossi, F. Mean absolute percentage error for regression models. Neurocomputing, 2016, 192, 38–48.