Predictive Modeling of NASDAQ's Closing Auction: An Empirical Comparison of Machine Learning Approaches for Stock Price Movement Forecasting

Maneesh Sarma Sistla Naga Sai
maneeshsistla@asu.edu
Arizona State University, SCAI
Tempe, Arizona, USA

Saurav Anchlia
sanchlia@asu.edu
Arizona State University, SCAI
Tempe, Arizona, USA

Jai Narula
jnarula1@asu.edu
Arizona State University, SCAI
Tempe, Arizona, USA

Shrinkhala Kayastha
skayast1@asu.edu
Arizona State University, SCAI
Tempe, Arizona, USA

Jay Ashokbhai Jayani
jjayani@asu.edu
Arizona State University, SCAI
Tempe, Arizona, USA

Abstract

This research project delves into the domain of high-frequency trading in stock exchanges, with a particular focus on the Nasdaq Stock Exchange's closing auction. The objective of this study is to assess the feasibility of predicting the 10-minute future price movements of individual stocks listed on NASDAQ relative to a synthetic index using a provided dataset comprising order book and auction data. Through the utilization of machine learning models, the research aims to identify patterns and relationships within the dataset, shedding light on the predictability of stock price movements during this critical time frame. The relative importance of various features is explored, including imbalance size, buy/sell flags, and reference prices, in the predictive accuracy of these models. Additionally, this study compares and contrasts the effectiveness of statistical and deep learning machine learning methods in capturing and exploiting the underlying patterns in the data for prediction. The outcomes of this project have the potential to enhance market efficiency and accessibility during the intense final ten minutes of trading, benefiting market participants and mirroring the real-world challenges faced by industry professionals like those at Optiver.

Keywords: Stock Market Forecasting, Time Series Analysis, Machine Learning, Recurrent Neural Networks, LSTM, GRU, Gradient Boosting

1 Introduction

We have chosen to participate in the Kaggle competition "Optiver - Trading at Close" on Kaggle. In this competition, we are challenged to develop a model capable of predicting the closing price movements for hundreds of Nasdaq-listed stocks using data from the order book and the closing auction of the stock. Information from the auction can be used to adjust prices, assess supply and demand dynamics, and identify trading opportunities.

Time series forecasting for stock prices is a challenging task primarily due to the non-stationary nature of financial data. Non-stationarity implies that the statistical properties of the time series, such as mean, variance, and covariance, change over time. Stock prices often exhibit non-stationary behavior characterized by trends, seasonality, and irregular fluctuations influenced by various macroeconomic and market-specific factors. In the context of stock price forecasting, the non-stationary behavior poses a significant challenge for traditional time series models, such as ARIMA, which assume the data to be stationary. To address this, various techniques have been developed to transform the data into a stationary form. This includes differencing, detrending, and other preprocessing methods to remove trends and seasonality from the data. Moreover, advanced machine learning models like recurrent neural networks (RNNs) and long short-term memory (LSTM) networks have demonstrated capabilities in capturing complex temporal dependencies and handling non-stationary data more effectively. These models can adapt to changing patterns and dynamics within the data, making them well-suited for stock price forecasting. Researchers and practitioners often integrate exogenous variables and financial indicators to enhance the predictive power of the models, accounting for external market influences and macroeconomic factors that contribute to non-stationarity in stock prices. Understanding the non-stationarity of stock price data and implementing appropriate techniques to handle this complexity are crucial steps in developing accurate forecasting models that can capture the intricate dynamics of financial markets. Effective risk management strategies, robust feature engineering techniques, and a comprehensive understanding of market dynamics are essential components in mitigating the challenges posed by non-stationary time series data in stock price forecasting.

2 Related Work

In recent years, the field of stock price prediction has witnessed substantial progress, with several studies proposing novel methodologies to enhance forecasting accuracy and robustness. Notably,[7] introduced a deep learning model based on Long Short-Term Memory (LSTM) for predicting stock prices, demonstrating superior performance. However, their evaluation was confined to a specific subset of stocks, warranting further investigation into the model's scalability and generalizability. Similarly, [9] approach leveraged technical indicators and machine learning techniques for predicting stock and stock price index movements, although its scope was limited to a specific index, raising questions about its broader applicability. In a more recent study, [2] evaluated various deep learning models for stock price prediction, emphasizing the importance of comprehensive model evaluation and cross-validation. Nevertheless, the study underscored the necessity of interpreting the underlying factors affecting stock prices, a facet that was not fully addressed. Additionally, [13] data-driven approach integrating disparate data sources for stock market movement prediction demonstrated promising results, albeit its application was restricted to a specific stock, warranting further investigation into its versatility across multiple stocks and indices. Moreover, [3] hybrid model of multiple regression analysis (MRA) and extreme learning machine (ELM) for stock price prediction showcased improved accuracy, but its limited scope emphasized the need for broader applicability and a more comprehensive analysis of input features. [6] proposal of an ARIMA model for stock market movement prediction underscored the significance of time series analysis and the challenges associated with the assumption of data stationarity. However, the study highlighted the limitations of ARIMA models in handling multivariate time series data and the requirement for careful parameter selection. Furthermore, [12] research on using deep learning to identify universal features of price formation in financial markets emphasized the enhanced forecasting accuracy of deep learning models. Yet, its limitations included the complexity of model interpretation and the sensitivity to data quality. Additionally, [11] comprehensive deep learning system for short-term stock market price trend prediction highlighted the significance of feature engineering and the challenges associated with varying prediction term lengths, providing valuable insights into optimizing feature engineering for improved model performance. Finally, [8] comprehensive literature review underscored the need for an in-depth understanding of the different machine learning algorithms and their applicability in stock market predictions.

Table 1. Train dataset summary

Instances	No. of Features	No. of Stocks	No. of Days
5270980	16	200	481

3 Data

3.1 Dataset Description

The Kaggle dataset by Optiver contains historic time series data for the daily ten minute closing auction on the NASDAQ stock exchange, with a target metric that estimates the 60 second future move in the weighted average price (wap) of the stock. Table 1 briefly summarizes the dataset.

3.2 Features

3.2.1 Basic Feature Engineering. The following features are extracted by performing straightforward transformations on the vanilla features, as they provide key insights that can enhance the predictive power of machine learning models in this context.

1. Directional Imbalance: Redefining imbalance size by multiplying it by the imbalance_buy_sell_flag effectively indicates the direction of imbalance, allowing the model to distinguish between buy-side and sell-side imbalances.
2. Imbalance-to-Matched-Size Ratio: Calculating the ratio of imbalance size to matched size provides a normalized measure of how significant the imbalance is relative to the traded volume, potentially aiding in the identification of market trends.
3. Bid-Ask Size Ratio: The bid size to ask size ratio can indicate the strength of demand for buying or selling in the market, assisting in assessing market sentiment and potential price movements.
4. Bid-Ask Size Difference: The difference between bid size and ask size serves as an estimate of the spread, offering insights into the supply and demand dynamics and potential market volatility.
5. Bid-Ask Price Ratio: The bid size to ask price ratio can indicate the disparity in temperature of buying and selling of a stock in the market.
6. Bid-Ask Size Difference: The difference between bid price and ask price serves as an estimate of the spread, offering insights into the cost of trading and potential price impact
7. Total Market Depth: The sum of bid size and ask size (total size) represents the overall market depth, which can be essential for understanding the liquidity and stability of the market.
8. Bid-Ask Spread Percentage: Calculating the bid-ask spread percentage relative to the total size provides a normalized measure of the spread, allowing for a more accurate assessment of the spread's significance in

relation to the overall market depth, which is valuable for price prediction and volatility analysis.

3.2.2 Technical Indicators. [1] By strategically integrating the following technical indicators into the feature space, we aim to capture complex market dynamics and price patterns, enhancing the predictive capabilities of our models and enabling more informed investment decisions.

1. Momentum indicators, such as the Relative Strength Index (RSI) and Stochastic Oscillator, help gauge the speed and change of price movements, providing insights into overbought or oversold conditions.
2. Moving averages, including Simple Moving Averages (SMA) and Exponential Moving Averages (EMA), offer valuable information on trend directions and potential reversal points.
3. Ratios, like the Price-to-Earnings (P/E) ratio and Price-to-Book (P/B) ratio, allow investors to evaluate a stock's valuation relative to its earnings and book value.
4. Volatility measures, such as Average True Range (ATR) and Bollinger Bands, quantify the degree of price fluctuations, aiding in assessing risk and potential price movements.
5. Oscillation indicators, including the Moving Average Convergence Divergence (MACD) and On-Balance Volume (OBV), help identify market momentum shifts and trend reversals.
6. Standard deviation calculations provide insights into the dispersion of data points around the mean, enabling a deeper understanding of price variability and market volatility.
7. Derived Features from Weighted Average Prices (WAP): Rolling mean and standard deviation of the WAP over various time windows offer insights into price trends and stability. Additionally, Volume Weighted Average Price (VWAP) can be useful because it reflects actual market dynamics by considering both trading volume and price.
8. Bollinger Bands around the WAP can help identify potential price breakouts and trend reversals, considering the upper and lower bands as critical support and resistance levels.
9. Moving Average Convergence Divergence (MACD) on WAP signals potential changes in the strength and direction of the price trend, aiding in identifying buy and sell signals.
10. Imbalance Persistence can be computed to measure the duration of an imbalance by tracking how many time periods it persists.

4 Methods

4.1 Statistical Machine Learning methods

In the field of stock price prediction, the accurate modeling of short time series data plays a critical role in providing insights into the volatile dynamics of financial markets. Various statistical methods have emerged as powerful tools for capturing the intricate patterns and trends present in these short time series, contributing significantly to the robustness and reliability of stock price forecasting models. Among these methodologies, Gradient Boosting Machines (GBM) have garnered substantial attention, with specific implementations like XGBoost and CatBoost serving as prominent choices for their adeptness in handling minute-level data.

Advantages and Disadvantages of XGBoost in Short Time Series Modeling: XGBoost, renowned for its capacity to handle vast datasets and deliver high predictive accuracy, has been widely adopted for short time series modeling. Its effective regularization techniques aid in mitigating overfitting, while its parallel processing capabilities cater to the demands of time-sensitive tasks. Despite its proficiency, XGBoost necessitates meticulous parameter tuning, and its intricate architecture may impede interpretability, posing potential challenges in model comprehension and analysis.

Advantages and Disadvantages of CatBoost in Short Time Series Modeling: In a similar vein, CatBoost, characterized by its robust treatment of categorical data, presents an advantageous option for short time series analysis. The algorithm's seamless handling of missing data and its intrinsic resistance to overfitting, owing to the application of ordered boosting, make it particularly appealing for modeling tasks involving time-sensitive financial data. However, the computational resources required for training and the potential complexities associated with hyperparameter tuning represent some of the limitations of utilizing CatBoost for short time series modeling in stock price prediction.

Significance of the Generalized Method of Moments (GMM) in Short Time Series Modeling: The Generalized Method of Moments (GMM) framework holds significance in the realm of short time series modeling, offering a flexible approach for capturing the nuanced relationships embedded within financial data. GMM's adaptability in addressing endogeneity and measurement errors within the modeling process makes it a valuable tool for understanding the intricate dynamics of stock price movements. However, the reliance on strong instruments and the sensitivity of results to the choice of moment conditions require careful consideration when employing GMM for short time series modeling in the context of stock price prediction.

By recognizing the strengths and limitations of these statistical methodologies, researchers and practitioners can effectively navigate the complexities of short time series modeling, ensuring the development of robust and reliable stock price forecasting models.

4.2 Deep Learning methods

4.2.1 Recurrent Neural Network (RNN). Recurrent Neural Networks (RNNs) are a specialized deep learning architecture well-suited for time series forecasting. They are characterized by their ability to process sequential data by maintaining an internal memory state that retains information from past time steps, allowing them to capture temporal dependencies and time lags in the data. This inherent memory makes them highly effective for understanding autoregressive patterns often observed in financial time series.

In the context of stock market forecasting, RNNs can accept various input features, including historical price data, trading volumes, or technical indicators. Their flexible architecture permits customization to account for multiple time steps in the prediction task, making them adaptable for scenarios where you need to forecast not just the next step but multiple steps into the future.

However, it is important to note that standard RNNs can face challenges with long sequences due to issues like vanishing or exploding gradients, which affect their ability to capture long-range dependencies effectively. To address this limitation, variations of RNNs like [5] Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) have been developed. These advanced RNN structures include mechanisms that better handle long sequences and mitigate the vanishing gradient problem.

4.2.2 Gated Recurrent Unit (GRU). [4] Gated Recurrent Units (GRUs) represent an advanced adaptation of Recurrent Neural Networks (RNNs) and they tackle key challenges encountered in traditional RNNs, especially when dealing with extended sequences and gradients that vanish or explode. GRUs strike a balance between the simplicity of standard RNNs and the complexity of Long Short-Term Memory (LSTM) networks.

Compared to LSTMs, GRUs possess a more streamlined architecture with fewer parameters, resulting in faster training and reduced computational demands. This efficiency is advantageous for real-time stock market forecasting scenarios where speed matters. GRUs offer a well-rounded solution for time series forecasting, balancing complexity with effective memory management, and thus serving as a valuable asset in the financial forecasting toolkit.

4.2.3 Long Short-Term Memory Networks (LSTM). Long Short-Term Memory (LSTM) networks, a distinctive category of Recurrent Neural Networks (RNNs), are meticulously designed to surmount the limitations that beset conventional RNNs, notably in the context of managing extensive sequences and circumventing challenges linked to gradients.

LSTM networks feature a more complex architecture with separate memory cells and gates for precise control of information flow, making them effective for capturing long-range dependencies and complex time series patterns. Gated Recurrent Units (GRUs) have a simpler structure with fewer parameters, making them computationally efficient and well-suited for tasks with shorter dependencies and where training speed is critical, yet they may be less effective on sequences with extensive long-term dependencies. The choice between LSTM and GRU depends on the specific requirements and resources of the task at hand.

4.2.4 Bidirectional Long Short-Term Memory Networks (Bi-LSTM). [10] Bidirectional Long Short-Term Memory (Bi-LSTM) networks are a specialized type of recurrent neural network architecture that offers a unique advantage in sequence modeling tasks, including natural language processing and time series analysis. Bi-LSTMs differ from traditional LSTMs by processing input sequences in both forward and backward directions. This bidirectional processing allows them to capture contextual information from past and future time steps simultaneously.

The architecture of a Bi-LSTM consists of two LSTM layers, one processing the sequence forward and the other in reverse. These two layers produce hidden states that capture contextual information from both directions. This bidirectional context understanding enhances the network's ability to comprehend the nuances of sequential data, enabling better feature extraction, improved understanding of long-range dependencies, and higher accuracy in tasks like sentiment analysis, named entity recognition, and stock price prediction. Bi-LSTMs have proven especially valuable when context in both directions is essential for making accurate predictions, as they provide a more comprehensive view of the sequence, making them a powerful tool for sequence modeling and prediction tasks.

5 Experiments

In this section, we outline the experiments conducted to evaluate and compare the performance of various machine learning models in predicting the closing auction movements of stocks on NASDAQ. The experiments are designed to provide insights into the strengths and weaknesses of individual algorithms, the impact of different techniques, and the overall effectiveness of the proposed models.

Recognizing the intrinsic sequential nature of data is paramount in various domains, particularly when dealing with time series information. Shifting partitions based on the date id serves as a strategic approach to uphold this temporal aspect. By sequentially organizing the dataset and creating partitions, a coherent representation of the evolving data structure is maintained. Dividing each partition into the first

80% as training data and the subsequent 20% as validation data further ensures that the model is trained on historical information and validated on more recent observations. This approach not only aligns with the inherent chronology of the data but also aids in evaluating the model's generalization performance on unseen future instances, reinforcing the importance of preserving the sequential order for robust and accurate modeling in time-sensitive scenarios.

Q1 Can we reduce computational requirements through data optimization?

Q2 Impact of technical indicators on model performance

Q3 Examination of DNN to capture temporal relationships.

5.1 Optimizing Data Types for Memory Usage

In order to enhance the computational efficiency of our data processing pipeline, we incorporated a strategy to optimize data types and reduce memory consumption. This involved a meticulous examination of each column's data type, with the objective of downcasting it to the most memory-efficient type without compromising data integrity. This optimization approach, particularly valuable for large datasets, contributes to faster computation and decreased storage requirements.

During this process, we iteratively assessed the numerical range of each column. For integer types, we downscaled to the smallest applicable type (e.g., int8, int16) based on the observed minimum and maximum values. Similarly, for floating-point types, we selected the most memory-efficient option (e.g., float16, float32) considering the range of values.

By implementing this data type optimization strategy, we achieved a notable reduction in memory usage. This not only facilitates faster computations but also contributes to more economical storage utilization, a critical aspect when dealing with extensive datasets.

5.2 Impact of Technical Indicators on Model Performance

In this investigation, we explore the impact of incorporating traditional technical indicators into machine learning models for predicting stock price movements. We aim to address the question of whether the inclusion of industry-standard indicators significantly enhances model performance compared to models built solely on engineered features derived from standard operations like ratio, multiplication, momentum, and volatility.

To assess the impact of technical indicators, we develop two sets of models: one using a comprehensive selection of traditional technical indicators (e.g., Relative Strength Index, Moving Averages, Bollinger Bands) and another relying solely on engineered features derived from fundamental financial metrics, market dynamics, and basic arithmetic operations. We implement machine learning algorithms, such as XGBoost, to evaluate and compare the predictive performance of these models.

5.3 Ensemble Learning

In this section, we delve into the experimentation with individual machine learning algorithms, specifically XGBoost and LightGBM. Our focus extends beyond algorithmic performance to encompass the critical aspects of feature engineering, feature importance analysis, and model interpretability.

5.3.1 Hyperparameter Tuning and Optimization.

Hyperparameter tuning is a critical aspect of optimizing the performance of machine learning models, and XGBoost offers a versatile set of parameters that can be fine-tuned to achieve optimal results. In this section, we delve into the hyperparameter tuning process for XGBoost, leveraging insights gained from the final parameters used for LightGBM.

Final Parameters:

XGBoost Parameters:	
Objective:	reg:squarederror
N Estimators:	5500
Learning Rate:	0.00877
Max Depth:	12
Min Child Weight:	0.001
Subsample:	0.6
Colsample bytree:	0.6
Reg Alpha:	0.1
Reg Lambda:	0.3
Gamma:	0
Max Delta Step:	0
Scale Pos Weight:	1
N Jobs:	4
Verbosity:	0

5.3.2 SHAP.

In our SHAP analysis, we employed the SHAP (SHapley Additive exPlanations) library, a powerful tool for interpreting the output of machine learning models. SHAP values provide a way to fairly distribute the contribution of each feature to the prediction across all possible feature combinations. Here's how we conducted SHAP analysis and feature pruning:

SHAP Value Calculation: We used the shap.TreeExplainer class from the SHAP library to create an explainer object specific to tree-based models like XGBoost. The explainer was then used to compute SHAP values for each prediction in our model. The SHAP values represent the contribution of each feature to the difference between the model's prediction and the mean prediction.

Summary Plot: We generated a summary plot to visualize the overall impact of each feature on the model's output.

This plot provides a summary of feature importance for all instances in the dataset.

Individual SHAP Value Plot: To understand the impact of features on individual predictions, we created individual SHAP value plots force plot. These plots illustrate how each feature contributes to the prediction for a specific instance.

Feature Pruning: Feature pruning involves identifying and retaining only the most important features based on SHAP values. We set a threshold for SHAP values and retained features whose contributions exceeded this threshold. This helped in focusing on the most influential features and simplifying the model for better interpretability without sacrificing predictive performance.

Dependency Plot: We utilized dependence plot to create dependency plots for selected features. These plots visualize the relationship between the values of a feature and the corresponding SHAP values, offering insights into how changes in a feature impact the model's predictions.

5.4 Capture Temporal relation in data

Time series data inherently involves sequential dependencies where past observations influence future outcomes. Recognizing patterns and dependencies over time can be challenging due to the dynamic and non-linear nature of financial markets. In this section, we explore recurrent Neural Network (RNN) based approaches as we believe they can offer a promising solution by effectively modeling sequential dependencies. As shown in Figure 1, the same architecture is applied with both RNNs and LSTM layers to do a comparative analysis.

5.4.1 Parameters for RNNs.

- The input data needed to be reshaped to add a sequence dimension so that it can be fed to the RNN. For this, sliding window sequences are generated, with a "lookback" parameter of 6. This intuitively means that 1 minute's worth of data is being considered in each sequence.
- Two RNN layers were stacked together to enhances the model's ability to capture hierarchical temporal patterns and foster improved feature abstraction in time series data.
- The hidden dimension size was chosen to be 32.

6 Results

The baseline models, consisting of XGBoost, LightGBM, CatBoost, LSTM-based RNN, and GRU-based RNN, demonstrate varying levels of predictive performance. In terms of MSE, XGBoost outperforms both LightGBM and CatBoost, showcasing its effectiveness in capturing underlying patterns in financial time series data. However, the RNN architectures (LSTM and GRU) exhibit slightly higher MSE values, suggesting challenges in capturing nuanced dependencies within the data compared to boosting methods.

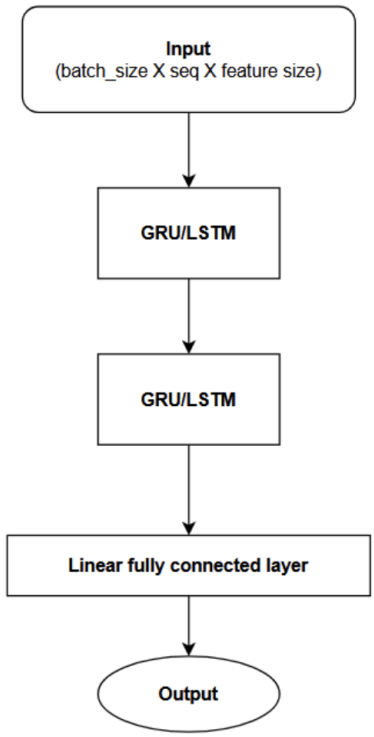

Figure 1. Architecure of the RNN model

Model	MSE	MAE
XGBoost	74.965	5.792
LightGBM	75.718	5.826
CatBoost	74.943	5.793
LSTM-based RNN	77.315	5.939
GRU-based RNN	77.127	5.914

Table 2. Baseline Model Results

Considering the MAE, XGBoost again excels, providing the lowest error among the boosting methods. The RNN models, while competitive, indicate that the baseline approach might benefit from additional feature engineering to improve predictive accuracy.

Model	MSE	MAE
XGBoost	73.505	5.742
LightGBM	73.056	5.717
CatBoost	74.041	5.931
LSTM-based RNN	75.953	5.862
GRU-based RNN	76.113	5.907

Table 3. Feature Engineering + Technical Indicators Model Results

The introduction of feature engineering and technical indicators aims to enhance the models' understanding of complex relationships within the financial data. Tables 3 reveal the impact of these enhancements on predictive performance.

Interestingly, XGBoost, LightGBM, and CatBoost exhibit improvements in both MSE and MAE with the inclusion of feature engineering and technical indicators. This underscores the importance of crafting meaningful features that capture relevant aspects of the financial data. The LSTM-based RNN and GRU-based RNN, while showing marginal improvements, still lag behind the boosting methods in this particular configuration.

In summary, our results highlight the efficacy of feature engineering and the incorporation of technical indicators in boosting methods, especially XGBoost, LightGBM, and CatBoost. However, further exploration is warranted to unlock the full potential of RNN architectures in the context of financial time series prediction.

7 Discussion

The empirical comparison of machine learning approaches for forecasting stock price movements during the Nasdaq Stock Exchange's closing auction uncovered crucial insights and considerations. Ensemble methods like XGBoost and CatBoost showcased robust predictive performance in capturing patterns within the closing auction dataset. However, applying ensemble methods to time series tasks, especially in the financial domain, presents challenges due to the non-stationary nature of stock prices. These challenges include evolving trends, abrupt shifts, and irregular fluctuations, which might limit the effectiveness of traditional ensemble techniques in handling temporal dependencies and evolving dynamics over time. Future research should explore hybrid approaches that combine ensemble methods with specialized time series models to address these challenges more effectively.

We realized the importance of extensive feature engineering in enhancing the predictive power of machine learning models for stock price movement forecasting. The inclusion of technical indicators, derived features from weighted average prices, and other relevant features significantly contributed to the models' ability to capture market dynamics. However, the challenge lies in striking a balance between feature richness and overfitting.

The computational demands of ensemble methods, particularly XGBoost and CatBoost, were significant considerations in this study. Extensive hyperparameter tuning and optimization, coupled with intricate architectures, demand substantial computational resources. Striking a balance between model complexity and interpretability is crucial. The trade-off between model accuracy and computational efficiency should be carefully navigated, especially in real-time financial forecasting scenarios where speed is paramount. Deploying these models in production environments requires robust computing infrastructure to ensure timely predictions.

The integration of SHAP analysis and feature pruning provided valuable insights into the interpretability of our models. SHAP values facilitated a granular understanding of feature contributions, enhancing the transparency of complex ensemble models. Feature pruning based on SHAP values allowed us to identify and retain the most influential features, simplifying the model without sacrificing predictive accuracy. This not only improved model interpretability but also contributed to faster inference times. In time-sensitive financial scenarios, the ability to streamline models without compromising predictive performance is a significant advantage.

8 Conclusion

Our exploration of machine learning models, including boosting methods (XGBoost, LightGBM, and CatBoost) and recurrent neural networks (GRUs and LSTMs), has provided valuable insights into their applicability in the context of financial time series data.

Boosting methods have demonstrated effectiveness in various machine learning tasks; however, their limitations in capturing rapid movements in stock prices, especially in shorter sequences, have been highlighted. We found that GRUs, despite their lower complexity compared to LSTMs, outperformed boosting methods in capturing quick price changes. This underscores the importance of selecting models that align with the specific characteristics of financial data.

To address outlier detection in short time frame movements, we proposed leveraging Principal Component Analysis (PCA) for clustering sectors. This approach aims to capture stock-sector-level patterns and enhance anomaly detection within specific market segments.

In longer sequences, our experiments with LSTMs showcased their potential to capture extended-term dependencies in the data. While LSTM models entail longer training times and increased computational resources, they are crucial for unraveling intricate, long-term relationships within financial time series.

Furthermore, our findings emphasize the necessity of heavy feature engineering to extract meaningful patterns from financial data. The combination of sophisticated models with carefully crafted features is crucial for achieving optimal predictive performance in the complex and dynamic domain of stock price prediction.

Looking ahead, we recommend exploring bi-directional LSTMs to leverage contextual information from both directions, potentially enhancing the model's understanding of complex dependencies in historical stock data. In summary, a nuanced approach that combines advanced models, tailored feature engineering, and thoughtful consideration of

the unique challenges posed by financial time series data is essential for achieving robust and accurate predictions.

References

[1] [n. d.]. Best trading indicators: A list of the 17 most used technical indicators. https://www.axi.com/int/blog/education/trading-indicators.

[2] Juan Arosemena, Noel Pérez, Diego Benítez, Daniel Riofrío, and Ricardo Flores-Moyano. 2021. Stock Price Analysis with Deep-Learning Models. In *2021 IEEE Colombian Conference on Applications of Computational Intelligence (ColCACI)*. IEEE, 1–6.

[3] Depei Bao and Zehong Yang. 2008. Intelligent stock trading system by turning point confirming and probabilistic reasoning. *Expert Systems with Applications* 34, 1 (2008), 620–627.

[4] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259* (2014).

[5] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[6] Sheikh Mohammad Idrees, M Afshar Alam, and Parul Agarwal. 2019. A prediction approach for stock market volatility based on time series data. *IEEE Access* 7 (2019), 17287–17298.

[7] J Kavinnilaa, E Hemalatha, Minu Susan Jacob, and R Dhanalakshmi. 2021. Stock price prediction based on LSTM deep learning model. In *2021 International Conference on System, Computation, Automation and Networking (ICSCAN)*. IEEE, 1–4.

[8] Mahinda Mailagaha Kumbure, Christoph Lohrmann, Pasi Luukka, and Jari Porras. 2022. Machine learning techniques and data for stock market forecasting: A literature review. *Expert Systems with Applications* 197 (2022), 116659.

[9] Jigar Patel, Sahil Shah, Priyank Thakkar, and Ketan Kotecha. 2015. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques. *Expert systems with applications* 42, 1 (2015), 259–268.

[10] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing* 45, 11 (1997), 2673–2681.

[11] Jingyi Shen and M Omair Shafiq. 2020. Short-term stock market price trend prediction using a comprehensive deep learning system. *Journal of big Data* 7, 1 (2020), 1–33.

[12] Justin Sirignano and Rama Cont. 2019. Universal features of price formation in financial markets: perspectives from deep learning. *Quantitative Finance* 19, 9 (2019), 1449–1459.

[13] Bin Weng, Mohamed A. Ahmed, and Fadel M. Megahed. 2017. Stock market one-day ahead movement prediction using disparate data sources. *Expert Systems with Applications* 79 (2017), 153–163. https://doi.org/10.1016/j.eswa.2017.02.041