November 5, 2025

1 Introduction

1.1 Data Description

Selection of all data used in this report was sourced from the Wind database. [Wind Information Co., Ltd., 2024] The stock sample consists of the constituent stocks of the A50 Index as of the data acquisition date. The time period spans from September 1, 2020, to September 26, 2025, covering 1,232 trading days. The data is of daily frequency and includes dimensions such as price, volume, and valuation metrics.

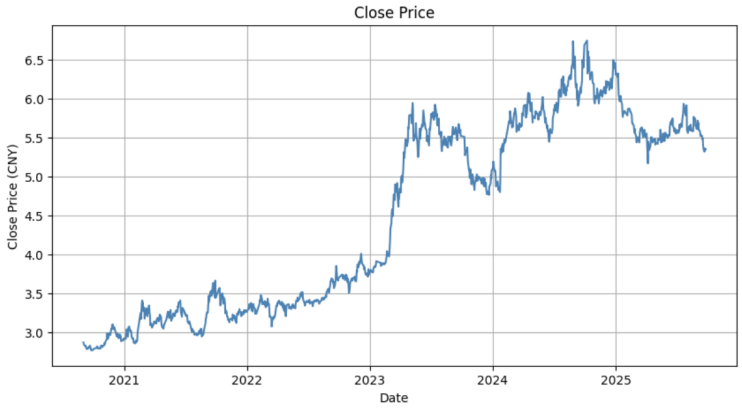
To clean and preprocess the data, we implement following method:

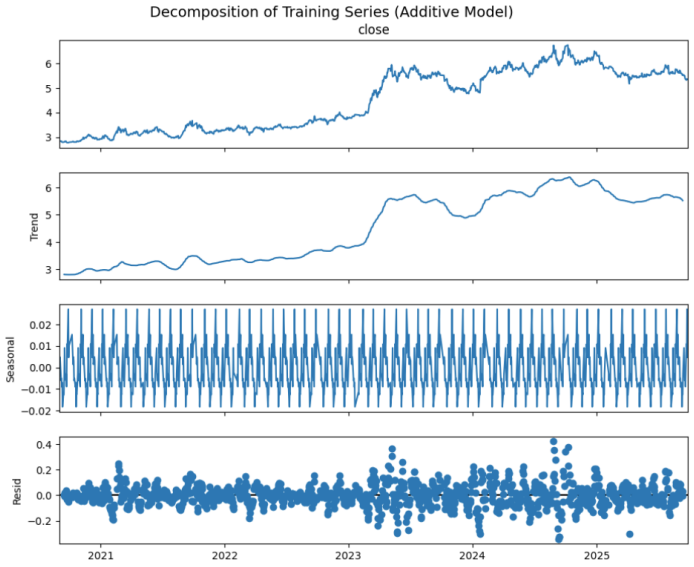
- **Missing Value Imputation**
  All missing data points were filled using the forward-fill method, where the last known observation is carried forward.

- **Data Type and Indexing**
  The date column was converted to a proper `datetime` format and set as the primary index for the time series. All price and volume columns were confirmed to be of a numeric data type.

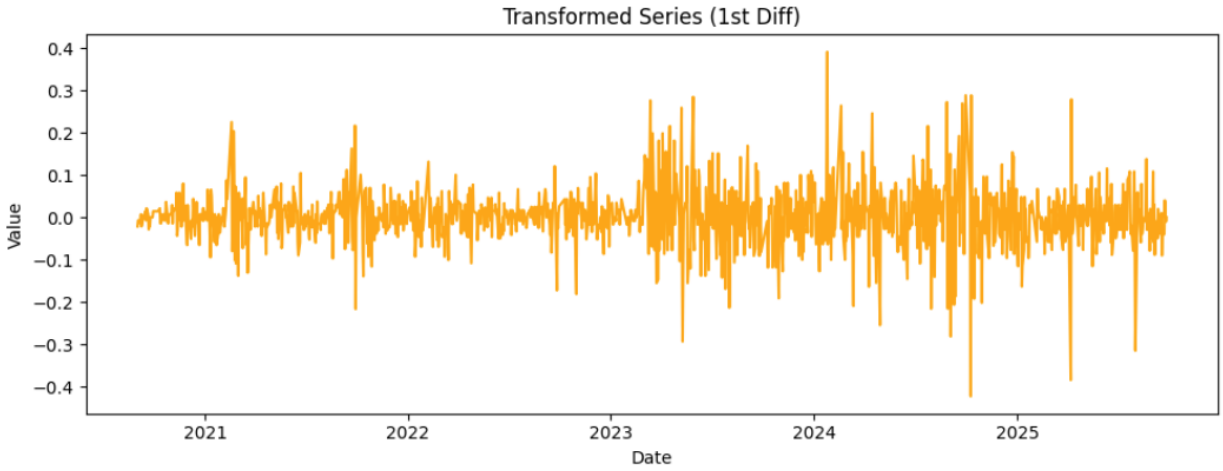
1.2 Data Display

We conduct our analysis on the daily close prices (from 2020/9/1 to 2025/9/25) of a single stock, **600028.SH**.





We perform the first-differencing transformations to achieve stationarity. We split the data into 80% training data and 20% testing data.



2 ARIMA Methodology

2.1 Model Overview

The Seasonal Autoregressive Integrated Moving Average (SARIMA) model is an extension of the ARIMA model that captures both non-seasonal and seasonal patterns in a time series. It is denoted as:

$$\text{SARIMA}(p, d, q) \times (P, D, Q, s)$$

The general SARIMA equation is:

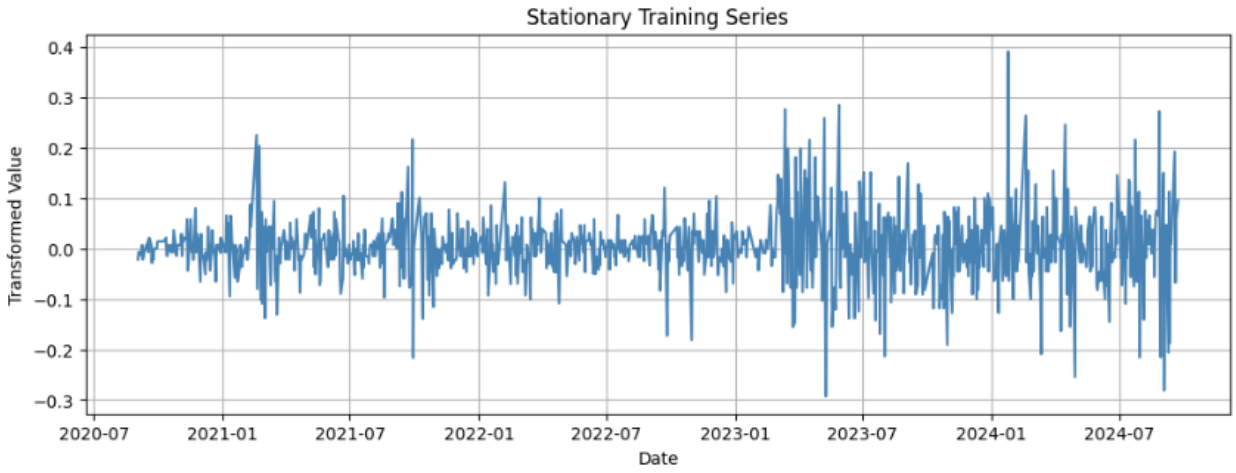
$$(1 - \phi_1 B)(1 - \Phi_1 B^s)(1 - B)(1 - B^s)y_t = (1 + \theta_1 B)(1 + \Theta_1 B^s)\varepsilon_t$$

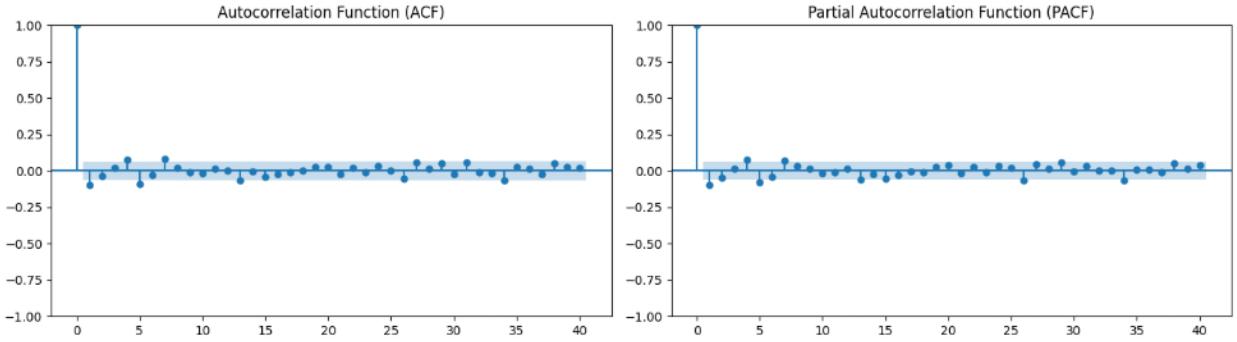
2.2 Stationarity Check

The stationarity tests (ADF test) were used to determine the order of differencing, d and D. The KPSS test rejects stationarity in levels but not after log-differencing, which is consistent with the ADF findings. Our time series passed the ADF and KPSS test after 1st differencing.

```
ADF Statistic: -16.96129005938971
p-value: 9.304247995922652e-30
Critical Values: {'1%': -3.435690695421723, '5%': -2.863898416697677, '10%': -2.5680256555204184}
Reject H0: Series is stationary

KPSS Statistic: 0.11956658074387824
p-value: 0.1
Critical Values: {'10%': 0.347, '5%': 0.463, '2.5%': 0.574, '1%': 0.739}
Fail to reject H0: Series is stationary
```





2.3 Parameter Estimation and Experiment Design

Autocorrelation (ACF) and Partial Autocorrelation (PACF) plots guided the choice of p, q, P, and Q. AICc, AIC and BIC were employed to compare competing models and select the best combination. These are conducted using the `statsmodels` library in Python.

We explore the Non-seasonal ARIMA model on the differenced series, where $p, q \in \{0, 1, 2\}$ and $d = 0$ (since differencing was performed explicitly), and the Seasonal SARIMA with trading-week seasonality $s = 5$: $P, Q \in \{0, 1, 2\}, D \in \{0, 1\}$. Because differencing is handled in preprocessing, the ARMA layer works with $d = 0$.

We also manually choose some models based on the ACF/PACF plot: Both ACF and PACF decay gradually (no clear cutoff or seasonal appearance), so we consider ARMA models with $p, q \neq 0$. To avoid overfitting, we choose p, q to be values less than 3. Therefore, we have candidates drawn from ACF, PACF: ARMA(1,1) ARMA(2,2) ARMA(1,2) ARMA(2,1)

We set several criteria to examine the performance of our models, including AIC, BIC and AICc to test the primacy, and RMSE, MAE, and R^2 (with the same weight distribution as before), from

1-step-ahead forecasts rolled through the test period to assess the general performance.
We fit the candidate on the training window, then produce rolling one-step-ahead forecasts across the full test window (expanding origin), collecting point forecasts and 95% confidence intervals.
and 95% confidence intervals.

2.4 Results

2.4.1 Model Fit Summaries

The results are shown below, where the ARIMA(1,0,0) is the baseline for the stationary series, and the SARIMA(0,0,0)×(0,0,0,5) is the best seasonal attempt.

```
===== SARIMA 结果 =====
 model     order seasonal_order          aic          bic          aicc  test_RMSE  test_MAE   test_R2  CompositeScore
SARIMA (0, 0, 0)   (0, 0, 0, 5) -2,547.002340 -2,542.111731 -2,546.990108   0.078368  0.051896 -0.002055        0.700000

===== ARIMA 结果 =====
model     order          aic          bic          aicc  test_RMSE  test_MAE   test_R2  CompositeScore
ARIMA (1, 0, 0) -2,554.706891 -2,540.035063 -2,554.666033   0.078722  0.052448 -0.011138        0.700000
```

We then test manual ARIMA models. ARIMA(2,0,2) achieved the lowest AIC at -2555.19, but test RMSE/MAE are nearly tied across simple ARIMA(1,0,0), ARIMA(1,0,1), and the seasonal-free SARIMA candidate. Seasonality at $s = 5$ does not improve the out-of-sample error on this series. ACF at multiples of 5 lags shows no salient seasonal spikes after differencing; therefore seasonal terms were not selected.

```
===== ARIMA 各参数结果（从高到低） =====
       model     order          aic          bic          aicc  test_RMSE  test_MAE   test_R2  CompositeScore
ARIMA_manual (2, 0, 1) -2,549.339955 -2,524.891998 -2,549.253977   0.078759  0.052441 -0.012073        0.961688
ARIMA_manual (1, 0, 2) -2,545.378498 -2,520.935636 -2,545.292521   0.078753  0.052445 -0.011922        0.945127
ARIMA_manual (1, 0, 1) -2,550.413224 -2,530.854859 -2,550.351874   0.078744  0.052448 -0.011702        0.942349
ARIMA_manual (2, 0, 2) -2,555.193389 -2,525.861955 -2,555.078635   0.079008  0.052477 -0.018488        0.000000
```

2.4.2 Residual Diagnostics

Although the ARIMA(2,0,2) (the test is based on stationary time series so we let d=0) model exhibits the most desirable residual diagnostics—its residuals are approximately white noise with the highest Ljung–Box p-values and the smallest residual standard deviation—the ARIMA(2,0,1) model achieves superior predictive performance, as reflected by its higher composite score of RMSE, MAE and R^2. Given that the residuals of ARIMA(2,0,1) show only mild autocorrelation and both models deviate similarly from normality, the improvement in forecasting accuracy outweighs

the slight misspecification in residual structure. Therefore, ARIMA(2,0,1) is selected as the preferred model for its stronger predictive capability and overall practical performance.

```
===== 残差诊断结果 =====
       model     order  LjungBox_p(10)  LjungBox_p(20)     JB_p     skew     kurt  resid_std  n_obs
  ARIMA_auto (1, 0, 0)        0.007394        0.027250 0.000000 0.330301 7.339488   0.065785    984
ARIMA_manual (1, 0, 1)        0.019009        0.048888 0.000000 0.342114 7.344211   0.065739    984
ARIMA_manual (2, 0, 2)        0.718090        0.691668 0.000000 0.343841 6.979310   0.065285    984
ARIMA_manual (1, 0, 2)        0.028937        0.064810 0.000000 0.347240 7.330018   0.065725    984
ARIMA_manual (2, 0, 1)        0.036870        0.075872 0.000000 0.349980 7.322156   0.065712    984
```

2.4.3 Evaluation and Forecasting

We first evaluate the performance on the test set of data. The 1-step forecasts by ARIMA(2,1,1) remain close to the test trajectory, with narrow 95% Confidence Intervals. For example, daily test window point forecasts hover around 6.21–6.22 CNY with sensible bands.

Table 1: Forecast with 95% Confidence Interval

Date	**Forecast**	**Lower_95%**	**Upper_95%**
2024-09-24	6.214500	6.085431	6.343569
2024-09-25	6.211630	6.037736	6.385524
2024-09-26	6.212709	6.005828	6.419589
2024-09-27	6.212677	5.976739	6.448615
2024-09-30	6.212628	5.950837	6.474419
2024-10-08	6.212638	5.927354	6.497921
2024-10-09	6.212639	5.906551	6.519626
2024-10-10	6.212638	5.885382	6.539894
2024-10-11	6.212638	5.866298	6.558978
2024-10-14	6.212638	5.848212	6.577064

We then use the selected ARIMA(2,0,1) model to forecast the five-day closing price into the future.

```
===== 未来 5 天预测结果 =====
            Forecast  Lower_95%  Upper_95%
2025-09-27  5.350410   5.216724   5.484096
2025-09-28  5.350381   5.173339   5.527424
2025-09-29  5.350370   5.140431   5.560309
2025-09-30  5.350371   5.111645   5.589097
2025-10-01  5.350372   5.085953   5.614791
```

2.5 Discussion

2.5.1 Strengths

- **Interpretability:** Each AR&MA coefficient exhibits a clear time dependence, meaning that terms are explicitly represented.
- **Data efficiency:** It works well with modest data and missing exogenous information.
- **Diagnostic transparency:** The use of ACF/PACF, Ljung-Box, and information criteria makes failure modes visible, thereby enhancing the explicitness of our experiment.

2.5.2 Limitations

- **Linearity:** SARIMA assumes linear dynamics, where structural breaks and nonlinear effects (such as regime shifts and market microstructure) are not captured.
- **Sensitivity to differencing:** Both over- and under-differencing can degrade forecasts and interpretability. In fact, the ACF/PACF plot shows that our time series are close to white noise after first differencing, which might be the reason for the bad performance of the ARIMA/SARIMA models.

3 Simple models Methodology

3.1 Simple Models Setting

All four models are fitted on $\log(y)$:

- **Mean:** A constant forecast equal to the mean of the training set.
- **Naive:** Repeats the last observed value.
- **Seasonal Naive ($m = 5$):** Repeats the most recent five business days, approximating weekly trading cycles.
- **Drift:** Projects a linear trend through the first and last training observations.

3.2 Simple Models Result

file	model	RMSE	MAE	MAPE	MASE	R2
600028	mean	1.796882	1.767093	30.317162	39.319968	-29.411989
600028	naive	0.546306	0.480754	8.556777	10.697367	-1.811110
600028	seasonal_naive	0.464098	0.404061	7.159677	8.990852	-1.028733
600028	drift	1.271877	1.109502	19.748352	24.687779	-14.236872

4 Proposed Model Introduction

4.1 Objectives

The primary objectives of this study are:

1. To develop a machine learning pipeline for predicting stock prices at multiple time horizons (1-day to 5-day ahead)
2. To engineer a comprehensive set of features from raw market data using technical analysis indicators
3. To systematically evaluate model performance using proper time series cross-validation
4. To compare prediction accuracy across different forecast horizons

4.2 Dataset

The dataset consists of panel data from multiple Chinese stocks (from 2020 to 2025). Stock Index 20 was selected from the panel dataset for detailed modeling. The stock data contains daily trading information including price data (OHLC), volume metrics, fundamental ratios (PB, PE), and market microstructure variables.

5 Proposed Model Methodology

5.1 Core Methodology

The key innovation of this approach is the **multi-model framework**:

1. **Five Independent Models**: One XGBoost model is trained for each prediction horizon (T+1, T+2, T+3, T+4, T+5 days)

2. **Return Prediction**: Each model predicts the forward return: $\text{return}[T+i] = (\text{price}[T+i]/\text{price}[T]) - 1$

3. **Price Transformation**: Predicted returns are converted back to prices using: $\text{predicted\_price}[T+i] = \text{price}[T] \times (1 + \text{predicted\_return}[T+i])$

4. **Price-Level Evaluation**: Final evaluation metrics (RMSE, MAE, R^2) are computed on the transformed prices, not returns

This approach allows each model to specialize in its specific prediction horizon, potentially capturing different patterns at different time scales.

6 Proposed Model Model Training

6.1 XGBoost Configuration

XGBoost was chosen for its ability to:

1. Handle non-linear relationships through tree-based models

2. Provide feature importance for interpretability

3. Prevent overfitting through built-in L1/L2 regularization

4. Handle missing values natively

6.2 Hyperparameter Optimization

For each of the 5 models, hyperparameters were optimized using:

1. **Method**: RandomizedSearchCV with TimeSeriesSplit (3 folds)

2. **Iterations**: 15 random combinations

3. **Scoring**: Negative Root Mean Squared Error

4. **Parameter Grid**: n_estimators [200, 400, 600], learning_rate [0.01, 0.03, 0.05], max_depth [4, 5, 6], subsample [0.8, 0.9], colsample_bytree [0.8, 0.9], reg_alpha [0, 0.1, 0.5], reg_lambda [0.1, 0.5, 1], gamma [0, 0.1], min_child_weight [1, 3]

6.3 Training Process

1. **Feature Selection**: Performed on training + validation set (80%)
2. **Hyperparameter Tuning**: Used training set for fitting, validation set for evaluation with TimeSeriesSplit
3. **Final Training**: Combined training + validation (80%) with early stopping (50 rounds)
4. **Evaluation**: Tested on held-out test set (20%)

Each model was trained independently, allowing different optimal hyperparameters for each prediction horizon.

7 Proposed Model Results

7.1 Model Performance Summary

The following table presents the comprehensive evaluation results for all five prediction horizons. Metrics are computed on **predicted prices** (after transforming returns to prices):

```
--- 最后一日预测（价格）vs 真实值对比 ---
Target_Horizon  True_Price  Predicted_Price  Price_Error  True_Return  Predicted_Return
     target_1d      5.3200           5.3652       0.0452      -0.0075            0.0010
     target_2d      5.3600           5.3696       0.0096       0.0000            0.0018
     target_3d      5.3600           5.3745       0.0145       0.0000            0.0027
     target_4d      5.3500           5.3794       0.0294      -0.0019            0.0036
     target_5d      5.3500           5.3849       0.0349      -0.0019            0.0046

--- 评估（基于最后一日的5天预测价格）---
MSE（价格）: 0.000885
RMSE（价格）: 0.029749
MAE（价格）:  0.026723
（注：MAE 意味着在这次5天预测中，价格预测平均偏离 0.0267 元）
```

7.2 Stock Characteristics

The selected stock (Index 20) exhibited:

1. **Volatility type**: Low volatility (std $\approx$ 0.015-0.020)
2. **Trend type**: Sideways (weak trend, mean return $\approx$ 0.001)
3. **Volume profile**: Low volume relative to dataset

These characteristics suggest a relatively stable stock, which may explain the model's strong prediction performance, particularly for shorter horizons.

8 Proposed Model Conclusion

This study successfully implemented a comprehensive machine learning pipeline for multi-horizon stock price prediction using XGBoost. The key innovation is the **multi-model framework** where five independent XGBoost models are trained to predict returns at different horizons (T+1 to T+5 days), which are then transformed to prices for evaluation. Strong R^2 values, particularly for shorter horizons (0.94 for T+1), indicate that the model captures a substantial portion of the price variation.

9 Benchmark Models VS XGBoost

Comparison between Simple Models and XGBoost

1. **Predictive Performance.** The simple models provide basic forecasts based solely on past price patterns, assuming constant mean or persistence of previous values. Their error metrics remain very high, reflecting limited accuracy. In contrast, the XGBoost model yields much lower prediction errors (RMSE $\approx$ 0.0297, MAE $\approx$ 0.0267), indicating a substantial improvement in forecasting precision.

2. **Modeling Approach.** The simple models depend on linear extrapolation or seasonal repetition, ignoring external factors. XGBoost, however, leverages a variety of nonlinear and multivariate features (e.g., lagged returns, trading volume, and technical indicators), enabling it to capture complex short-term market dynamics that simple models cannot represent.

Comparison between ARIMA and XGBoost

Feature	**ARIMA**	**XGBoost**
Model Basis	Linear Regression	Ensemble of Decision Trees
Core Assumption	The future is a **linear combination** of the past	The future is a **nonlinear combination** of multiple features
Data Input	Usually a **single variable** (price)	**Multiple variables / features** (price, volume, indicators...)

Table 2: Comparison between ARIMA and XGBoost models

9.1 Strength of XGboost

XGBoost is an ensemble model based on decision trees, while ARIMA is built upon linear regression. Since future values often involve complex nonlinear interactions among many features, XGBoost's predictions tend to be closer to reality. Moreover, XGBoost uses multiple variables

(such as price, trading volume, and other indicators) for prediction, whereas ARIMA relies only on a single variable (typically the closing price). Unlike ARIMA, which requires differencing the data to achieve stationarity—thereby losing some information—XGBoost does not need this preprocessing step.

9.2 Weakness of XGBoost

1. **Lack of Temporal Dependency Modeling.** The XGBoost model treats each trading day as an independent observation, relying solely on current-day features to forecast future values. This design neglects the inherent temporal dependency of financial data, making the model effective in capturing short-term momentum but inadequate for learning long-term sequential patterns.

2. **Feature Construction Limitation for Multi-Step Forecasting.** In the implementation, all horizons ($T+1$ to $T+5$) are predicted directly from the same set of features at time T. Such a direct multi-horizon setup prevents the model from incorporating intermediate temporal information, leading to cumulative prediction bias and poor adaptability to dynamic market changes.

References

[Wind Information Co., Ltd., 2024] Wind Information Co., Ltd. (2024). Wind economic database (edb). https://www.wind.com.cn/portal/zh/EDB/index.html. Accessed: 2025-11-05.