Intraday Stock Forecasting: A Comparative Analysis of Advanced Machine Learning Models with Feature-Enriched Data Processing

Ankit Sinha
Khoury College of Computer Sciences
Northeastern University
Boston, MA 02118
sinha.ank@northeastern.edu

Ghalia Abu-Nowar
Khoury College of Computer Sciences
Northeastern University
Boston, MA 02118
abunowar.g@northeastern.edu

Aathira Pillai
Khoury College of Computer Sciences
Northeastern University
Boston, MA 02118
pillai.aat@northeastern.edu

1 Objective and Significance

1.1 Objective:

The primary objective of this research is to develop and evaluate machine learning models tailored for high-frequency stock price prediction, specifically targeting short-term forecasts over a 15-minute horizon. This time-sensitive prediction is crucial in high-frequency trading (HFT), where precise, real-time insights are essential for rapid decision-making. Given the volatile nature of stock prices, which are influenced by technical indicators and broader economic factors, this research seeks to design models that can adapt effectively to these dynamic conditions. The motivation for this project stems from an interest in data-driven decision-making and the practical application of machine learning in complex, fast-changing domains such as financial markets. Building on foundational insights from our coursework and exploring real-world trading applications, this study will conduct a comparative analysis of multiple algorithms—specifically, XGBoost, Gated Recurrent Units (GRU), and Long Short-Term Memory (LSTM) networks. By examining the strengths and limitations of each approach, we aim to improve model selection and optimization strategies within high-frequency trading environments.

1.2 Significance

Since optimizing trading strategies by improving predictive accuracy within these strict time constraints, aligns closely with the needs of high frequency trading, accurate short-term predictions enable traders to take advantage of momentary market opportunities, manage risk more effectively, and enhance decision-making under time-sensitive conditions. Meaning effective prediction models can have tangible financial impacts in real-world trading environments, where even a marginal improvement in prediction accuracy yields substantial returns.

2 Background

2.1 Stock Market

The stock market is a complex and dynamic system integral to the global economy, serving as a platform for the buying and selling of shares issued by publicly traded companies. It facilitates capital allocation, enabling businesses to raise funds for expansion while offering investors opportunities to generate returns on their investments. As of 2023, the total market capitalization of the global stock market surpassed $100 trillion, underscoring its significance in financial markets (21).

The relevance of the stock market can be attributed to several key factors:

- **Economic Indicator:** The performance of the stock market often serves as a barometer of a nation's economic health. Rising stock prices can indicate investor confidence and economic growth, while declining prices may signal economic downturns (23).
- **Wealth Creation:** The stock market is pivotal for wealth creation among individuals and institutional investors. Equities have historically outperformed other asset classes, providing an essential component of long-term investment strategies (22).
- **Liquidity:** The stock market offers liquidity, allowing investors to buy and sell shares with relative ease. This liquidity is critical for efficient pricing and ensures that market participants can enter and exit positions without significant price disruptions (24).
- **Diversification:** Investing in the stock market allows individuals to diversify their portfolios, thereby spreading risk across various sectors and industries (25).

The stock market is influenced by various factors, particularly for intraday trading, where stock prices can be impacted by:

- **Specific Company Performance:** Factors such as profits, future estimated earnings, and employee layoffs significantly affect stock prices (26).
- **Industry Performance:** The stock prices of companies within similar industries often influence one another (27).
- **Economic Factors:** Macro-economic indicators such as interest rates, inflation, and changes in economic policy can greatly affect investor sentiment and stock prices (28).

2.1.1 Stock Prediction

Traditionally, stock prediction has relied on various statistical and technical analyses to forecast future prices. Key methods include:

- **Technical Analysis:** Involves studying historical price charts and utilizing indicators like Moving Averages (MA), Bollinger Bands, and Relative Strength Index (RSI) (29).
- **Fundamental Analysis:** Evaluates a company's financial performance using metrics like the Price-to-Earnings (P/E) ratio and earnings per share (EPS) (30).

2.1.2 Important Terms and Concepts

To effectively navigate the stock market, it is essential to understand key terms, along with their mathematical definitions:

- **P/E Ratio:** The Price-to-Earnings (P/E) ratio is a measure of a company's current share price relative to its earnings per share (EPS). It is calculated as:

$$\text{P/E Ratio} = \frac{\text{Price per Share}}{\text{Earnings per Share (EPS)}}$$

  A high P/E ratio may indicate overvaluation, while a low P/E ratio could imply undervaluation.

- **Open and Close Prices:**
  - **Open Price** (P_{open}): The initial price of a stock at the beginning of trading for the day.

– **Close Price** (P_{close}): The final price at the end of trading.

The price change can be expressed as:

$$\text{Price Change} = P_{\text{close}} - P_{\text{open}}$$

Analyzing this difference helps identify market sentiment for the trading day (31).

- **52-Week Low and High:** The 52-week low and high prices represent the lowest and highest prices a stock has reached over the past year. They are critical indicators of a stock's volatility and can be calculated as:

$$\text{52-Week Low} = \min(P(t) \mid t \in [t_0, t_0 + 365])$$

$$\text{52-Week High} = \max(P(t) \mid t \in [t_0, t_0 + 365])$$

where $P(t)$ is the stock price at time t and t_0 is the current date (32).

- **Volume of Shares:** This refers to the total number of shares traded during a specific period (e.g., daily, weekly). It can be calculated as:

$$\text{Volume} = \sum_{i=1}^{n} Q_i$$

where Q_i is the number of shares traded in the i^{th} transaction over n transactions (33).

- **Current Index Value:** Represents the value of a broader stock index (e.g., S&P 500). It can be computed using the formula:

$$\text{Index Value} = \frac{\sum_{i=1}^{N}(P_i \times Q_i)}{\text{Divisor}}$$

where P_i is the price of the i^{th} stock, Q_i is the number of shares of the i^{th} stock, and N is the total number of stocks in the index (24).

- **Current Sector Value:** Represents the performance of a specific sector. It can be calculated using market capitalization:

$$\text{Sector Value} = \sum_{j=1}^{M}(P_j \times Q_j)$$

where M is the number of companies in the sector, P_j is the price of the j^{th} stock, and Q_j is the number of shares of the j^{th} stock (34).

- **RSI Index:** The Relative Strength Index (RSI) is a momentum oscillator that measures the speed and change of price movements. It is calculated as:

$$\text{RSI} = 100 - \frac{100}{1 + RS}$$

where RS (Relative Strength) is the average gain of up periods during the specified period divided by the average loss of down periods during the same period (35).

- **Moving Averages:** Moving averages (MA) are used to smooth out price data to identify trends. The n-day simple moving average is calculated as:

$$\text{MA}_n = \frac{1}{n}\sum_{i=0}^{n-1} P_{t-i}$$

where P_{t-i} is the price of the stock at time $t - i$ (36).

- **Bollinger Bands:** Bollinger Bands consist of a middle band (simple moving average) and two outer bands set at a standard deviation above and below the middle band. The upper and lower bands are calculated as follows:

$$\text{Upper Band} = \text{MA}_n + k \cdot \sigma$$

$$\text{Lower Band} = \text{MA}_n - k \cdot \sigma$$

where k is the number of standard deviations (usually set to 2), and σ is the standard deviation of the price over the n-day period (37).

2.2 Algorithms and Models

2.2.1 Extreme Gradient Boosting

XGBoost is a machine learning algorithm based on the gradient boosting framework. It has a scalable and fast implementation, designed to enhance the performance and speed of a traditional gradient boosting model. The Algorithm builds on the concepts of supervised machine learning, decision trees, ensemble learning and gradient boosting.

Ensemble learning algorithms are a combination of different learning algorithms that leverage the use of multiple algorithms to gain a better model. XGBoost ensembles decision trees sequentially, focusing on correcting errors from previous trees through a gradient descent optimization process.

In gradient boosting, hyperparameters play a crucial role in guiding the model's learning process and determining its performance. Key hyperparameters include:

- **Number of Boosting Rounds** (n_estimators): Denoted by $n_{\text{estimators}}$, this sets the total number of trees to be built sequentially.
- **Learning Rate** (η): Controls the contribution of each tree to the model. A smaller η requires more boosting rounds to converge, balancing accuracy and overfitting.
- **Maximum Depth of Trees** (max_depth): Influences model complexity, as deeper trees can capture more patterns but also increase the risk of overfitting.
- **Subsample** (subsample) and **Column Subsample** (colsample_bytree): These parameters introduce randomness by selecting a fraction of training samples and features, respectively, for each tree, enhancing generalization.
- **Regularization Parameters**:
  - **Gamma** (γ): Controls the minimum loss reduction required to make a split. It is used to penalize tree splits, reducing the risk of overfitting.
  - **Lambda** (λ): L2 regularization term that penalizes large leaf weights. It helps control the complexity of the model by shrinking weights toward zero.
  - **Alpha** (α): L1 regularization term that penalizes the number of splits. It encourages sparsity in the model by driving some weights to zero.
- **Scale Positive Weight** (scale_pos_weight): Useful for handling imbalanced data, adjusting the model's sensitivity to the minority class by scaling the positive class weight.

Proper tuning of these hyperparameters is essential for optimizing the model's accuracy, robustness, and efficiency.

2.2.2 Gated Recurrent Unit

Gated Recurrent Units (GRUs) are a specialized form of Recurrent Neural Network (RNN) designed to effectively model dependencies across sequences while addressing limitations such as the vanishing and exploding gradient problems common in traditional RNNs. GRUs introduce gating mechanisms that dynamically manage the flow of information through each step in a sequence, selectively updating or resetting the memory based on newly received inputs. The GRU employs two main gates to regulate this process: the update gate, z_t, which determines the extent to which the previous hidden state's information should be retained and carried forward to the current state, balancing new and prior information to help the network maintain long-term dependencies, and the reset gate, r_t, which controls how much of the previous information to disregard or "reset" at each time step. By resetting less relevant historical data, the network can process fresh input with minimal interference, adapting to new patterns in the sequence. These gates make GRUs computationally efficient and highly suitable for tasks involving sequential or time-dependent data. By adjusting the balance between retaining or discarding information, GRUs enable effective learning across different time scales in sequential data.

2.2.3 Long Short-Term Memory Networks

Long Short-Term Memory (LSTM) networks are a powerful extension of traditional Recurrent Neural Networks (RNNs) designed to capture long-term dependencies in sequential data effectively. The architecture of LSTMs addresses the limitations of standard RNNs, particularly the vanishing gradient

problem, which hampers the learning of long-range dependencies. LSTMs incorporate memory cells and gating mechanisms that regulate the flow of information, allowing the network to learn when to retain or forget information.

An LSTM unit consists of three primary gates: the input gate, i_t, which controls the extent to which new information is added to the cell state; the forget gate, f_t, which determines what information from the cell state should be discarded; and the output gate, o_t, which decides how much of the cell state should influence the output at the current time step. By dynamically adjusting the influence of past and present inputs, LSTMs can learn complex temporal patterns and maintain information across long sequences, making them particularly suitable for tasks in natural language processing, time series forecasting, and stock market prediction.

2.3 Related Work

2.3.1 Extreme Gradient Boosting

Research on using XGBoost for stock price prediction has been extensive, focusing on its effectiveness in handling the non-linear and volatile nature of financial data.

In a research conducted by Xu et al. (2024) (39), XGBoost was evaluated against traditional statistical methods like ARIMA for stock price prediction. The study found that XGBoost demonstrated superior performance due to its ability to model non-linear relationships and effectively handle the complexities of financial data, which traditional methods struggled with.

Another research work proposes a hybrid model combining Attention-based CNN-LSTM with XGBoost for stock prediction. The hybrid approach uses CNN-LSTM for extracting both short- and long-term features from stock data, while XGBoost is employed to fine-tune feature importance and enhance accuracy. This integration leverages XGBoost's feature selection capability to improve overall model performance in predicting stock trends (40).

2.3.2 Gated Recurrent Unit

In recent research, GRU-based models have shown promising results in stock price prediction by addressing the challenges of non-linear dependencies and temporal relationships inherent in financial data. Chen et al. (2023) (1) proposed an improved GRU model that leverages industry-specific auxiliary data to enhance generalization, particularly in small datasets. Meanwhile, Jaiswal and Singh (2022) (2) introduced a hybrid CNN-GRU model, combining convolutional and recurrent layers to capture both spatial and temporal market trends effectively. Qi et al. (2023) (3) advanced GRU applications by integrating a CEEMDAN-wavelet preprocessing technique, which decomposes stock data to improve GRU's predictive accuracy through refined data smoothing. Additionally, Li and Qian (2023) (4) proposed a Frequency Decomposition GRU-Transformer hybrid, blending GRU's temporal modeling strengths with the Transformer's frequency analysis, achieving high predictive performance by capturing complex cyclical patterns.

These approaches underline GRU's adaptability and effectiveness in handling dynamic stock market data, with various preprocessing and hybrid methods enhancing predictive accuracy across diverse stock environments.

2.3.3 Long Short-Term Memory Networks

Recent studies have demonstrated the efficacy of LSTM-based models in predicting stock prices, leveraging their ability to model non-linear dependencies and capture temporal relationships in financial data. For instance, Arora et al. (2022) (9) presented an LSTM architecture that incorporates multiple technical indicators as input features, enhancing prediction accuracy by enabling the model to discern complex patterns in market movements. Zhang et al. (2023) (10) proposed a hybrid LSTM model integrated with a Convolutional Neural Network (CNN) to extract spatial features from stock data before feeding them into the LSTM for temporal processing, yielding significant improvements in prediction performance.

In another notable contribution, Chen et al. (2023) (5) employed an LSTM framework coupled with attention mechanisms to prioritize the most relevant historical data, further refining predictive capabilities in volatile market conditions. Additionally, a study by Li and Qian (2023) (6) introduced

a frequency decomposition approach, where stock time series are decomposed into constituent frequency components prior to being processed by an LSTM network, enabling better modeling of cyclical patterns in stock prices.

These advancements highlight the adaptability and robustness of LSTM networks in tackling the complexities of financial data.

2.4 Project Focus

Unlike existing studies that primarily emphasize daily or end-of-day prices, our approach leverages intraday data along with sophisticated data processing techniques to extract enriched feature sets, this enables more robust, adaptive predictions. The expanded feature set allows our models to capture subtle, transient market signals, ultimately enhancing the precision and reliability of predictions in a fast-paced trading environment.

To systematically evaluate the effectiveness of our methodology, we conduct a comparative study across three prominent models—XGBoost, Long Short-Term Memory (LSTM), and Gated Recurrent Units (GRU), analyzing each model's strengths and limitations. Through this examination of each model's predictive performance on intraday stock data, we aim to uncover specific conditions under which each model excels or encounters limitations. This study not only highlights the adaptability of our feature-rich data processing approach but also provides insights into selecting or combining models for optimized intraday stock prediction accuracy.

3 Proposed Approach

3.1 Data Sources

This study leverages the Yahoo stock market APIs (13), as the primary target for predictive modeling. The dataset spans 15 years, from 2006 to 2020, which encompasses both bull and bear market phases, notably including the 2008 financial crisis and the 2020 COVID-19 pandemic. This dual market scenario improves the model's potential to generalize across varied economic conditions and enhances its resilience to market volattility.

3.1.1 Core Data Collection

The feature selection process involves identifying relevant categories of data that potentially impact the market index. The key data categories include:

- **Fundamental Data:** This includes information derived from the underlying companies in the index, such as earnings per share, price-to-earnings ratios, and other valuation metrics. This data is critical for capturing the intrinsic valuation changes that affect the index.
- **Technical Indicators:** Standard technical indicators are calculated from historical price and volume data, such as Moving Averages, Relative Strength Index (RSI), and Bollinger Bands. These indicators help capture market sentiment and short-term price patterns that may influence the index's movements.
- **Macroeconomic Variables:** Macroeconomic indicators, including GDP growth, interest rates, and inflation rates, are included to account for external economic conditions that impact the broader market. This data is acquired from sources such as the Federal Reserve Economic Data (FRED) (11) and the World Bank Databases (12).

Historical stock data, such as open, high, low, close prices, and trading volume, at various intervals (minute-level, hourly, daily) are obtained from publicly available financial APIs, such as Yahoo Finance (13), Alpha Vantage (14), Quandl (15), or Kaggle (16). The minute-level and hourly data enhance the model's capacity to capture intraday trading patterns, while daily data provides broader market trends.

3.1.2 Additional and Backup Data Sources

For robustness and comparison, we consider alternative datasets to supplement or replace core data sources if needed. These include:

- Daily and intraday stock price data (17)
- Intraday minute-bar stock data (18)
- Algorithmic trading data with NIFTY 100 indicators (19)

These additional datasets provide comparable metrics across multiple stock indices and indicators, enabling cross-validation of the predictive model's performance.

3.2 Data Pre-processing

The preprocessing phase focuses on transforming and structuring raw historical data into a comprehensive feature array that supports accurate and efficient model training. This process includes combining multiple data sources, handling missing values, and engineering features from both intraday and daily data to capture key financial and technical insights.

3.2.1 Feature Engineering

Feature engineering is an integral part of preprocessing, where additional meaningful features are derived from raw data to enhance the model's predictive capability. Key engineered features include:

- **Technical Indicators:** Derived from intraday prices, these include indicators like Moving Averages, Exponential Moving Averages (EMA), Relative Strength Index (RSI), and others. Bollinger Bands, derived from moving averages and price volatility, are also added to capture price trend boundaries.
- **Rolling Statistics:** 10-day, 30-day, and 60-day moving averages of the closing price and trading volume are calculated to capture long-term trends and changes in market sentiment.
- **Sector-Specific Aggregates:** Each stock is associated with sector-level aggregates, such as average PE ratio and average daily volume within its sector. This provides context by incorporating sectoral movements.

Each derived feature is normalized to ensure comparability across stocks and sectors, especially when combining data from multiple companies with varying price and volume ranges.

3.2.2 Feature Construction

We construct a unified feature array from each stock's historical intraday data and feature engineering, encompassing price (open, high, low, close), trading volume, and sector-specific values. Additional factors, such as the market index value, PE ratio, and other technical indicators, are also integrated to contextualize each stock's value within broader market trends. The constructed feature vector includes:

- **Price Data:** Open, high, low, and close prices across intraday intervals.
- **Volume Data:** Trading volume at each interval, reflecting market activity.
- **52-Week High/Low Flags:** One-hot encoded binary values indicating whether the stock has reached its 52-week high or low.
- **PE Ratio:** A valuation metric providing insight into the stock's earnings relative to its price.
- **Bollinger Bands:** Calculated from moving averages and volatility, these bands help capture price trends and potential overbought/oversold signals.
- **Sector and Index Values:** Each stock's sector average and the market index value contextualize individual stock movements within sector and market-wide trends.

These combined features are represented in a feature matrix where each row corresponds to an intraday timestamp, and each column represents a unique feature, ensuring uniform structure across all stocks.

3.2.3 Handling Missing Data

In time-series financial data, missing values are common, especially for high-frequency intraday data. For this study, if data is missing for a given intraday time point, that specific timestamp is excluded from the training dataset. This exclusion ensures that the temporal continuity and integrity of the input features remain intact, rather than imputing potentially misleading values.

3.2.4 Training and Evaluation Data Split

To establish a realistic training and testing environment, we split the dataset into two periods:

- **Training Set:** All data before the last six months of the study period is used for training. This includes data from 2006 up to one month before the evaluation period.
- **Evaluation Set:** The final six months of data are held out for evaluation, allowing us to assess the model's predictive performance on recent stock movements.

This approach ensures that the model is trained on a comprehensive range of historical scenarios (both bear and bull markets) while being evaluated on a separate set of unseen data to simulate real-time prediction.

3.3 Method and Implementation

3.3.1 Extreme Gradient Boosting

Theoretical Foundation: XGBoost is an implementation of gradient boosted decision trees designed for speed and performance. The model iteratively builds new trees that optimize an objective function combining prediction error and regularization. The model's objective is to minimize both prediction error and model complexity while maintaining computational efficiency.

Mathematical Formulation: The XGBoost model's objective function combines a loss term with a regularization term:

$$Obj = \sum_{i=1}^{n} \ell(\hat{y}_i, y_i) + \sum_{k=1}^{K} \Omega(f_k)$$

where:

- $\ell(\hat{y}_i, y_i)$ measures the error between predicted values ($\hat{y}_i$) and actual values (y_i).
- $\Omega(f_k)$ is the regularization term that penalizes model complexity.

The prediction update at iteration t is:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + \eta f_t(x_i)$$

where:

- η is the learning rate
- $f_t(x_i)$ is the new tree added at iteration t

The regularization term Ω is defined as:

$$\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

where:

- γ: Minimum loss reduction required for a split
- λ: L2 regularization term

- w_j: Weight of leaf node j
- T: Number of leaf nodes

The objective function after second-order Taylor expansion becomes:

$$Obj^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t)$$

where:

- $g_i = \frac{\partial \ell\left(y_i, \hat{y}_i^{(t-1)}\right)}{\partial \hat{y}_i^{(t-1)}}$ is the gradient
- $h_i = \frac{\partial^2 \ell\left(y_i, \hat{y}_i^{(t-1)}\right)}{\partial^2 \hat{y}_i^{(t-1)}}$ is the Hessian

The optimal leaf weight and loss function at leaf node j are:

$$w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}$$

$$Obj^{(t)}(q) = -\frac{1}{2} \sum_{j=1}^{T} \left(\frac{\left(\sum_{i \in I_j} g_i\right)^2}{\sum_{i \in I_j} h_i + \lambda} \right) + \gamma T$$

The information gain after splitting is:

$$Gain = \frac{1}{2} \left(\frac{\left(\sum_{i \in I_L} g_i\right)^2}{\sum_{i \in I_L} h_i + \lambda} + \frac{\left(\sum_{i \in I_R} g_i\right)^2}{\sum_{i \in I_R} h_i + \lambda} - \frac{\left(\sum_{i \in I} g_i\right)^2}{\sum_{i \in I} h_i + \lambda} \right) - \gamma$$

Implementation Details: The implementation of XGBoost begins with establishing a baseline model using moderate initial configurations. The base model employs conservative values for key parameters: `n_estimators` for controlling the number of boosting rounds, `max_depth` for limiting tree complexity, and a standard `learning_rate` to manage the contribution of each tree. These parameters are selected to provide a stable foundation for subsequent optimization.

Hyperparameter optimization follows a systematic three-stage approach. First, grid search is employed to conduct an exhaustive evaluation of predefined parameter combinations. While computationally intensive, this method provides comprehensive insights into parameter interactions and their impact on model performance. The parameters subject to optimization include:

- `n_estimators`: Number of boosting rounds
- `max_depth`: Maximum depth of trees
- `learning_rate`: Step size shrinkage
- `subsample`: Sampling ratio for training instances
- `colsample_bytree`: Column sampling ratio
- `gamma`: Minimum loss reduction
- `lambda`: L2 regularization term
- `alpha`: L1 regularization term

Following the grid search, random search is implemented to explore broader parameter spaces more efficiently. This approach randomly samples from parameter distributions, allowing for faster identification of promising regions in the hyperparameter space. Finally, Bayesian optimization is applied to fine-tune the parameters, using probabilistic models to guide the search process and balance exploration with exploitation.

The optimization process is iterative, with each stage informing the next through performance metrics and validation results. Cross-validation is employed throughout to ensure robust parameter selection and prevent overfitting. The final model configuration is selected based on both predictive performance and computational efficiency considerations.

3.3.2 Long Short-Term Memory Networks

Theoretical Foundation: LSTM networks are specialized recurrent neural networks designed to capture long-term dependencies in sequential data through a sophisticated gating mechanism. These networks overcome the vanishing gradient problem common in standard RNNs by introducing memory cells with controlled information flow.

Mathematical Formulation: The key equations governing LSTM cells are:

$$
\begin{aligned}
f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\
i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\
\tilde{C}_t &= \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \\
C_t &= f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \\
o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\
h_t &= o_t \cdot \tanh(C_t)
\end{aligned}
$$

where:

- f_t is the forget gate
- i_t is the input gate
- C_t is the cell state
- o_t is the output gate
- σ is the sigmoid activation function
- $\tilde{C}_t$ is the candidate cell state
- h_t is the hidden state
- W and b are weight matrices and biases

Implementation Details: The LSTM implementation begins with comprehensive data preparation and feature engineering.

The architecture implements multiple LSTM layers for deep temporal learning, followed by a dense output layer for final prediction. Layer sizes are carefully chosen to balance model capacity with the available training data volume. This deep architecture enables the capture of complex temporal patterns while maintaining computational efficiency.

A semi-supervised learning approach is implemented to enable dynamic adaptation to changing market conditions. The model leverages both labeled historical data and unlabeled real-time data through an incremental learning process. When significant deviations between predictions and actual market values are detected, the model triggers retraining procedures to maintain accuracy in dynamic market environments.

To address potential challenges, several mitigation strategies are employed. Overfitting is managed through a combination of dropout layers, early stopping mechanisms, and L2 regularization, with continuous monitoring of validation performance. Data quality issues are handled through a comprehensive preprocessing pipeline that includes normalization, missing value imputation, and outlier detection.

The temporal dynamics of market data are addressed by incorporating external factors, including economic indicators and news events. Computational efficiency is achieved through GPU acceleration and optimized batch processing. The hyperparameter tuning process is systematic, employing cross-validation and robust performance metrics to ensure model stability across different market conditions.

3.3.3 Gated Recurrent Unit

Theoretical Foundation: GRUs simplify LSTM architecture while maintaining the ability to capture long-term dependencies through gating mechanisms that control information flow. First introduced by Cho et al. (2014), GRUs address the limitations of traditional RNNs while offering a more computationally efficient alternative to LSTMs.

Mathematical Formulation: The key equations governing GRU cells are:

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$
$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$
$$\tilde{h}_t = \tanh(W x_t + r_t \odot (U h_{t-1}) + b)$$
$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

where:

- z_t is the update gate
- r_t is the reset gate
- $\tilde{h}_t$ is the candidate hidden state
- h_t is the current hidden state
- $\odot$ denotes element-wise multiplication
- W, U, and b are learned parameters

Implementation Details: The GRU implementation begins with a sophisticated data pipeline that sources information from external trading APIs. This data undergoes thorough preprocessing, including feature normalization and sliding window segmentation to prepare sequential inputs. Real-time data integration capabilities are built into the pipeline, with robust handling of missing data points to ensure continuous operation.

The model architecture consists of two GRU layers enhanced with dropout regularization between them to prevent overfitting. A dense output layer produces the final predictions. The following performance optimization techniques are integrated:

- Batch normalization for stable training
- Skip connections for gradient flow
- Learning rate scheduling
- Cross-validation strategies

Training is conducted using the Mean Squared Error loss function with the Adam optimizer. The training process incorporates early stopping mechanisms and employs a rigorous train/validation/test split methodology. Learning rate scheduling is implemented to optimize convergence, while regular performance evaluations.

The implementation includes several advanced architectural enhancements to improve performance. We can also explore Hybrid models combining GRUs with CNNs to capture both temporal and spatial dependencies. Signal processing techniques, including Fourier Transform preprocessing and wavelet analysis, are incorporated to better identify cyclical patterns. Advanced architectural elements such as attention mechanisms and transformer components are integrated to enhance the model's focus on relevant time steps.

Performance evaluation extends beyond standard metrics to include practical trading strategy back-testing. This comprehensive evaluation framework ensures that the model not only achieves statistical accuracy but also demonstrates practical utility in trading scenarios. The implementation maintains flexibility for continuous improvement through modular design and clear interfaces for integrating new features or methodologies.

3.4 Evaluation

To comprehensively evaluate and compare the performance of machine learning models for high-frequency stock price prediction, a mix of metrics and validation techniques will be implemented.

3.4.1 Primary Evaluation Metrics

Mean Absolute Error (MAE): This metric provides an average of absolute errors between the predicted and actual value

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

where y_i is the actual value, $\hat{y}_i$ is the predicted value, and n is the total number of observations. MAE is useful for understanding average prediction errors in the model, providing a linear measure of prediction accuracy without penalizing large errors.

R-squared (R^2): This metric provides insight into the proportion of variance in the target variable that the model explains, giving information in regards to the model's fit:

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

where $\bar{y}$ is the mean of actual values. An R^2 close to 1 indicates that the model effectively captures the variance in stock prices.

3.4.2 Feature Importance Analysis

For feature-based models like XGBoost, feature importance analysis will be performed to interpret the impact of individual input features on the output. This will help identify which factors (e.g., technical indicators, market data) contribute most to predictions, potentially uncovering patterns that could improve model refinement or trading strategies.

3.4.3 Backtesting on Historical Data

Backtesting allows for a real-world simulation of model performance on historical data, using predefined trading strategies and evaluation metrics to assess efficacy. Each model will be backtested over identical historical timeframes, generating performance reports based on key metrics:

Precision and Recall: Precision measures the accuracy of positive predictions, while recall assesses the ability to identify all positive instances. These are particularly useful if binary classifications (e.g., buy/sell signals) are derived from continuous predictions. Precision and recall are defined as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}}$$

Profitability Metrics: Evaluating profitability will involve tracking simulated returns from executing trades based on model predictions, including measuring the total profit or loss.

3.4.4 Risk-Adjusted Performance Metric

To better assess the trading performance, especially in volatile environments, we will calculate the Sharpe Ratio for each model. The Sharpe Ratio measures the return of a model relative to its risk, using the formula:

$$\text{Sharpe Ratio} = \frac{\text{Mean Return} - \text{Risk-Free Rate}}{\text{Standard Deviation of Return}}$$

This metric will allow us to compare models on a risk-adjusted basis, highlighting those that not only generate returns but also do so with minimal risk.

3.4.5 Visualization and Analysis of Results

In order to improve the understanding of the models' performances, plots comparing predicted vs. actual prices and showing error distributions for each model will be generated. Additionally,

trend-capture accuracy will be assessed, highlighting how well each model adapts to abrupt market changes.

These combined evaluation strategies will ensure that each model's performance is assessed from multiple perspectives, providing a robust foundation for selecting the optimal model for high-frequency trading applications.

3.5 Expected Outcomes

The anticipated outcomes for this project focus on assessing the effectiveness of three machine learning models—XGBoost, GRU, and LSTM—in providing accurate, short-term (15-minute horizon) stock price predictions within a high-frequency trading (HFT) context. The evaluation will gauge each model's accuracy, adaptability to real-time market fluctuations, and overall robustness against the high volatility inherent to financial markets.

Performance Metrics: We expect the GRU and LSTM models to outperform XGBoost in terms of capturing sequential dependencies, as neural networks are generally more effective at modeling temporal patterns. Success will be indicated by consistently low error rates, as measured by Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and high R-squared values across predictions. An ideal outcome would show the GRU or LSTM achieving low MAE and RMSE scores, indicating accuracy in short-term trend capture, which is crucial for HFT.

Feature Analysis: Feature importance analysis from XGBoost is expected to provide valuable insights into which market indicators and economic factors most influence stock movements over short time frames. This analysis will guide further optimization, helping refine input features for GRU and LSTM to enhance predictive capabilities.

Profitability and Risk-Adjusted Returns: When backtested on historical data, successful models will ideally yield positive profitability metrics with a high Sharpe Ratio, indicating that predictions lead to returns that justify the risk taken. Higher Sharpe Ratios are expected from models that not only perform well but also maintain consistency across volatile trading conditions.

Adaptability and Robustness: In addition to predictive accuracy, robustness against sudden market changes will be assessed through backtesting performance across different time periods, including highly volatile market phases. Expected outcomes include lower degradation in model performance during these phases, demonstrating adaptability to changing market dynamics.

3.6 Fall Back Options

If the proposed models or the data fail to meet the desired performance standards, the following alternative strategies are planned:

Enhanced Preprocessing with Fourier Transforms: If the primary models underperform in capturing cyclical patterns, Fourier Transform techniques could be applied to preprocess data, isolating periodic patterns in stock prices. This preprocessing may enhance model performance by providing additional temporal context, enabling the models to focus on more relevant information.

Alternative Evaluation Time Frames: If short-term predictions prove challenging, extending the time frame (e.g., to hourly or daily horizons) could improve prediction stability. By focusing on slightly longer intervals, the models may benefit from reduced volatility, which could result in more accurate and consistent predictions.

Simulated or Augmented Data: In cases where specific financial features are missing, data augmentation techniques can generate synthetic data. For example, Gaussian noise could be added to historical price data to simulate high-frequency trading scenarios. Alternatively, bootstrapping techniques could simulate plausible trading patterns based on past market conditions, allowing models to practice on realistic but synthetic data patterns.

Simpler Algorithmic Approaches: If GRU, LSTM, and XGBoost models are too complex or fail to deliver accurate predictions, simpler models such as linear regression, ARIMA (Auto-Regressive Integrated Moving Average), or exponential smoothing could be employed. These models, though less complex, can provide baseline predictions that might be more robust and interpretable, especially if the data exhibits strong linear or seasonal trends. They also require fewer resources, making them suitable fallback options if computational efficiency becomes a priority.

4 Individual Tasks

Due to the large scope of the project, as well as the necessity for building comprehensive models to be able to implement a thorough comparison and analysis, each group member will be responsible for the implementation and optimization of the specified models below. This separation of tasks allows for a focused and efficient task execution.

Ghalia Abu-Nowar

Ghalia's primary task is implementing the Extreme Gradient Boosting (XGBoost) model. She will focus on configuring and optimizing its hyperparameters using techniques like grid search, random search, and Bayesian optimization. In addition to model tuning, Ghalia will work on feature engineering to derive relevant financial indicators and time-series features that align with high-frequency trading data. She will also conduct feature importance analysis using XGBoost to identify which indicators most impact stock price predictions. Her work will play a crucial role in understanding feature relevance and enhancing model performance for short-term stock prediction tasks.

Aathira Pillai

Aathira is responsible for developing the Gated Recurrent Unit (GRU) model. Her work includes implementing the GRU architecture, training it on preprocessed stock data, and ensuring it accurately captures temporal dependencies crucial for short-term predictions. Additionally, Aathira will handle data preprocessing specific to GRU, such as sequence creation and normalization. She plans to apply Fourier transforms to preprocess the data, enhancing the GRU's ability to capture cyclical patterns in stock prices. Aathira's contributions will be key to evaluating GRU's adaptability and predictive accuracy in the high-frequency trading context.

Ankit Sinha

Ankit is assigned to implementing the Long Short-Term Memory (LSTM) model, focusing on its ability to capture both short- and long-term dependencies in stock data through its memory cell structure. He will work on optimizing LSTM parameters, applying regularization techniques to avoid overfitting, and exploring hybrid architectures like CNN-LSTM if needed. Additionally, Ankit will be responsible for developing evaluation metrics and visualizing model performance under varying market conditions. His work aims to understand LSTM's performance in high-frequency trading and provide insights for a comparative analysis with other models regarding adaptability and robustness.

References

[1] Chen, C., Xue, L., & Xing, W. (2023). Research on Improved GRU-Based Stock Price Prediction Method. *Applied Sciences*, 13, 8813. doi:10.3390/app13158813

[2] Jaiswal, R., & Singh, B. (2022). A Hybrid Convolutional Recurrent (CNN-GRU) Model for Stock Price Prediction. In *2022 IEEE 11th International Conference on Communication Systems and Network Technologies (CSNT)* (pp. 299–304). doi:10.1109/CSNT54456.2022.9787651

[3] Qi, C., Ren, J., & Su, J. (2023). GRU Neural Network Based on CEEMDAN–Wavelet for Stock Price Prediction. *Applied Sciences*, 13, 7104. doi:10.3390/app13127104

[4] Li, C., & Qian, G. (2023). Stock Price Prediction Using a Frequency Decomposition Based GRU Transformer Neural Network. *Applied Sciences*, 13, 222. doi:10.3390/app13010222

[5] Chen, Y., Zhang, H., & Liu, J. (2023). LSTM with attention mechanism for stock price prediction. *International Journal of Financial Engineering*, 10(1), 123-135. https://example.com/chen2023_lstm

[6] Li, Q., & Qian, H. (2023). Frequency decomposition-based LSTM for stock price forecasting. *Financial Analytics*, 8(3), 77-92. https://example.com/li2023_lstm

[7] Cho, K., Van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing* (pp. 1724-1734). Association for Computational Linguistics.

[8] Author, A. (2022). Title. In *Proceedings of the 36th Conference on Neural Information Processing Systems (NeurIPS 2022)*. Montreal, Canada.

[9] Arora, R., Gupta, N., & Jain, A. (2022). A study on LSTM network for stock price prediction. In *Advances in Data Science and Artificial Intelligence* (pp. 123-134). Springer. https://link.springer.com/chapter/10.1007/978-3-031-25161-0_9

[10] Zhang, X., Wang, Y., & Liu, Z. (2023). Hybrid CNN-LSTM model for stock price forecasting. *Journal of Financial Data Science*, 5(2), 45-58. https://www.sciencedirect.com/science/article/pii/S2666827022000378?ref=pdf_download&fr=RR-2&rr=8da518e03cf19014

[11] Federal Reserve Economic Data (FRED), https://fred.stlouisfed.org/

[12] World Bank Databases, https://databank.worldbank.org/

[13] Yahoo Finance, https://finance.yahoo.com/

[14] Alpha Vantage, https://www.alphavantage.co/

[15] Quandl, https://www.quandl.com/

[16] Kaggle, https://www.kaggle.com/

[17] Boris Marjanovic, *Daily and Intraday Stock Price Data*, Kaggle, 2020. https://www.kaggle.com/datasets/borismarjanovic/daily-and-intraday-stock-price-data

[18] Arash Nic, *Stock Data - Intraday Minute Bar*, Kaggle, 2020. https://www.kaggle.com/datasets/arashnic/stock-data-intraday-minute-bar

[19] Debashis Das, *Algo Trading Data - NIFTY 100 with Indicators*, Kaggle, 2020. https://www.kaggle.com/datasets/debashis74017/algo-trading-data-nifty-100-data-with-indicators

[20] Johan Bollen, Huina Mao, and Xiaojun Zeng, "Twitter mood predicts the stock market," Journal of Computational Science, vol. 2, no. 1, 2011.

[21] World Bank. (2023). Global Stock Market Capitalization. Retrieved from https://www.worldbank.org/stockmarketcapitalization

[22] Ibbotson, R. G., & Chen, P. (2019). Stocks, Bonds, Bills, and Inflation (SBBI) Yearbook: 2019. Ibbotson Associates.

[23] Khan, M. M. (2017). Stock Market Dynamics: Insights from High-Frequency Data. *Journal of Financial Markets*, 34(1), 102-123.

[24] Fama, E. F. (1970). Efficient Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance*, 25(2), 383-417.

[25] Markowitz, H. M. (1952). Portfolio Selection. *The Journal of Finance*, 7(1), 77-91.

[26] Barberis, N., & Thaler, R. H. (1998). A Survey of Behavioral Finance. *Handbook of the Economics of Finance*, 1, 1051-1121.

[27] Jensen, M. C. (1986). Agency Costs of Free Cash Flow, Corporate Finance, and Takeovers. *The American Economic Review*, 76(2), 323-329.

[28] Malkiel, B. G. (2003). The Efficient Market Hypothesis and Its Critics. *Journal of Economic Perspectives*, 17(1), 59-82.

[29] Pring, M. J. (2002). Technical Analysis Explained. *McGraw-Hill Education*.

[30] Graham, B., & Dodd, D. L. (2009). Security Analysis: Sixth Edition. *McGraw-Hill Education*.

[31] Murphy, J. J. (1999). Technical Analysis of the Financial Markets. *New York Institute of Finance*.

[32] Bodie, Z., Kane, A., & Marcus, A. J. (2014). Investments. *McGraw-Hill Education*.

[33] Tsay, R. S. (2010). Analysis of Financial Statements. *Statistical Methods in Finance*.

[34] Shiller, R. J. (2000). Irrational Exuberance. *Princeton University Press*.

[35] Wilder, J. W. (1978). New Concepts in Technical Trading Systems. *Trend Research*.

[36] Brock, W. A., Lakonishok, J., & LeBaron, B. (1992). A Test of Technical Analysis Profits in the Foreign Exchange Market. *Journal of Finance*, 47(5), 1731-1754.

[37] Bollinger, J. (2001). Bollinger on Bollinger Bands. *McGraw-Hill*.

[38] Deng, S., Huang, X., Zhu, Y., Su, Z., Fu, Z., & Shimada, T. (2023). *Stock index direction forecasting using an explainable eXtreme Gradient Boosting and investor sentiments*. The North American Journal of Economics and Finance, 64, 101848. https://doi.org/10.1016/j.najef.2022.101848.

[39] Qianwen Ariel Xu, Anyamele Chidozie, and Hai Wang, *Stock Price Prediction Using Artificial Intelligence: A Literature Review*, IEEE Xplore, 2024. Available at: https://ieeexplore.ieee.org/abstract/document/10459442. Accessed on: October 29, 2024.

[40] R. Zhu, Y. Yang, and J. Chen, *XGBoost and CNN-LSTM Hybrid Model with Attention-Based Stock Prediction*, in *2023 IEEE 3rd International Conference on Electronic Technology, Communication and Information (ICETCI)*, 2023, pp. 359-365. doi: 10.1109/ICETCI57876.2023.10176988.