CSE 258 Assignment 2: Predictive Modeling for Stock Prices

1 Dataset

The dataset used in this study consists of the constituent stocks of the SSE 50 Index. The SSE 50 is composed of 50 stocks selected by the Shanghai Stock Exchange, characterized by large market capitalization and good liquidity, classified using scientific and objective methods. These 50 stocks are essentially high-quality blue-chip stocks from various industries, and it is rare for them to face bankruptcy or delisting. Moreover, the constituent stocks of the SSE 50 are generally large-scale companies with long histories, making their data cycles suitable for training neural network models. Therefore, selecting the SSE 50 constituent stocks as the subject of this research is reasonable. Specifically, We used the trading data of each stock in the SSE 50 from January 2, 2014, to December 29, 2023, spanning a period of 10 years.

The market data selected for this study includes date, opening price (*open*), closing price (*close*), highest price (*high*), lowest price (*low*), and trading volume (*volume*). The data is sourced from the Choice Financial Terminal, which is considered authoritative. The date is used as the index of the time series, while the other five groups of trading data are used as input features for the stock price prediction model, with the closing price as the target variable. For example, the trading data of Haitian Flavouring & Food stock on February 11, 2014, is shown in Table 1.

Table 1: The trading data of Haitian Flavouring & Food on February 11, 2014.

Date	Open	Close	High	Low	Volume
2014/2/11	61.5	66.41	73.8	61.5	9391914

In Figure 2, we present the statistical analysis of various trading data of Haitian Flavouring & Food stock. The x-axis of the histogram represents the values of the trading data, while the y-axis represents the frequency of each value. As observed from the figure, we can see that the distributions of the trading data are relatively concentrated, with few outliers. The four price indicators—open, close, high, and low—are mostly distributed between 25 and 225 yuan, while the volume indicator is mainly distributed between 0×10^7 and 3×10^7.

1.1 Data preprocessing

1.1.1 Missing values processing

After organizing the data, we found that not all constituent stocks were listed before 2014. Some stocks in the SSE 50 Index were listed between 2014 and 2023, resulting in missing trading data from January 2, 2014, until their listing dates. Additionally, since the SSE 50 Index selects constituent stocks based on their performance over the past six months, some stocks had insufficient trading data to meet the experimental requirements. Specifically, a total of 10 stocks had too many missing values, making effective model training impossible. The codes of these stocks are 688981.SH, 688599.SH, 688111.SH, 688041.SH, 603986.SH, 603799.SH, 603501.SH, 603259.SH, 601728.SH, and 600905.SH. Therefore, we conducted model training and prediction analysis on the remaining 40 stocks only.

1.1.2 Normalization

To ensure model stability and improve training efficiency, it is necessary to standardize the training samples. We selected different types of input features, each with different scales. Directly using these raw data for modelling could result in features with larger scales having a disproportionate influence on the model, significantly affecting its predictive performance and generalization ability. As shown in Figure 2, we observe that the distribu-

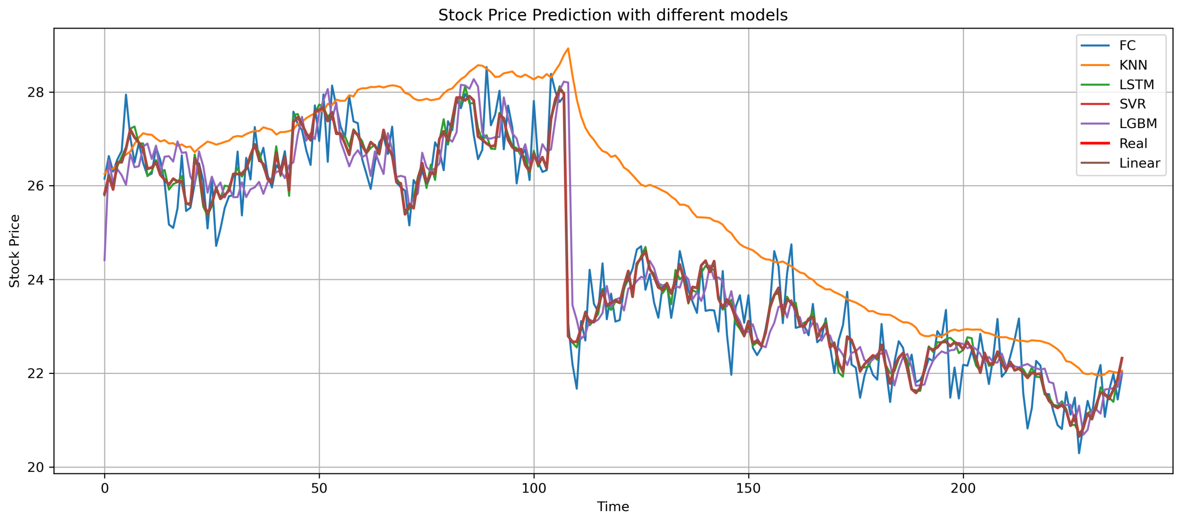

Figure 1: Stock Price Prediction with different models

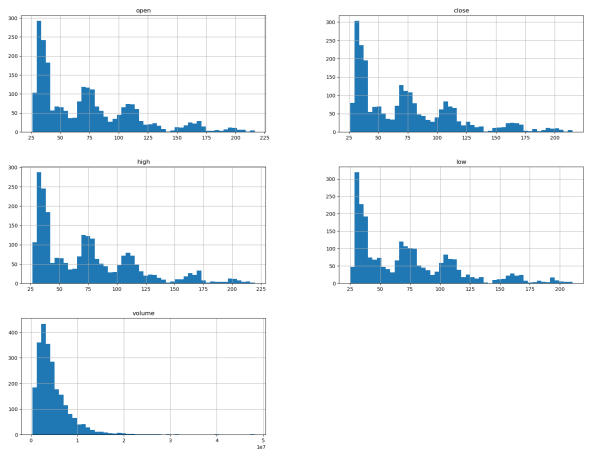

Figure 2: Histogram of Haitian Flavouring & Food Trading Data

tions are relatively concentrated, with few outliers. So we use the MinMaxScaler to scale the data in each column to the range [0, 1]. The formula is as follows:

$$x_{\text{scaled}} = \frac{x - x_{\text{min}}}{x_{\text{max}} - x_{\text{min}}} \tag{1}$$

where x represents the original input data, and x_{min} and x_{max} represent the minimum and maximum values of the respective feature column.

1.2 Dataset split

We split the dataset into training, validation, and test sets in an 8:1:1 ratio. Specifically, we use data from early 2014 to the end of 2021 as the training set, data from 2022 as the validation set, and data from 2023 as the test set to ensure consistency across different models for comparison.

The stock price prediction model is essentially a supervised learning model, so labels need to be assigned during the training process. The model is designed with a timestep of 60, meaning that the model will look at data from the past 60 timesteps to predict the next one. In other words, the model learns stock features from the past 60 days to predict the stock price, with the target being the closing price on day 61. This mechanism is implemented by creating a sliding window in the loop, with a window size of 60 that moves one step at a time. The dimension of a single training sample generated by this sliding window is (60, n), where ”60” represents the window size—each iteration extracts a block of data (60 consecutive data points) from the normalized dataset—and ”n” represents the number of features for each stock. The label for each training sample is the closing price at the data point immediately following the corresponding sliding window (i.e., the next data point to be predicted). Ultimately, a total of 95,901 samples are generated from the 40 stocks, providing sufficient data to effectively train the stock price prediction model constructed in this study.

2 Predictive task

Stock price prediction is a classic predictive task. In real markets, stock values can be influenced by various factors, such as short-term fluctuations, industry conditions on a given day, and policy impacts, all of which affect stock prices. Among these, some factors are unpredictable. However, aside from these uncertainties, there are certain common patterns in stocks. If we can study and analyze these patterns to extract relevant features, it is possible to achieve a certain level of accuracy in stock market prediction tasks.

In this task, we will disregard the influence of other factors and use only the historical price trends of a single stock as input to predict its future price changes. In other words, we treat stock prices as a time series dependent solely on their current and past values. Such an approach tends to perform more accurately in short-term stock market predictions, as short-term stock prices are less likely to be affected by external factors.

2.1 Evaluation

To evaluate the performance of the stock price prediction model, three metrics were used: RMSE (Root Mean Squared Error), MAPE (Mean Absolute Percentage Error), and R^2 (coefficient of determination). These metrics assess the model's prediction accuracy and goodness-of-fit, providing a comprehensive evaluation of its performance.

2.1.1 Root Mean Squared Error (RMSE)

RMSE measures the standard deviation of the residuals (prediction errors). A lower RMSE indicates a better fit of the model to the observed data. It is defined as:

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

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the total number of samples.

2.1.2 Mean Absolute Percentage Error (MAPE)

MAPE evaluates the model's prediction error as a percentage of the actual values, making it scale-independent. It is particularly useful for comparing models across different datasets. The formula is:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100 \tag{3}$$

A lower MAPE indicates higher prediction accuracy.

2.1.3 Coefficient of Determination (R^2)

The R^2 score quantifies the proportion of variance in the target variable that is predictable from the features. It ranges from 0 to 1, with higher values indicating better predictive performance. It is calculated as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{4}$$

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

These metrics collectively provide a robust evaluation framework for assessing the predictive capabilities of the model.

2.2 Baselines

2.2.1 Linear Regression

Linear Regression is one of the simplest and most widely used statistical models for regression analysis. It attempts to establish a linear relationship between the input features and the target variable by fitting a linear equation of the form:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \epsilon \tag{5}$$

where y represents the target variable, x_i are the input features, β_i are the model coefficients, and ϵ is the error term. The coefficients are estimated using a method called Ordinary Least Squares (OLS), which aims to minimize the residual sum of squares (RSS) between the observed values and the values predicted by the model. Linear regression is particularly useful for datasets with linear relationships and serves as a strong baseline model for more complex regression tasks.

2.2.2 KNN regression

The K-nearest neighbours regression(KNN regression) is a simple parameter regression way. Unlike other regression methods, KNN regression does not require assumptions about the data, instead, it directly uses instances from the dataset for predictions by doing a weighted average for the nearest K samples. The KNN regression does well in small datasets.

2.3 Feature engineering

The purpose of our stock price prediction model is to forecast the future closing price of the stock. Since the future closing price is influenced by many factors, incorporating these factors as input features into our model can significantly improve the prediction performance. Although stock closing prices are volatile and non-stationary, they can exhibit relative stability within small time intervals without significant changes. Therefore, we designed our model to predict the closing prices for the next n days using the relevant data from the previous m days. Based on the previous research of scholars, we selected trading data indicators and technical indicators that reflect market trends as the input features of our model. Specifically, the trading data indicators include five types, while the technical indicators include seven types. Table 2 presents the names and abbreviations of these indicators.

2.3.1 Trading data

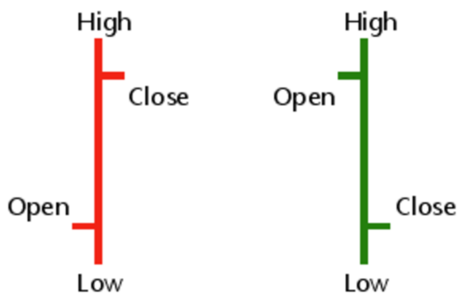

Figure 3: The Open-High-Low-Close Bar

We analyze key price indicators to better understand their impact on stock price prediction. In most cases, the five most important factors to a stock are its high, low, open, close price and volume3. The open price represents the first transaction price after the market opens, formed through an auction in the morning where both buyers and sellers submit quotes based on the historical price and their expectations. Therefore, the open price reflects market participants' expectations for the future value of the stock.

The high price and low price during a trading day represent the competition between buyers and sellers. A large difference between the high and low indicates significant disagreement about the stock's future prospects, whereas a small difference suggests that both parties share a similar positive outlook if the high price is much higher than the previous closing price and the current opening price.

The close price is the price generated by an auction in the final three minutes of trading. The closing price represents the market condition at the end of the trading day and serves as an important reference for the next day's opening price, which makes it useful for predicting future price trends.

The trading volume is the number of shares traded during the day, reflecting the true state of buying and selling forces. A significant increase in trading volume suggests high market activity, with both sides having a positive outlook on the stock's future and vice versa.

The trading data of a stock, including open, close, high, low, and volume, can provide insights into future price trends. Furthermore, volume and price data can be used to construct technical indicators that enhance stock price predictions. Therefore, we include these trading data as part of the input features for our stock price prediction model.

2.3.2 Technical indicators

Based on stock volume and price data, many technical indicators can be constructed through calculations. The main types of technical indicators include overlap indicators, momentum indicators, volume indicators, cycle indicators, and volatility indicators. To avoid overfitting caused by selecting too many technical indicators, we chose several representative technical indicators as input features for the model.

The technical indicators we use are as follows:

SMA_5 (5-day Simple Moving Average): The simple average of closing prices over the past 5 trading days, helping to analyze short-term market trends and smooth short-term fluctuations.

SMA_10 (10-day Simple Moving Average): The simple average of closing prices over the past 10 trading days, used for analyzing relatively longer market trends.

The calculation formula is as follows.

$$SMA_n = \frac{1}{n}\sum_{i=0}^{n-1} P_{t-i} \tag{6}$$

where P_{t-i} represents the price on day $t-i$, and n is the length of the moving average window.

EMA_12 (12-day Exponential Moving Average): The exponential average of closing prices over the past 12 trading days, which places more emphasis on recent data compared to SMA, making it more sensitive to price changes.

EMA_26 (26-day Exponential Moving Average): The exponential average of closing prices over the past 26 trading days, reflecting longer-term trends.

The calculation formula is as follows.

$$EMA_n = \alpha \cdot P_t + (1-\alpha) \cdot EMA_{t-1} \tag{7}$$

where α is the smoothing factor, defined as:

Table 2: Stock Indicators

Category	Stock Indicator	Abbreviation
Trading Data	Open price	$open$
	Close price	$close$
	High price	$high$
	Low price	low
	Trading volume	$volume$
Technical Indicators	5-day, 10-day simple moving average	SMA_5, SMA_10
	12-day, 26-day exponential moving average	EMA_12, EMA_26
	Exponential smoothing of moving average	$MACD$
	Signal line	$Signal\_Line$
	14-day relative strength index	RSI_14

$$\alpha = \frac{2}{n+1} \tag{8}$$

MACD (Moving Average Convergence Divergence): Calculated as the difference between the 12-day EMA and the 26-day EMA, this indicator measures the momentum and trend changes of the stock. MACD helps identify the strength of trends and potential buy or sell signals.

The calculation formula is as follows.

$$MACD = EMA_{12} - EMA_{26} \tag{9}$$

Signal Line: Typically the 9-day EMA of the MACD, used to identify buy and sell signals. When the MACD line crosses the Signal Line, a trading signal is often formed.

The calculation formula is as follows.

$$Signal\ Line = EMA_9(MACD) \tag{10}$$

RSI_14 (14-day Relative Strength Index): The relative strength index calculated based on price changes over the past 14 trading days, used to measure overbought or oversold conditions of an asset. RSI values range from 0 to 100, with values above 70 indicating overbought and values below 30 indicating oversold.

The calculation formula is as follows.

$$RS = \frac{\text{Average Gain over } n \text{ periods}}{\text{Average Loss over } n \text{ periods}} \tag{11}$$

$$RSI_n = 100 - \frac{100}{1 + RS} \tag{12}$$

where RS is the ratio of the average gain to the average loss over the past n periods.

3 Model

3.1 SVR

Support Vector Regression(SVR) is a type of Support Vector Machine(SVM) which is designed for regression tasks. SVR predicts continuous values, which is different from SVM. SVR finds a function that deviates from the actual target values by a small margin while being as flat as possible. The difference between SVM and SVR is that SVM maximizes the distance from the hyperspace to the nearest sample while SVR minimizes the distance from the hyperspace to the farthest sample.

3.2 Neural Network

We use two kinds of neural network methods to predict stock prices. The fully connected neural network and the Long short-term memory network(LSTM)[1]. Deep Neural Network(DNN) is a kind of neural network that performs multiple tasks including regression. We apply a fully connected network, which contains two fully connected layers, a dropout layer and two ReLU layers. In each layer, there are cells that simulate human neural cell actions. each cell receives input from the previous layer, multiple with inner weight, adds bias to it and delivers it to the next layer. We add dropout layers in the hidden layer. The dropout layer will randomly drop some neural cells from the layer to generate different outputs during training and avoid overfitting probability. The ReLU layer is a kind of activation function in deep learning. It is defined as $f(x) = max(0, x)$, which introduces non-linearity to the neural network, allowing the model to learn complex patterns and sparse activation to reduce overfitting.

After the forward pass, the backward propagation helps minimize the error in its predictions.

By using the chain rule, we compute the partial derivative of each variable in the neural network from the output layer to the hidden layers. Then we use gradients to update the weights and apply algorithms like Stochastic Gradient Descent(SGD).

The second model we apply is LSTM. LSTM is the updated version of Recurrent Neural Networks(RNN), which is meant to solve sequence data. The stock price itself is a kind of sequence data, in which the future price is related to the present price and the previous price. The RNN introduced the hidden state, which is able to extract features from sequential data and turn it into output. However, RNN faces some problems when it comes to longer sequences. The earlier input has little influence on the output, as newer input data change the hidden state. The problem is called the Long-Term Dependencies. LSTM avoids the problem by introducing different gate structures, the forget gate, the input gate and the output gate. The forget gate decides what information to discard from the cell state. The input gate decides what new information to add to the cell state and the output gate decides what part of the cell state contributes to the output. The cell remains in the network as well as the hidden state. The structure of LSTM can remember long-term memory.

3.3 LightGBM

LightGBM (Light Gradient Boosting Machine)[2] is an efficient, open-source, distributed machine learning tool developed by Microsoft based on the gradient boosting framework. It uses decision tree-based learning algorithms and is highly suitable for handling large-scale datasets, excelling particularly in classification and regression tasks.

LightGBM is built on the gradient boosting framework, which combines multiple weak learners (typically decision trees) into a strong learner to minimize residuals. In each iteration, a new tree tries to fit the residuals from the previous round, thereby gradually improving the model's predictive performance. Unlike traditional decision trees that grow based on a hierarchical approach, LightGBM uses a leaf-wise growth strategy, prioritizing the leaf node with the highest gain for splitting. Although this approach may lead to an unbalanced tree structure, it is very effective in reducing training errors and also improves training speed.

In this model, we use RMSE as the evaluation metric and Gradient Boosting Decision Tree (GBDT) as the boosting method. The number of leaf nodes is set to 31, and the learning rate is 0.05. In each iteration, 90% of the features are randomly selected for training to prevent overfitting. We iterate up to 200 times until the model performance no longer improves, and employ an early stopping strategy that halts training if the validation performance does not improve within 10 rounds.

4 Related work

4.1 Dataset source and usage

The dataset used in this study is derived from the SSE 50 Index, which represents the 50 most representative companies listed on the Shanghai Stock Exchange (SSE). The SSE 50 constituents were chosen due to their size and high liquidity, which provide strong representativeness in the Chinese equity market. The data spans from January 2, 2014, to December 29, 2023, including daily trading information such as open price, close price, high, low, and volume. In addition, the study incorporates various technical indicators as features, such as Moving Average (MA), Relative Strength Index (RSI), and Bollinger Bands, to diversify the feature set and help the model better capture the underlying patterns of stock price fluctuations [3].

4.2 Similar datasets

Similar datasets used in past studies include the CSI 300 Index and the Dow Jones Industrial Average (DJIA). Researchers commonly applied models like ARIMA [4] and LSTM [5] to these indices for stock price prediction. Zhang & Wang [6] used LSTM with technical indicators on the SSE 50 Index, demonstrating improved accuracy by incorporating additional features like Moving Average (MA).

4.3 Past studies and latest methods

In recent years, significant advances have been made in stock market prediction methods, particularly in deep learning and ensemble learning. These include combinations of LightGBM (LGBM), LSTM, fully connected neural networks (FCNN), as well as Graph Neural Networks (GNN) and Transformer models. For instance, GNN has been used to model the relationships between stocks, effectively capturing market dependencies [7]. Transformer models also excel at capturing important patterns in long-term sequences, providing superior performance compared to traditional models [8]. Additionally, combining LSTM with LightGBM has improved prediction stability and accuracy by leveraging different feature relationships.

More recent techniques include the use of Reinforcement Learning for optimizing trading strategies in dynamic environments [9]. Ensemble learning, like combining LSTM and LightGBM, further enhances prediction performance by utilizing the strengths of each model. These advancements have allowed models to better manage complex market dynamics, improving responsiveness and accuracy in predicting stock market fluctuations.

4.4 Conclusion and comparison

Existing studies generally agree that the SSE 50 constituents exhibit high liquidity, and their price movements tend to reflect broader market trends. Traditional models such as ARIMA and LSTM have been proven effective for single time-series forecasting, and model performance is further improved after incorporating technical indicators. For example, Ke et al.'s research showed that using LightGBM combined with technical indicators performs better in handling nonlinear features compared to traditional time-series models [10]. In this study, by combining LSTM, LightGBM, and fully connected neural networks, we found that these models perform better in capturing short-term trends and abnormal fluctuations in different market environments. Preliminary results indicate that multi-feature model combinations provide greater stability in dealing with complex financial market fluctuations, and prediction accuracy has also improved, making it consistent with the conclusions of existing literature and further validating its effectiveness.

5 Results and conclusions

In the task, we tried different models including regression, statistic analysis, and neural network. By analyzing 42 stocks on nearly 5 years of price and testing the model for the later year, we obtain the results and analyze them based on three criteria, RMSE, MAPE and R^2. The performance of neural networks is mostly better than that of naive machine learning methods, while linear regression has a better performance than expected.

From the table, the naive machine model, which

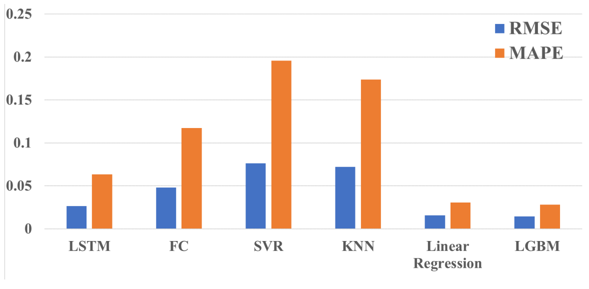

Figure 4: Criteria on different models

is SVR and KNN, has a worse result in nearly all the stocks. They have an average of 0.03 and more error higher than other models. In most cases, KNN is a little bit worse than SVR. KNN is a non-parameter model, which means it does not have assumptions on the distribution of data and is easier to be affected by noise. Also, KNN is more sensitive than SVR. SVR can optimize the target to be more robust on the not normally distributed data. The neural network models are trained with fixed hyperparameters, so in some cases, the vanilla neural network model cannot fit the data well. The efficient gradient boosting algorithm LGBM, without doubt, is better than vanilla FC and LSTM. Surprisingly, the linear regression has quite good performance. By using LinearRegression in sklearn, the linear regression achieves nearly a similar performance as LGBM.

References

[1] S Hochreiter. Long short-term memory. *Neural Computation MIT-Press*, 1997.

[2] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: a highly efficient gradient boosting decision tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 3149–3157, Red Hook, NY, USA, 2017. Curran Associates Inc.

[3] J. Bollinger. *Bollinger on Bollinger Bands*. McGraw Hill Professional, 2002.

[4] George EP Box, Gwilym M. Jenkins, Gregory C. Reinsel, and Greta M. Ljung. *Time Series Analysis: Forecasting and Control*. John Wiley & Sons, 2015.

[5] Thomas Fischer and Christopher Krauss. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669, 2018.

[6] X. Zhang and Y. Wang. Improving stock price prediction using lstm with technical indicators on sse 50 index. In *Proceedings of the International Conference on Artificial Intelligence*, 2022.

[7] X. Cao, Z. Wang, and Y. Li. Graph neural networks for stock market prediction. *Journal of Financial Data Science*, 2023.

[8] J. Zhao and Q. Liu. Application of transformer models in long-term sequence forecasting for financial markets. *Journal of Machine Learning Research*, 2022.

[9] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

[10] Guolin Ke, Qiwei Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qi Ye, and Tie-Yan Liu. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

A Predicted Data

Model	Criterial	Stock								
		baoli	beifangxitu	changcheng	changjiangdianli	fenjiu	guodiannanrui	haier	haitian	hangfa
	rmse	0.020569	0.01962	0.013004	0.026672	0.036558	0.017327	0.032907	0.011417	0.015394
LSTM	mape	0.02809	0.054919	0.027412	0.024365	0.065822	0.026563	0.046463	0.077722	0.035208
	$R\hat{2}$	0.968301	0.880003	0.892241	0.696937	0.644393	0.762546	0.617009	0.984735	0.899609
	rmse	0.024979	0.016226	0.033485	0.035137	0.032248	0.037783	0.03273	0.015703	0.01919
FC	mape	0.036967	0.042358	0.07901	0.033045	0.057709	0.065401	0.043272	0.092139	0.045376
	$R\hat{2}$	0.949834	0.92815	0.096924	-0.107736	0.711922	-0.113046	0.643624	0.974248	0.848104
	rmse	0.0401	0.048769	0.026179	0.095382	0.093561	0.036521	0.045614	0.057168	0.021461
SVR	mape	0.059638	0.162246	0.05581	0.098232	0.198015	0.085603	0.067102	0.238377	0.050832
	$R\hat{2}$	0.882528	0.206497	0.632997	-5.18933	-1.73448	0.67945	0.351345	0.490168	0.858918
	rmse	0.096846	0.041945	0.039556	0.26837	0.094359	0.045293	0.10251	0.041446	0.0485
KNN	mape	0.167001	0.141933	0.089112	0.305043	0.179205	0.10244	0.179084	0.172209	0.11877
	$R\hat{2}$	0.314796	0.413025	0.162065	-47.9979	-1.78132	0.506967	-2.276	0.73204	0.27945
	rmse	0.018398	0.010294	0.012691	0.011127	0.014982	0.016764	0.016726	0.008361	0.012227
Linear	mape	0.025052	0.02593	0.028266	0.010035	0.025796	0.032085	0.022255	0.027807	0.026225
	$R\hat{2}$	0.975272	0.964649	0.913741	0.915771	0.929884	0.932458	0.912786	0.989094	0.954205
	rmse	0.018384	0.010477	0.012903	0.025176	0.015065	0.015717	0.015494	0.013678	0.014586
LGBM	mape	0.025773	0.028875	0.027347	0.024112	0.02702	0.026806	0.020425	0.053928	0.03075
	$R\hat{2}$	0.975309	0.963375	0.910847	0.568785	0.929104	0.940631	0.925161	0.970816	0.934828