VIETNAM NATIONAL UNIVERSITY, HANOI

INTERNATIONAL SCHOOL

GRADUATION PROJECT

FORECASTING STOCK PRICE MOVEMENTS OF VN30 CONSTITUENTS BASED ON MACHINE LEARNING METHOD AND SENTIMENT ANALYSIS

Vu Tuyen Hoang

Hanoi - Year 2025

VIETNAM NATIONAL UNIVERSITY, HANOI

INTERNATIONAL SCHOOL

GRADUATION PROJECT

FORECASTING STOCK PRICE MOVEMENTS OF VN30 CONSTITUENTS BASED ON MACHINE LEARNING METHOD AND SENTIMENT ANALYSIS

SUPERVISOR: ASSOC. PROF., PHD. TRAN THI NGAN

(Academic title, academic degree, full name)

STUDENT: VU TUYEN HOANG

STUDENT ID: 21070141

COHORT: QHQ2021

SUBJECT CODE: INS401101

MAJOR: BUSINESS DATA ANALYTICS

Hanoi - Year 2025

ACKNOWLEDGEMENT

During my thesis, I received a lot of help and enthusiastic support, which motivated and guided me to overcome all obstacles and successfully complete this thesis. First of all, I would like to thank the teachers at the International School, Vietnam National University, and sincerely thank them for supporting me in completing this thesis. In addition, I am especially thankful to my thesis supervisor, Assoc. Prof., PhD. Tran Thi Ngan with valuable advice, and insightful feedback were very helpful to every stage of my research. Her dedicated support and patience have helped me refine my work and greatly enhance its quality.

At the same time, I would like to thank other lecturers in the Faculty of Applied Sciences who have provided useful knowledge and essential skills throughout my academic years. Next, I would like to send my sincere thanks to my friends and the online communities, especially HuggingFace, GitHub, Stack Overflow, and other academic forums, for directly supporting and helping me with technical issues, valuable suggestions, and sharing many important resources throughout my thesis project.

Because the research period was limited, my initial steps into practical research were challenging, my knowledge was still very limited, and due to many objective reasons, shortcomings are certainly unavoidable. I sincerely look forward to receiving feedback from teachers, friends, and experts from online communities throughout my learning and understanding journey. Finally, I am extremely grateful to my family for their endless love, encouragement, and unwavering support, enabling me to confidently complete this important thesis.

GUARANTEE

I confirm that this thesis is all my own original work and shows my own research and efforts. All the findings, analysis, and explanations in this thesis are done with the guidance and help of Assoc. Prof., PhD. Tran Thi Ngan. Also, I make sure that I have cited and acknowledged clearly all the outside sources that I used in this thesis following the academic rules and standards. I take full responsibility for the truth, correctness, and honesty of everything presented in this thesis. I also follow all the academic requirements and rules of the International School, Vietnam National University.

Student,

Vu Tuyen Hoang

LIST OF ACRONYMS

No	Abbreviations	Script
1	AI	Artificial Intelligence
2	API	Application Programming Interface
3	ARIMA	AutoRegressive Integrated Moving Average
4	BERT	Bidirectional Encoder Representations from Transformers
5	BiGRU	Bidirectional Gated Recurrent Unit
6	CPU	Central Processing Unit
7	DL	Deep Learning
8	EMA	Exponential Moving Average
9	ETS	Exponential Smoothing
10	GRU	Gated Recurrent Unit
11	HOSE	Ho Chi Minh City Stock Exchange
12	JSON	JavaScript Object Notation
13	LSTM	Long Short-Term Memory
14	MAE	Mean Absolute Error
15	MAPE	Mean Absolute Percentage Error
16	ML	Machine Learning
17	MSE	Mean Squared Error
18	OHLC	Open-High-Low-Close
19	R^2	R-squared
20	RNN	Recurrent Neural Network
21	RMSE	Root Mean Square Error
22	RSI	Relative Strength Index
23	SSI	Saigon Securities Incorporation
24	TPE	Tree-structured Parzen-Estimator
25	VN30	Vietnam 30 Index
26	XGBoost	Extreme Gradient Boosting

LIST OF FIGURES

LIST OF TABLES

TABLE OF CONTENTS

INTRODUCTION

Forecasting stock price movements is an important topic in financial analytics, especially in developing markets like Vietnam where the market is often affected by high volatility and external economic factors. Among various indices reflecting market performance, the VN30 index, which includes the 30 most liquid and large-cap stocks listed on the Ho Chi Minh City Stock Exchange (HOSE), is considered a key benchmark. These 30 stocks represent around 80% of the total market capitalization and more than 60% of the trading volume on HOSE, making the VN30 index a reliable sample for studying investor behavior and price fluctuations in the Vietnamese stock market (Nguyen *et al.*, 2021). This makes the VN30 index a strong representative and indicator for analyzing investor behavior and stock price trends in the Vietnamese market.

Traditional time-series models such as ARIMA and Exponential Smoothing (ETS) have been commonly used to predict stock prices in Vietnam and have shown fairly good results. In particular, a study using ARIMA to forecast both daily and monthly prices of VN30 stocks found that predicted values usually stayed within the confidence intervals, and ARIMA provided better RMSE than ETS, confirming its reliability in this market context (Nguyen and Tetiana, 2024).

However, there are still important research gaps that need to be addressed. First, although these traditional models are useful, most studies have only tested them on a small number of stocks or for a limited period, and very few have offered a full comparison between different forecasting models using a unified dataset. Second, while features such as sentiment scores from financial news have become popular in global stock prediction research, their role in the Vietnamese market, especially when integrated into an automated forecasting system, has not been clearly studied.

In addition, many existing studies only focus on predicting the direction of stock price movement, such as whether it will go up or down, instead of forecasting the actual future price. Very few of them provide a complete system that can generate predicted prices for all VN30 stocks, update the results daily, and visualize them in a dashboard that supports decision making. These limitations present an opportunity for a more comprehensive and practical approach to apply in real-life conditions.

Aims to address these gaps, this thesis presents a fully automated forecasting system designed for all stocks in the VN30 index. The pipeline includes collecting the newest list of stocks, historical stock price data and financial news, then, preprocessing the data, generating both technical and sentiment-based features, and training various forecasting models. These models include both machine learning and deep learning techniques, and their performance is evaluated using metrics such as R-squared (R^2), Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE).

The entire system is automated using Apache Airflow, allowing data to be updated and predictions to be generated every day. Forecast results are future prices for each VN30 stock for the next 30 trading days from now. The results are displayed through a dashboard built with Power BI. In addition, the dashboard showed some charts using historical data to bring insights to users. One more important thing that the dashboard is automatically refreshed daily using Power BI Gateway to ensure that users always see the most recent data and forecasts.

This thesis contributes not only by comparing different forecasting models on real data from the Vietnamese market, but also by building a complete, end-to-end system that connects data collection, modeling, automation, and visualization. This solution can serve both academic research and real-world financial forecasting applications.

Chapter 1 THEORETICAL BACKGROUND AND LITERATURE REVIEW

1.1. Research context

In Vietnam's fast-growing financial market, there are more and more investors enter the stock market. So that, the need for accurate stock forecasting tools is becoming more important. As the economy develops and more investors join the market, especially individual and retail investors, stock data becomes more complex and harder to predict. The VN30 index plays a crucial role in the whole market performance. As mentioned above, this index includes the 30 largest and most liquid companies listed on the Ho Chi Minh City Stock Exchange (HOSE), accounting for about 80% of the market capitalization and over 60% of the daily trading volume (Nguyen *et al.*, 2021). Because of its representativeness, the VN30 index is widely used as a benchmark for evaluating market movements and building predictive models.

1.2. Theoretical background

Time series forecasting is a popular technique in financial analysis. It helps investors and researchers estimate future prices based on historical patterns. Among the earliest models used for this purpose are ARIMA and Exponential Smoothing (ETS). ARIMA was introduced by George Box and Gwilym Jenkins in the 1970s. It is known for its ability to capture short-term linear trends and seasonal effects in static data. On the other hand, ETS, developed by Hyndman and others, provides a flexible way to forecast trended and seasonal data, even if the data is dynamic.

Although these traditional models perform well in stable conditions, they often struggle with sudden market changes or non-linear behaviors, which are characteristics of emerging markets like Vietnam. To handle these limitations, many studies have explored other machine learning (ML) and deep learning (DL) models. ML models such as Random Forest, XGBoost, and LightGBM can handle large amounts of data and identify complex patterns without requiring strong assumptions about the data's structure. DL models, including Long Short-Term Memory (LSTM) and Bidirectional Gated Recurrent Unit (BiGRU), are designed to learn from sequences, which makes them useful for stock forecasting, where past movements can influence future trends.

In addition to historical prices and technical indicators, financial news and investor sentiment also affect stock prices. Sentiment analysis allows us to extract positive, negative, or neutral signals from news articles and use them as inputs for forecasting models. This is especially helpful in markets like Vietnam, where investor confidence can shift quickly in response to events. Thanks to Vietnamese language models like PhoBERT, it is now possible to extract sentiment from Vietnamese financial texts with good accuracy and use these scores as part of a forecasting system.

1.3. Literature review

The use of traditional forecasting methods in finance dates back to the early 2000s. (Hyndman *et al.*, 2002) introduced a complete ETS modeling framework that could handle level, trend, and seasonality without differencing. This approach became a strong competitor to ARIMA in many time series applications.

Specifically for Vietnamese stock market, (Nguyen and Tetiana, 2024) applied ARIMA and ETS to daily and monthly VN30 index. Their study showed evidence that ARIMA consistently outperformed ETS in short-term forecasting. But, both models struggled with sudden market fluctuations. Following this, (Hongyu, 2025) conducted a comparative study using both ARIMA and ETS to forecast VN30 index values from 2016 to 2023. The findings showed that ARIMA provided more accurate results in terms of RMSE, while both models were effective in producing reliable forecast intervals.

As more financial data became available, machine learning models gained popularity. (Krauss, Do and Huck, 2017) tested some models including Random Forest, XGBoost, and deep neural networks on US stock returns. Their study revealed that XGBoost delivered the best performance among all models and outperformed traditional techniques such as ARIMA. And as a development from the previous research, in 2018, (Fischer and Krauss, 2018) extended this work by applying LSTM to the S&P 500 index, which tracking the stock performance of top 500 companies listed on stock exchanges in the United States. They concluded that LSTM was more capable and suitable of learning from long-term dependencies and showed significantly improved accuracy when a longer historical data was used. More recently, (Oak *et al.*, 2024) developed a Bi-LSTM model to forecast stocks in Indian market in short-term using multiple technical indicators. Their experiments achieved strong R^2 scores, indicating that deep learning models are highly effective when supported by sufficient feature sets.

In Vietnam, (Vuong *et al.*, 2022) used XGBoost and Random Forest to forecast banking stock prices. They found that machine learning models could deal with local market noise and performed much better than ARIMA in terms of error and stability.

In recent years, through many experiments and research, people found out that combining sentiment analysis with forecasting models has proven effective in improving prediction accuracy. In the Vietnamese context, researchers applied PhoBERT, a Vietnamese-language Bidirectional Encoder Representations from Transformers (BERT) model, to classify more than 1,000 headlines from CafeF.vn with an accuracy over 93%. The resulting sentiment labels were then used as model inputs, confirming the effectiveness of sentiment signals in price prediction (Tun *et al.*, 2021). Further supporting this, a study using PhoBERT-based sentiment analysis on nearly 40,000 financial news articles demonstrated over 81% classification accuracy. Although the authors found no immediate price movement as the news releases, the data still revealed a quite strong relationship between sentiment change and stock return volatility (Vu *et al.*, 2023). Globally, (Gu *et al.*, 2024) developed a FinBERT–LSTM model that ingests both financial news sentiment and price data to forecast NASDAQ-100 stocks. Their hybrid approach achieved the lowest forecasting error among compared models, demonstrating the potential value of sentiment features. Altogether, these works reinforce the importance of including sentiment features, processed via modern language models, as valuable complements to technical indicators in forecasting systems.

1.4. Research gaps

Although prior research has produced many useful results, there are still significant limitations that we can seeks to address. Most studies only focus on one or a few stocks, or predict for the whole index value. This makes it difficult for investors to observe specific predicted future price movements of specific stocks they are interested in. In addition, there is a lack of studies that compare different forecasting models under the same experimental conditions using unified datasets. While sentiment analysis has shown clear benefits, very few models have included it as part of a real-time, automated forecasting pipeline. Most previous studies generate predictions manually and are not designed to update daily. Moreover, many papers focus only on predicting price direction (up or down), while actual price forecasting, which is more practical for investors, has been explored far less.

Addressed the limitations of the previous research above, this thesis builds a full and automatic system to forecast VN30 stock prices. The system uses two main types of data: past stock prices and news sentiment. It runs different machine learning and deep learning models to make predictions, such as Ridge Regression, Long Short Term Memory (LSTM), and Bidirectional Gated Recurrent Unit (BiGRU) with attention. The models are evaluated based on common metrics like RMSE, MAE, MAPE, and R^2. The system gives results are specific close price value of each stock in VN30 group for the next 30 days, calculated from the present time. To make sure the system runs smoothly and can grow in the future, it is built with Apache Airflow to work automatically. After that, all the results are shown in a Power BI dashboard, which being automatically updated every day through Power BI Gateway. This thesis not only tests the models with real data but also gives a working system that can be used for research and solve real-world problems.

Chapter 2 THEORETICAL FOUNDATION, APPLIED METHODOLOGY AND TECHNOLOGY

2.1. Artificial intelligence and machine learning

2.1.1. Artificial intelligence

Artificial intelligence (AI) is a field of computer science that aims to develop systems that can perform tasks that typically require human intelligence, such as natural language understanding, image recognition, image processing, and decision making. The overall goal of AI is to improve efficiency, automation and productivity in many industries.

Nowadays, AI is the most preffered tool. AI development and its application in real life are becoming more and more popular. There are many fields that AI has been applied to help people with their tasks. For example, in the field of transportation, AI supports the development of autonomous car and traffic management systems. Self-driving cars are used AI algorithms to detect and recognize obstacles and make real-time driving decisions with high precision. In addition, in e-commerce field, businesses applied AI to analyze customer behaviour and then offer personalized product recommendations.

Figure 2.1 AI application in self-driving car

On the other hand, AI also contributed to healthcare field, by helping doctors and medical systems improve the accuracy and speed of diagnosis by analyzing medical images and detecting diseases at early stages. Moreover, in the financial sector, AI is used for detecting fraud, scoring credit risk, and making forecast in economic trends.

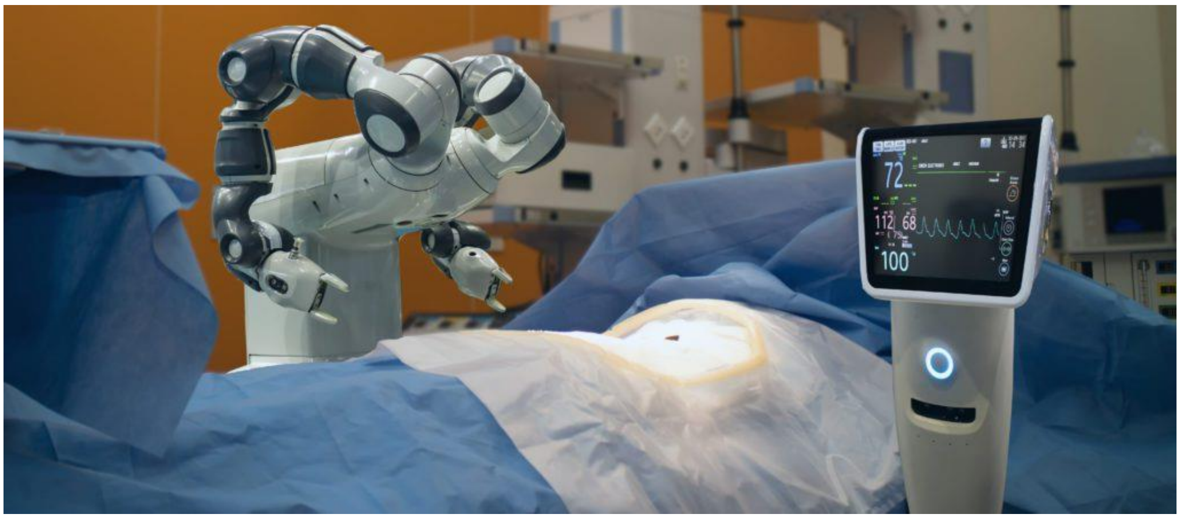


Figure 2.2 AI implementation in hospital to monitor patient conditions

2.1.2. Machine learning

Machine learning (ML) is a branch of artificial intelligence that focuses on building algorithms which can learn from data and make predictions without being explicitly programmed. In financial forecasting, ML is usually used to discover patterns in large datasets and produce more accurate predictions than traditional statistical models, especially when dealing with complex, fluctuations, and non-linear market behavior.

Machine learning techniques can be divided into three main types: supervised learning, unsupervised learning, and reinforcement learning.

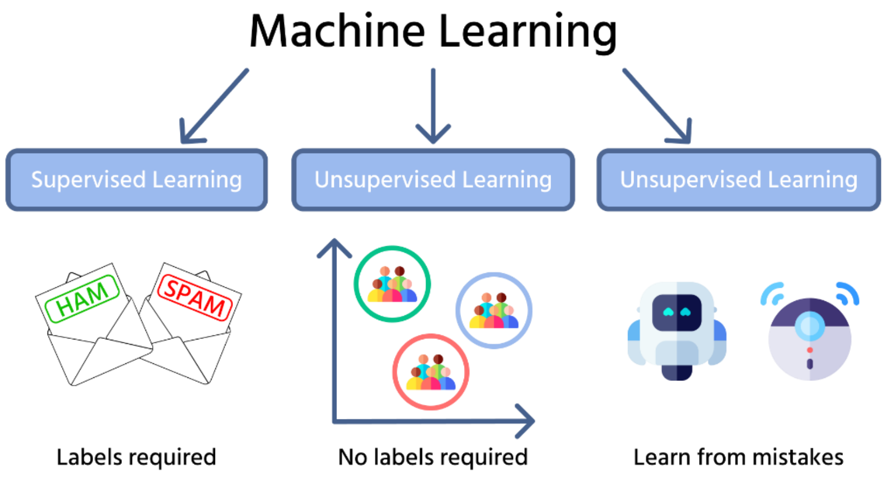


Figure 2.3 Types of machine learning (ML Introduction with scikit-learn, 2025)

This thesis mainly focused on supervised learning. In supervised learning, a predictive model is trained using input and output pairs after the algorithm has learned from labeled data. Compared to other regression methods like Linear Regression, Ridge Regression is a linear model that adds L2 regularization to reduce overfitting. Although Ridge Regression is simpler than advanced models like tree-based methods and neural networks, it serves as a useful benchmark. In this research, Ridge Regression is included to compare how a basic and traditional linear model performs against more sophisticated algorithms and deep learning models.

Pipeline of machine learning:

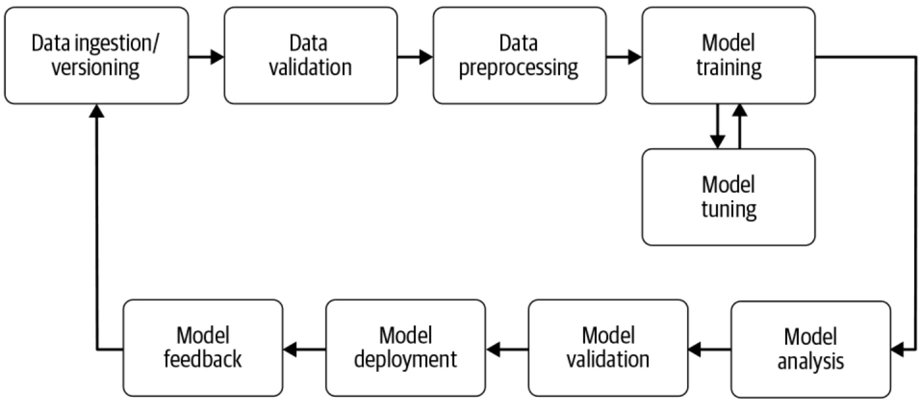


Figure 2.4 Pipeline chart of machine learning (Khalid Nazarov, 2023)

In a typical machine learning pipeline, the process begins with data collection. Raw data is gathered and stored, then validated to detect errors, outliers, or missing values. After validation, data is preprocessed through steps such as scaling, feature creation, or annotation to enhance model performance. Once prepared, the data is used for model training, where patterns are learned. During training, hyperparameters are fine-tuned to optimize results. The trained model is then evaluated using standard metrics. If the outcomes meet expectations, the model is deployed to make predictions on new, unseen data. After deployment, performance is monitored, and feedback is collected to detect issues or concept drift. This feedback is then used to update the dataset and improve the model over time.

In this thesis, the pipeline starts by crawling daily stock prices and financial news for all VN30 companies. Preprocessing steps such as feature engineering and data mapping are then applied. The models are trained and evaluated before producing future forecasts. To ensure consistency and efficiency, the entire process is scheduled and executed automatically using Apache Airflow.

2.1.3. Deep learning

2.1.3.1. Definition and overview

Deep learning sits within machine learning and works by using artificial neural networks with many layers to capture complex patterns. Many people in finance now use deep learning because it can handle highly non-linear data and track sequences over time. Unlike traditional machine learning, deep learning needs more data and stronger computers, but it often gives better results when we predict stock prices.

Can say that deep learning is a small branch of machine learning. However, in recent years, deep learning is mentioned as a booming trend in AI revolution. Businesses and companies are increasingly adopting deep learning models into their workflows and decision support, helping them significantly improve efficiency.

Deep learning training process can be expressed as the following mathematical formula:

$$y=f(x;\theta)$$

where:

- x: input data
- y: prediction value
- θ: parameters that model learns in training phase
- f: network architecture

2.1.3.2. Recurrent neural networks (RNN)

Recurrent Neural Networks (RNNs) are a type of neural network that allow to store, learn and remember what happened before, so they work well with sequential data. This makes them quite handy for forecasting time series.

Important components and characteristics of RNN:

- Recurrent connections: the output of each neuron at one point in time will be used as the input for the same neuron or other neurons in the next computation. This gives RNN the ability to remember information from previous step in the sequence.
- Hidden state: RNNs maintain a hidden state, which is a short-term memory, allows it to remember previous steps in the sequence. This state is continuously updated through each time step in the sequence.

- Processing sequential data: due to the ability to store information from previous steps, RNNs are suited to processing sequential data such as text, audio, or time-series data in financial.

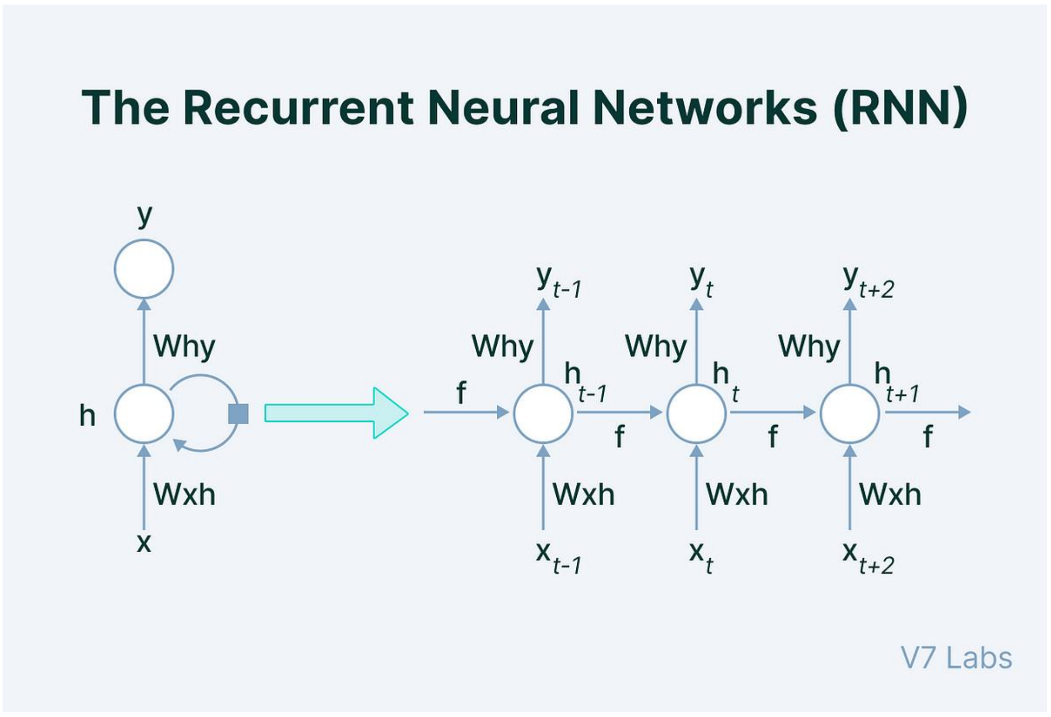


Figure 2.5 RNN's architecture (Jamdar, 2024)

But when the sequences get long, regular RNNs often run into the vanishing gradient problem (Pascanu, Mikolov and Bengio, 2013). To fix this, researchers created Long Short-Term Memory (LSTM) networks. LSTMs use memory cells and gates to keep track of long-term information more effectively. Because of this, LSTMs have become one of the most popular models for time series forecasting. Another useful model is the Bidirectional Gated Recurrent Unit (BiGRU), which reads data in both directions to understand more context (Azman, Pathmanathan and Balakrishnan, 2025).

2.1.3.3. Long Short-Term Memory (LSTM)

LSTM is designed to solve the existed problems of RNN. LSTM allows to store and maintain long-term information in the sequence. LSTM's structure includes 3 gates: input gate, forget gate, and output gate. LSTM can select which information need to be stored and which information need to be skipped. So that, LSTM is an ideal model for forecasting stock prices. It can remember patterns and signals from far in the past, which helps create forecasts that follow real trends.

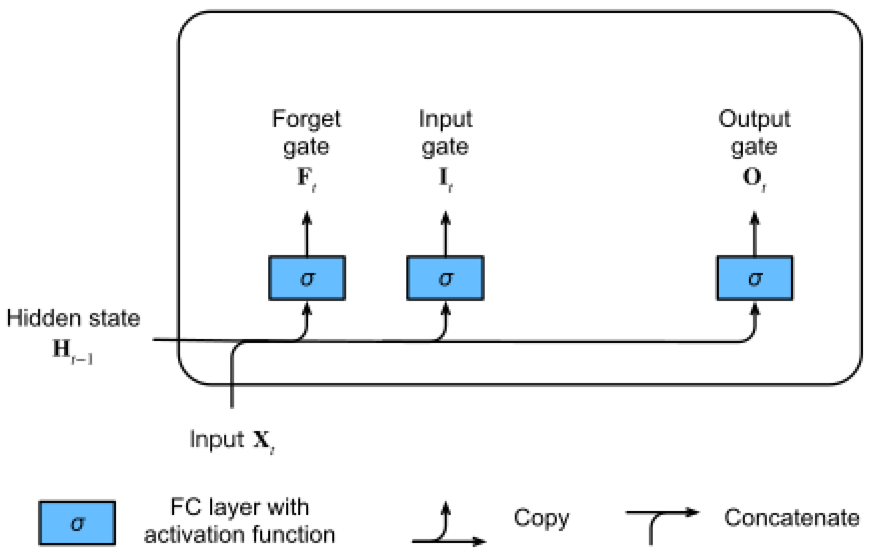

Forget gate $\mathbf{F}_t$
Input gate $\mathbf{I}_t$
Output gate $\mathbf{O}_t$
σ
σ
σ
Hidden state $\mathbf{H}_{t-1}$
Input $\mathbf{X}_t$
σ
FC layer with activation function
Copy
Concatenate

Figure 2.6 LSTM architecture (Aston Zhang, 2021)

Forget gate determines which information from the previous state need to be deleted. It helps the model forget unimportant information to avoid storing too much unnecessary information in memory.

Input gate controls how much new information from the current input will be added to the cell state. It secures the LSTM's memory from being overloaded.

Output gate regulates how much information from the state will be used to compute the new hidden state. This is the information that the model will output at the current time step.

2.1.3.4. Gated Recurrent Unit (GRU)

GRU is a simplified version of LSTM with fewer control gates but still perform well at retaining long-term information. GRU has only 2 gates: reset gate and update gate. GRU tends to learn faster and is less computationally demanding than LSTM.

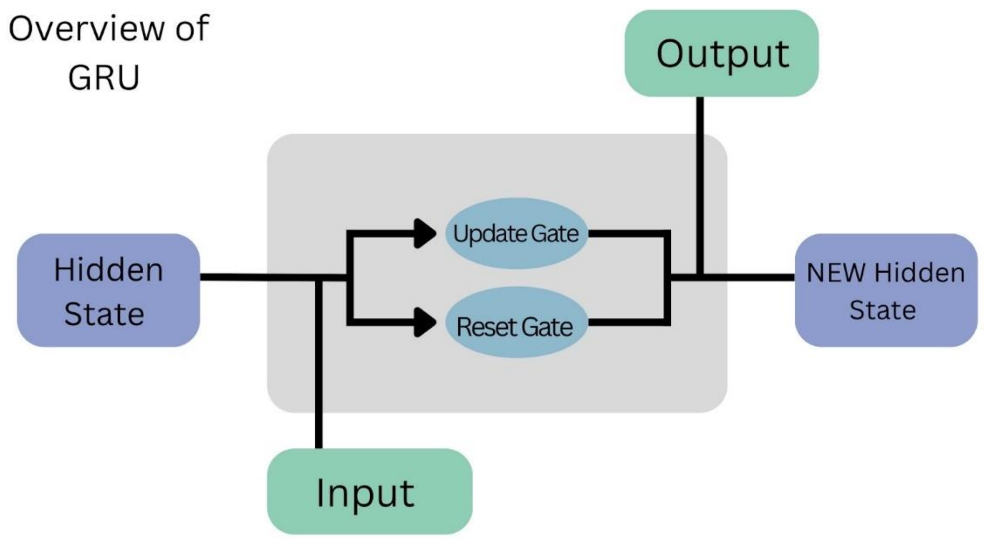


Figure 2.7 GRU's architecture (Sayed Fahim Ali Dawdye, 2024)

Update gate decides how much information from the previous hidden state should be kept and how much new information from the current input should be added. It helps the model keep important long-term information and prevents it from forgetting useful patterns over time.

Reset gate controls how much of the past information should be used to calculate the new hidden state. In other words, it helps the model “forget” unneeded old information. When the reset gate is close to zero, the model ignores most of the past and focuses more on the new input.

New hidden state is computed using both the update gate and the reset gate. Unlike LSTM, which has more complex steps, GRU combines these two gates to update the hidden state in a simpler and faster way.

In some conditions, you can consider add Attention mechanism on top of LSTM or BiGRU layers. Attention helps the model look more at the important time steps and not treat all data the same. This way, the model can find key signals that really help with prediction. It is really useful for time series data.

The combination between LSTM and BiGRU brings a strong power for learning sequences and also smart focus on what matters. This is very suitable for forecasting stock prices because it can see the trend but also notice sudden changes. As a results, this helps make predictions more accurate.

2.2. Model optimization

Building an accurate and reliable forecasting model depends not only on choosing a good architecture but also on applying proper optimization techniques. This section describes each key technique used in this research, along with clear explanations of the main parameters involved.

2.2.1. Loss function

A loss function tells us how close the model's predictions are to the actual results. For regression problems, the Mean Squared Error (MSE) is widely used. It finds the average of the squared gaps between predicted values and true values. Squaring these gaps means that larger errors have a bigger impact, pushing the model to reduce big mistakes as much as possible. When the MSE is lower, it means the predictions fit the real data better.

Mathematical expression of MSE:

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

Where:

- y_i: actual value
- $\hat{y}_i$: predicted value
- n: number of predictions

In Ridge Regression, the loss function is extended by adding an L2 penalty term to reduce overfitting by shrinking the model's weights.

The Ridge objective is:

$$J(\theta) = MSE + \alpha \sum_{j=1}^{p} \theta_j^2$$

Where:

- α: regularization parameter
- p: number of model coefficients
- θ_j: coefficient for feature j

2.2.2. Optimizer: Adam

An optimizer updates the weights of a model while training, so that the loss function becomes smaller. Adam (Adaptive Moment Estimation) is one of the most popular optimizers in deep learning because it combines the good points of AdaGrad and RMSProp (Ba and Kingma, 2015). Adam adjusts the learning rate for each parameter automatically by looking at the mean and the variance of the gradients. This helps the model train faster and more smoothly.

2.2.3. Hyperparameter tuning: Optuna and TPE

Hyperparameters are settings that you choose before training, like how many units a layer has, how much dropout you use, or what the starting learning rate is. Good hyperparameters are very important because they can improve how well the model works. In this study, I uses Optuna (Akiba *et al.*, 2019) to tune hyperparameters automatically. Optuna uses an algorithm called Tree-structured Parzen Estimator (TPE). TPE learns which parameter settings give good results and then suggests better values for the next trials.

Some important hyperparameters tuned include units, dropout rate, and learning rate. Units are number of neurons in models layers. More units can capture more patterns but increase computation. In addition, dropout rate is a fraction of neurons randomly turned off during training to reduce overfitting. One more important hyperparameter is learning rate, which is initial step size for updating weights.

2.2.4. Early stopping

Early stopping (Prechelt, 1998) is a simple but effective method to prevent overfitting. It stops training automatically when the validation loss does not improve for a defined number of epochs, called patience. This prevents the model from memorizing noise in the training data and saves time.

2.2.5. Learning rate scheduling: ReduceLROnPlateau

Even when the starting learning rate is good, it often helps to lower it step by step during training. Reduce Learning Rate on Plateau (Smith, 2017) watches the validation loss, and if it stays the same for a while, it reduces the learning rate by multiplying it by a number smaller than 1. For example, a factor of 0.5 means cutting the learning rate in half. This allows the optimizer to make smaller, better moves to find the best minimum.

The rule is:

$$\eta_{new} = \eta_{old} \times factor$$

2.2.6. Data scaling: StandardScaler and MinMaxScaler

Before feeding data into a neural network, input features must be normalized. Two common scalers are used for this purpose.

StandardScaler standardizes each feature to have zero mean and unit variance. This ensures all features contribute equally and stabilizes the learning process.

$$z = \frac{x - \mu}{\sigma}$$

Where μ is the mean and σ is the standard deviation.

MinMaxScaler rescales the standardized data to a fixed range, usually from 0 to 1. This keeps input values within the activation function's sensitive region.

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

Combining both scalers helps the models converge faster and prevents issues caused by large feature differences.

2.3. Model evalutation for regression problem

To measure how well the forecasting models perform, this research uses four standard regression evaluation metrics with clear mathematical definitions.

2.3.1. R² score

R^2 score (coefficient of determination) shows how much of the variance in actual prices is explained by the predicted prices. A higher R^2 indicates better fit. It is defined as:

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

Where:

- y_i: actual value
- $\hat{y}_i$: predicted value
- $\bar{y}$: mean of actual values
- n: number of data points

2.3.2. Root Mean Square Error (RMSE)

RMSE measures the square root of the average squared differences between predictions and actual values. It shows the typical size of prediction errors in the same units as the stock price.

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

2.3.3. Mean Absolute Error (MAE)

MAE calculates the average absolute difference between predicted and actual values. MAE is easy to interpret because it does not square the errors.

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

2.3.4. Mean Absolute Percentage Error (MAPE)

MAPE expresses errors as a percentage of the true values. This helps understand how big the errors are relative to the actual prices.

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

2.4. Financial time series forecasting and automated pipeline design

2.4.1. Financial time series forecasting

Financial time series forecasting involves predicting future stock prices based on their historical patterns. Unlike other time series, financial data often shows high volatility, sudden jumps, and is sensitive to unexpected news or economic changes.

To address this, it is important to enrich the historical price data with additional features that help capture the market's behavior. In this research, the dataset includes daily stock prices for each VN30 stock, plus some technical indicators. In addition, average daily sentiment scores are derived from financial news articles using a sentiment analysis model.

Combining these technical and sentiment features provides a new more complete view of the factors that influence stock prices. This helps the forecasting models learn meaningful patterns and make more accurate predictions.

2.4.2. Automated pipeline design

This research develops a fully automated forecasting pipeline designed to handle daily stock prediction tasks for all VN30 companies. The main part of this pipeline is Apache Airflow, an open-source platform for orchestrating complex workflows. Airflow allows users to define tasks as Directed Acyclic Graphs (DAGs), meaning each task runs in a defined order without looping back. This structure ensures that each step depends on the successful completion of previous steps, making the pipeline robust and organized.

The pipeline follows clear stages. First, it crawls both historical and new daily stock price data using custom scripts. Second, it gathers recent financial news for each stock from trusted sources like CafeF. To analyze this text, the pipeline uses a fine-tuned PhoBERT model.

BERT (Bidirectional Encoder Representations from Transformers) is a language model based on the Transformer architecture. It uses self-attention to understand word context from both directions in a sentence, making it highly effective for tasks like sentiment analysis. PhoBERT adapts BERT to Vietnamese by pre-training it on a large Vietnamese news corpus, allowing it to extract relevant sentiment signals from local financial news.

After sentiment extraction, the pipeline merges these sentiment scores with daily stock data, generating a rich dataset that includes both technical indicators and investor mood signals. The next stage is training. The pipeline feeds this enriched data into machine learning and deep learning models to learn stock price patterns. After training, the models generate predictions and output an evaluation report summarizing how well each model performed.

Apache Airflow controls this entire workflow. It schedules each task automatically, tracks execution logs, and retries failed steps when needed. This design ensures that the forecasting process is scalable, transparent, and repeatable every day without manual supervision.

Once the daily predictions and evaluation results are generated, they are securely stored for reporting and decision support. This research uses Power BI, a leading business intelligence tool developed by Microsoft, to build a dynamic dashboard for visualizing the forecasts and performance metrics.

To keep the dashboard current, the system uses Gateway. This Gateway acts as a secure bridge between on-premises data storage and the Power BI Service in the cloud. It allows scheduled automatic refreshes of the data source without manual uploads. As a result, investors, analysts, and stakeholders always have access to the latest forecasts and performance insights.

Chapter 3 IMPLEMENTATION AND EXPERIMENTS RESULTS

3.1. Implementation framework

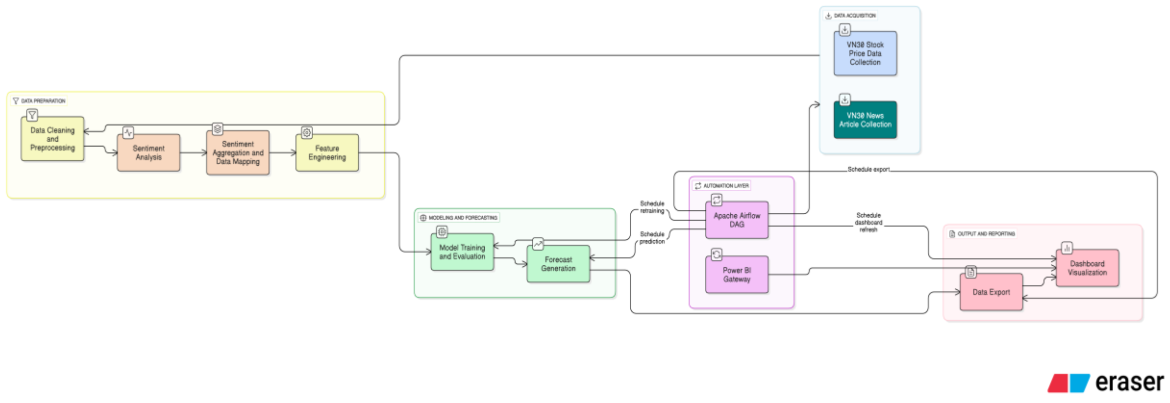


Figure 3.1 Implementation framework flowchart

This chapter presents the full process of how the forecasting system was built and tested, from data collection to model training, evaluation, and final deployment.

The first step focuses on building a reliable dataset that combines two types of information: daily historical stock prices and financial news articles.

To extract market sentiment from the news articles, a pre-trained PhoBERT model is used to classify each article as positive, negative, or neutral. A numeric sentiment score is also assigned to each article based on prediction confidence. These scores are then grouped by day and merged with stock price data to create a complete dataset that reflects both technical indicators and investor sentiment for each trading day.

Once the dataset is ready, three forecasting models are trained: Ridge Regression, BiGRU with Attention, and LSTM. Each model is optimized using hyperparameter tuning and evaluated using standard metrics such as R-squared, RMSE, MAE, and MAPE. After comparison, the best-performing model is selected to generate 30-day future price forecasts for all VN30 stocks.

The results are visualized through an interactive dashboard built in Power BI. The design focuses on clarity, usability, and providing quick insights to investors and analysts.

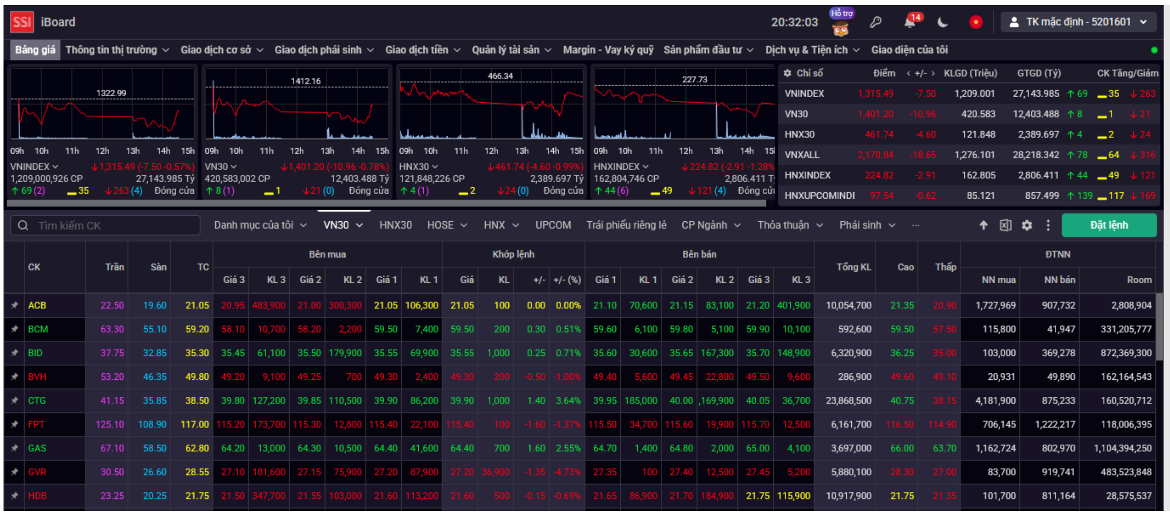
To make the system fully functional and hands-free, the final step is to automate the entire pipeline using Apache Airflow. All tasks are scheduled and monitored through Airflow DAGs. In addition, Power BI Gateway is used to ensure that the dashboard always stays up to date by automatically syncing the latest output files without manual uploads. This makes the entire system reliable, scalable, and ready for real-world daily use.

3.2. Data scraping and processing

In this thesis, the construction of a complete and high-quality dataset was considered the most essential foundation before any forecasting models could be designed. The dataset was expected to include both daily stock trading data and financial news articles related to VN30 companies, in a form that could be updated automatically and consistently. By combining technical market data with sentiment signals extracted from news, a more comprehensive view of stock price behavior could be formed.

To collect price data, several public and semi-public sources were evaluated. VNDIRECT’s open API was initially tested, as it provided live trading information through a well-structured interface. However, its historical coverage was found to be incomplete in many cases, and unstable rate limits often caused interruptions in access. TradingView, while widely used for its charting capabilities, did not support bulk data downloads and imposed scraping limitations that made it difficult to use reliably in a daily pipeline. Attempts to extract price tables directly from local finance websites were also blocked due to strong anti-bot protections that could not be bypassed effectively.

As a solution, the Fast Connect API from SSI Securities Corporation was selected. This service is officially supported, trusted by developers and investors in Vietnam, and provides daily OHLC and volume data in a clean JSON format. Secure access was granted through a Consumer ID and Secret obtained directly from the SSI office. After being tested, this API was found to be the most stable and consistent source, especially suitable for regular updates.



VNINDEX ↓1,315.49 (-7.50 -0.57%)
1,209,000,926 CP 27,143.985 Tỷ
↑ 69 (2) ▬ 35 ↓ 263 (4) Đóng cửa

VN30 ↓1,401.20 (-10.96 -0.78%)
420,583,002 CP 12,403.488 Tỷ
↑ 8 (1) ▬ 1 ↓ 21 (0) Đóng cửa

HNX30 ↓461.74 (-4.60 -0.99%)
121,848,226 CP 2,389.697 Tỷ
↑ 4 (1) ▬ 2 ↓ 24 (0) Đóng cửa

HNXINDEX ↓224.82 (-2.91 -1.28%
162,804,746 CP 2,806.411 T
↑ 44 (6) ▬ 49 ↓ 121 (4) Đóng cử

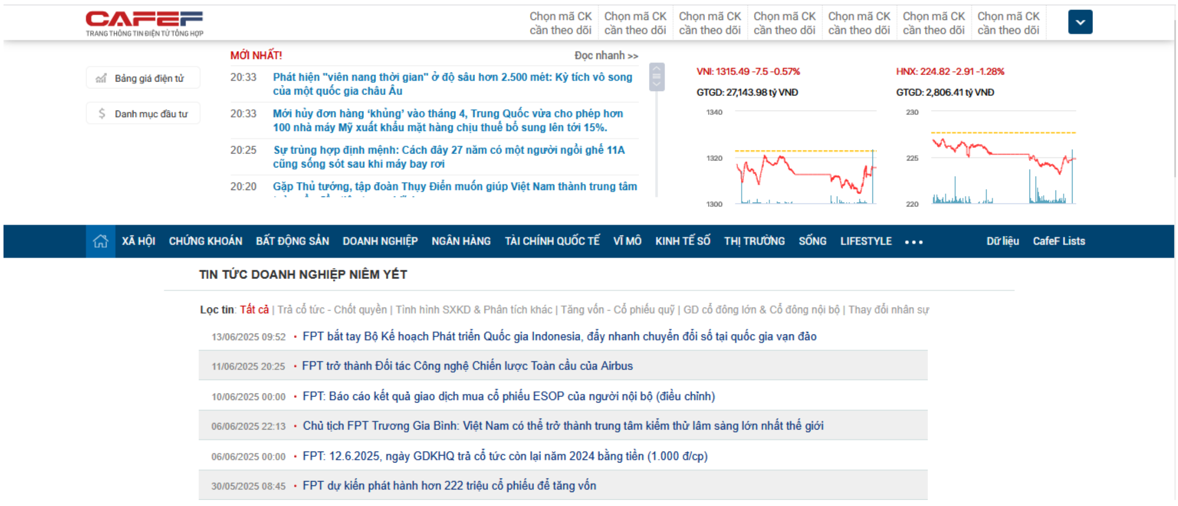
Chỉ số	Điểm	+/-	KLGD (Triệu)	GTGD (Tỷ)	CK Tăng/Giảm		
VNINDEX	1,315.49	-7.50	1,209.001	27,143.985	↑ 69	▬ 35	↓ 263
VN30	1,401.20	-10.96	420.583	12,403.488	↑ 8	▬ 1	↓ 21
HNX30	461.74	-4.60	121.848	2,389.697	↑ 4	▬ 2	↓ 24
VNXALL	2,170.84	-18.65	1,276.101	28,218.342	↑ 78	▬ 64	↓ 316
HNXINDEX	224.82	-2.91	162.805	2,806.411	↑ 44	▬ 49	↓ 121
HNXUPCOMINDI	97.54	-0.62	85.121	857.499	↑ 139	▬ 117	↓ 169

CK	Trần	Sàn	TC	Bên mua						Khớp lệnh				Bên bán						Tổng KL	Cao	Thấp	ĐTNN		
				Giá 3	KL 3	Giá 2	KL 2	Giá 1	KL 1	Giá	KL	+/-	+/- (%)	Giá 1	KL 1	Giá 2	KL 2	Giá 3	KL 3				NN mua	NN bán	Room
ACB	22.50	19.60	21.05	20.95	483,900	21.00	300,300	21.05	106,300	21.05	100	0.00	0.00%	21.10	70,600	21.15	83,100	21.20	401,900	10,054,700	21.35	20.90	1,727,969	907,732	2,808,904
BCM	63.30	55.10	59.20	58.10	10,700	58.20	2,200	59.50	7,400	59.50	200	0.30	0.51%	59.60	6,100	59.80	5,100	59.90	10,100	592,600	59.50	57.50	115,800	41,947	331,205,777
BID	37.75	32.85	35.30	35.45	61,100	35.50	179,900	35.55	69,900	35.55	1,000	0.25	0.71%	35.60	30,600	35.65	167,300	35.70	148,900	6,320,900	36.25	35.00	103,000	369,278	872,369,300
BVH	53.20	46.35	49.80	49.20	9,100	49.25	700	49.30	2,400	49.30	200	-0.50	-1.00%	49.40	5,600	49.45	22,800	49.50	9,600	286,900	49.60	49.10	20,931	49,890	162,164,543
CTG	41.15	35.85	38.50	39.80	127,200	39.85	110,500	39.90	86,200	39.90	1,000	1.40	3.64%	39.95	185,000	40.00	,169,900	40.05	36,700	23,868,500	40.75	38.15	4,181,900	875,233	160,520,712
FPT	125.10	108.90	117.00	115.20	173,700	115.30	12,800	115.40	22,100	115.40	100	-1.60	-1.37%	115.50	34,700	115.60	19,900	115.70	12,500	6,161,700	116.50	114.90	706,145	1,222,217	118,006,395
GAS	67.10	58.50	62.80	64.20	13,000	64.30	10,500	64.40	41,600	64.40	700	1.60	2.55%	64.70	1,400	64.80	2,000	65.00	4,100	3,697,000	66.00	63.70	1,162,724	802,970	1,104,394,250
GVR	30.50	26.60	28.55	27.10	101,600	27.15	75,900	27.20	87,900	27.20	36,900	-1.35	-4.73%	27.35	100	27.40	12,500	27.45	5,200	5,880,100	28.30	27.00	83,700	919,741	483,523,848
HDB	23.25	20.25	21.75	21.50	347,700	21.55	103,000	21.60	113,200	21.60	500	-0.15	-0.69%	21.65	86,900	21.70	184,900	21.75	115,900	10,917,900	21.75	21.35	101,700	811,164	28,575,537

Figure 3.2 SSI trading board interface (SSI iBoard)

For news data, even more challenges were encountered. News sections from VNDIRECT and Vietstock were examined, but the majority of content consisted only of short headlines or bullet-point summaries, which lacked the detail needed for sentiment analysis. TradingView did not host full articles, instead linking to external sources, and articles were not tagged by stock symbols, making proper alignment difficult. In general, the depth and structure of news content from these sources were not considered sufficient.

After these issues were identified, CafeF.vn was chosen as the final news source. Its content includes full-text articles, clearly separated by stock, and its site structure is stable enough to support long-term scraping. Among all tested options, CafeF provided the richest, most reliable, and most usable news dataset for sentiment modeling.



MỚI NHẤT!

20:33 Phát hiện "viên nang thời gian" ở độ sâu hơn 2.500 mét: Kỳ tích vô song của một quốc gia châu Âu

20:33 Mới hủy đơn hàng 'khủng' vào tháng 4, Trung Quốc vừa cho phép hơn 100 nhà máy Mỹ xuất khẩu mặt hàng chịu thuế bổ sung lên tới 15%.

20:25 Sự trùng hợp định mệnh: Cách đây 27 năm có một người ngồi ghế 11A cũng sống sót sau khi máy bay rơi

20:20 Gặp Thủ tướng, tập đoàn Thụy Điển muốn giúp Việt Nam thành trung tâm

VNI: 1315.49 -7.5 -0.57%
GTGD: 27,143.98 tỷ VND

HNX: 224.82 -2.91 -1.28%
GTGD: 2,806.41 tỷ VND

TIN TỨC DOANH NGHIỆP NIÊM YẾT

Lọc tin: Tất cả | Trả cổ tức - Chốt quyền | Tình hình SXKD & Phân tích khác | Tăng vốn - Cổ phiếu quỹ | GD cổ đông lớn & Cổ đông nội bộ | Thay đổi nhân sự

13/06/2025 09:52 • FPT bắt tay Bộ Kế hoạch Phát triển Quốc gia Indonesia, đẩy nhanh chuyển đổi số tại quốc gia vạn đảo

11/06/2025 20:25 • FPT trở thành Đối tác Công nghệ Chiến lược Toàn cầu của Airbus

10/06/2025 00:00 • FPT: Báo cáo kết quả giao dịch mua cổ phiếu ESOP của người nội bộ (điều chỉnh)

06/06/2025 22:13 • Chủ tịch FPT Trương Gia Bình: Việt Nam có thể trở thành trung tâm kiểm thử lâm sàng lớn nhất thế giới

06/06/2025 00:00 • FPT: 12.6.2025, ngày GDKHQ trả cổ tức còn lại năm 2024 bằng tiền (1.000 đ/cp)

30/05/2025 08:45 • FPT dự kiến phát hành hơn 222 triệu cổ phiếu để tăng vốn

Figure 3.3 CafeF news interface for FPT stock (CafeF.vn)

Once data sources were finalized, processing functions were written, tested in Jupyter Notebook. After confirming the stability of each data extraction step, the code was refined into organized scripts that could be run repeatedly to update the dataset with new information and keep it consistent over time.

3.2.1. Price data collection

To collect daily price data, a set of Python functions was developed to connect directly to the Fast Connect API using a registered Consumer ID and Secret. Each time the script ran, a fresh access token was requested from the authentication endpoint, as the token expired after a short time and needed to be renewed.

Once authenticated, the script called the IndexComponents endpoint to retrieve the latest list of VN30 stocks. This ensured the workflow always matched the current index composition, even when stocks were updated by the exchange.

For each stock, daily OHLC data was downloaded in batches to handle the API's page size limits. A retry mechanism was included to make the process more stable by automatically attempting failed requests up to three times with short pauses.

Before downloading, the script checked whether a CSV file already existed for the stock in the Price_update folder. If it did, the most recent trading date was read and used to continue the crawl from the next day. If no file was found, data was fetched from January 1, 2015.

The raw data was then converted to a Pandas DataFrame, where date formats were standardized and incomplete rows were removed. For older records missing the Volume field, a default value of zero was added to maintain structural consistency.

After cleaning, new records were merged with any existing data, sorted by date and volume, and checked for duplicates. The final dataset was saved back into the Price_update folder, ready to be used for further processing.

3.2.2. News data collection

Collecting news from CafeF required extra care because the site loads content dynamically using JavaScript and applies different HTML structures to different articles. To handle this, a custom crawler was built using Selenium with undetected_chromedriver, allowing it to mimic human browsing. Random wait times were added between actions to reduce the risk of being blocked.

For each stock, the crawler was directed to its news page where it scanned visible articles and extracted the publication time, headline, and link. Extracting the article body was more difficult due to inconsistent layouts. To address this, a fallback strategy was applied. The crawler first tried a standard XPath, then looked for a div with the class name detail-content, and if both failed, it parsed the entire page body line by line, filtering out short or irrelevant content. Even when full text could not be retrieved, basic information like title, link, and time was still saved for later review.

To avoid duplicates in repeated runs, a filtering step was added. Before crawling, the script checked for an existing CSV file. If found, previously saved links and timestamps were loaded, and any article already recorded or older than the latest was skipped. Only new articles were kept. The updated results were then merged and saved as a clean file in the News_update folder.

The crawler continued navigating through multiple pages by clicking the Next button until no more new articles appeared. To avoid memory overload, the browser driver was automatically restarted after every five stocks, which kept the crawler stable and running smoothly.

3.2.3. Challenges and solutions

Getting the full data collection process to run smoothly was not simple. Several challenges had to be solved, especially when working with news data. One of the first problems was avoiding detection while scraping articles from CafeF. The site's anti-bot system was sensitive, and access could be blocked quickly if requests looked unnatural. To reduce this risk, different user-agent strings were tested, stealth settings were added to the browser driver, and random wait times were introduced between actions to help mimic human browsing behavior.

Another issue was the inconsistent layout across CafeF articles. A single HTML pattern could not be applied to all pages. Some articles were structured with tables, others used nested divs, and a few placed the text directly in the body of the page. To handle this, multiple examples were reviewed, and a fallback logic was built. It was tested and adjusted until a sequence was found that worked for nearly all cases.

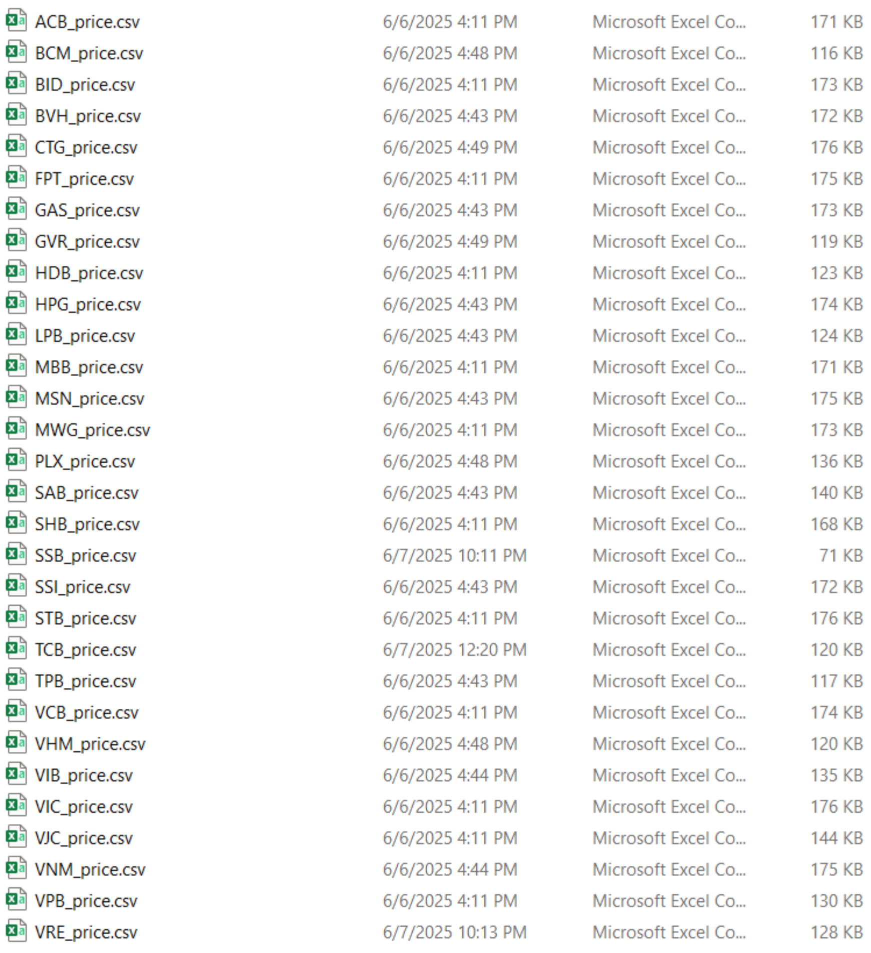
Older articles also caused problems when timestamps were missing or written in strange formats. If not handled correctly, this would have made it hard to match news with stock trading dates. To deal with that, flexible date parsing was implemented, along with backup formats and a default value in case parsing failed completely.

Keeping the data clean during repeated crawling was another priority. A filtering step was added to check previously saved links and timestamps before starting a new run. This logic was tested with multiple stocks to make sure that only new articles were added and no duplicates were stored.

Finally, memory issues had to be considered. Long crawling sessions sometimes caused Selenium to crash due to high RAM usage. To prevent this, the driver was set to restart automatically after every five stocks. Although simple, this method worked well and helped the crawler stay stable through the entire run.

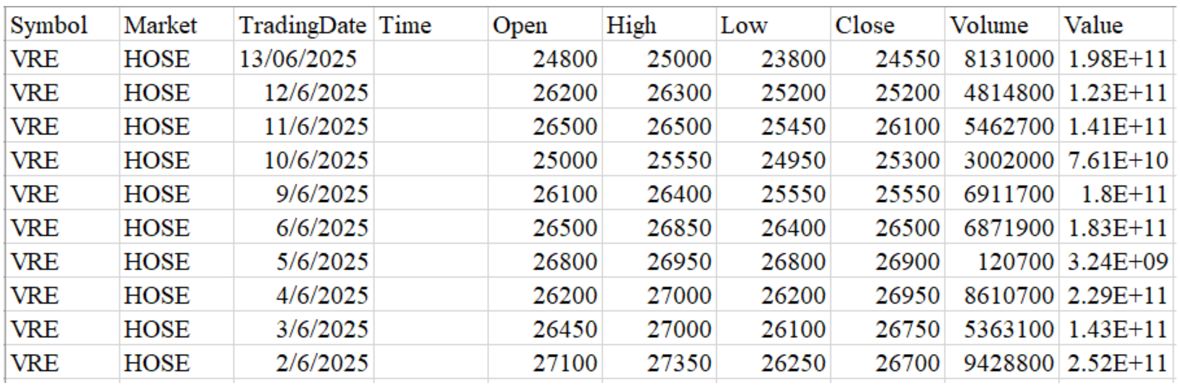
3.2.4. Final results and data organization

After refining each step, a clean and well-organized dataset is being ready for the next analysis phases. The Price_update folder contains one CSV file for each VN30 stock.



ACB_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	171 KB
BCM_price.csv	6/6/2025 4:48 PM	Microsoft Excel Co...	116 KB
BID_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	173 KB
BVH_price.csv	6/6/2025 4:43 PM	Microsoft Excel Co...	172 KB
CTG_price.csv	6/6/2025 4:49 PM	Microsoft Excel Co...	176 KB
FPT_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	175 KB
GAS_price.csv	6/6/2025 4:43 PM	Microsoft Excel Co...	173 KB
GVR_price.csv	6/6/2025 4:49 PM	Microsoft Excel Co...	119 KB
HDB_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	123 KB
HPG_price.csv	6/6/2025 4:43 PM	Microsoft Excel Co...	174 KB
LPB_price.csv	6/6/2025 4:43 PM	Microsoft Excel Co...	124 KB
MBB_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	171 KB
MSN_price.csv	6/6/2025 4:43 PM	Microsoft Excel Co...	175 KB
MWG_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	173 KB
PLX_price.csv	6/6/2025 4:48 PM	Microsoft Excel Co...	136 KB
SAB_price.csv	6/6/2025 4:43 PM	Microsoft Excel Co...	140 KB
SHB_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	168 KB
SSB_price.csv	6/7/2025 10:11 PM	Microsoft Excel Co...	71 KB
SSI_price.csv	6/6/2025 4:43 PM	Microsoft Excel Co...	172 KB
STB_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	176 KB
TCB_price.csv	6/7/2025 12:20 PM	Microsoft Excel Co...	120 KB
TPB_price.csv	6/6/2025 4:43 PM	Microsoft Excel Co...	117 KB
VCB_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	174 KB
VHM_price.csv	6/6/2025 4:48 PM	Microsoft Excel Co...	120 KB
VIB_price.csv	6/6/2025 4:44 PM	Microsoft Excel Co...	135 KB
VIC_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	176 KB
VJC_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	144 KB
VNM_price.csv	6/6/2025 4:44 PM	Microsoft Excel Co...	175 KB
VPB_price.csv	6/6/2025 4:11 PM	Microsoft Excel Co...	130 KB
VRE_price.csv	6/7/2025 10:13 PM	Microsoft Excel Co...	128 KB

Figure 3.4 30 price data files for 30 stocks in updated VN30 list are stored



Symbol	Market	TradingDate	Time	Open	High	Low	Close	Volume	Value
VRE	HOSE	13/06/2025		24800	25000	23800	24550	8131000	1.98E+11
VRE	HOSE	12/6/2025		26200	26300	25200	25200	4814800	1.23E+11
VRE	HOSE	11/6/2025		26500	26500	25450	26100	5462700	1.41E+11
VRE	HOSE	10/6/2025		25000	25550	24950	25300	3002000	7.61E+10
VRE	HOSE	9/6/2025		26100	26400	25550	25550	6911700	1.8E+11
VRE	HOSE	6/6/2025		26500	26850	26400	26500	6871900	1.83E+11
VRE	HOSE	5/6/2025		26800	26950	26800	26900	120700	3.24E+09
VRE	HOSE	4/6/2025		26200	27000	26200	26950	8610700	2.29E+11
VRE	HOSE	3/6/2025		26450	27000	26100	26750	5363100	1.43E+11
VRE	HOSE	2/6/2025		27100	27350	26250	26700	9428800	2.52E+11

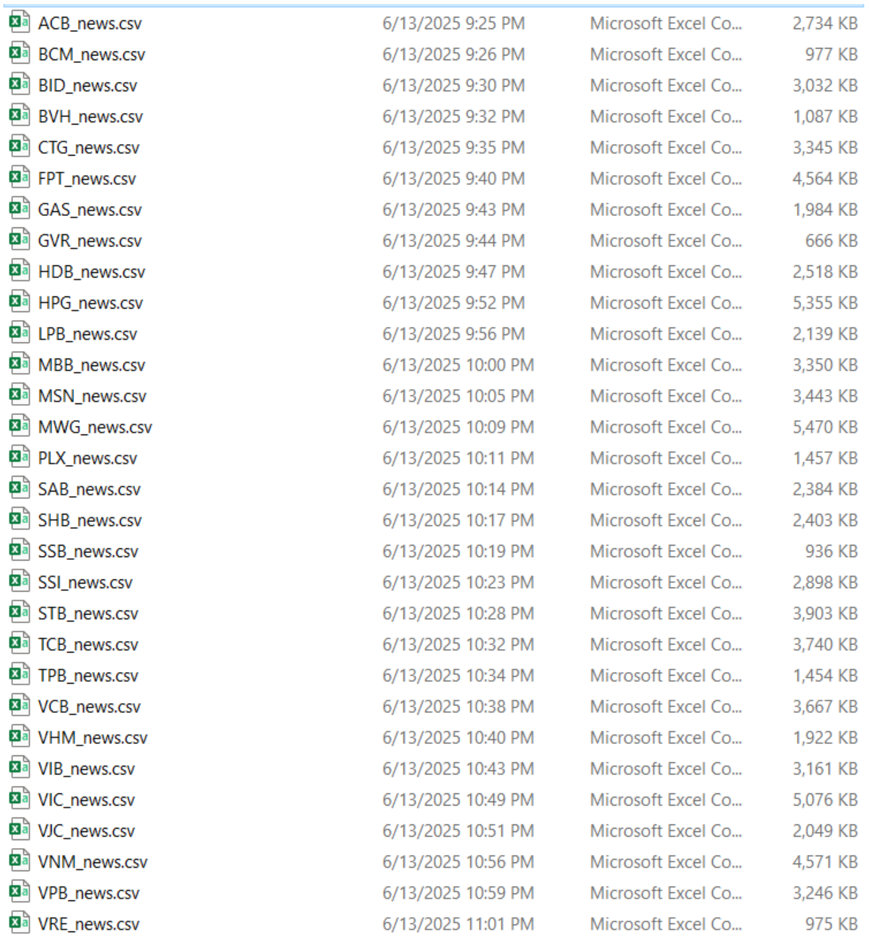
Figure 3.5 Capture of part of price data file of VRE stock

Attributes of the dataset:

Table 3.1 Attributes of the price data

Attribute	**Description**
Symbol	Stock code
Market	Name of the stock exchange on which the stock is listed
TradingDate	Stock trading date
Time	Trading hours (if any)
Open	Opening price of a stock on a trading day (in VND)
High	Highest price of a stock on a trading day (in VND)
Low	Lowest price of a stock on a trading day (in VND)
Close	Closing price of a stock on a trading day (in VND)
Volume	Total number of shares bought and sold in a trading day
Value	Total transaction value of shares in a trading day (in VND)

On the other hand, the News_update folder holds one CSV per stock, with Time, Title, Link, and Content. All files are encoded in UTF-8 and checked for duplicates each time updated. In addition, a missing_symbols_log.csv file also be kept to note any stock or date range that could not be processed, making it easy to inspect and fix later.



ACB_news.csv	6/13/2025 9:25 PM	Microsoft Excel Co...	2,734 KB
BCM_news.csv	6/13/2025 9:26 PM	Microsoft Excel Co...	977 KB
BID_news.csv	6/13/2025 9:30 PM	Microsoft Excel Co...	3,032 KB
BVH_news.csv	6/13/2025 9:32 PM	Microsoft Excel Co...	1,087 KB
CTG_news.csv	6/13/2025 9:35 PM	Microsoft Excel Co...	3,345 KB
FPT_news.csv	6/13/2025 9:40 PM	Microsoft Excel Co...	4,564 KB
GAS_news.csv	6/13/2025 9:43 PM	Microsoft Excel Co...	1,984 KB
GVR_news.csv	6/13/2025 9:44 PM	Microsoft Excel Co...	666 KB
HDB_news.csv	6/13/2025 9:47 PM	Microsoft Excel Co...	2,518 KB
HPG_news.csv	6/13/2025 9:52 PM	Microsoft Excel Co...	5,355 KB
LPB_news.csv	6/13/2025 9:56 PM	Microsoft Excel Co...	2,139 KB
MBB_news.csv	6/13/2025 10:00 PM	Microsoft Excel Co...	3,350 KB
MSN_news.csv	6/13/2025 10:05 PM	Microsoft Excel Co...	3,443 KB
MWG_news.csv	6/13/2025 10:09 PM	Microsoft Excel Co...	5,470 KB
PLX_news.csv	6/13/2025 10:11 PM	Microsoft Excel Co...	1,457 KB
SAB_news.csv	6/13/2025 10:14 PM	Microsoft Excel Co...	2,384 KB
SHB_news.csv	6/13/2025 10:17 PM	Microsoft Excel Co...	2,403 KB
SSB_news.csv	6/13/2025 10:19 PM	Microsoft Excel Co...	936 KB
SSI_news.csv	6/13/2025 10:23 PM	Microsoft Excel Co...	2,898 KB
STB_news.csv	6/13/2025 10:28 PM	Microsoft Excel Co...	3,903 KB
TCB_news.csv	6/13/2025 10:32 PM	Microsoft Excel Co...	3,740 KB
TPB_news.csv	6/13/2025 10:34 PM	Microsoft Excel Co...	1,454 KB
VCB_news.csv	6/13/2025 10:38 PM	Microsoft Excel Co...	3,667 KB
VHM_news.csv	6/13/2025 10:40 PM	Microsoft Excel Co...	1,922 KB
VIB_news.csv	6/13/2025 10:43 PM	Microsoft Excel Co...	3,161 KB
VIC_news.csv	6/13/2025 10:49 PM	Microsoft Excel Co...	5,076 KB
VJC_news.csv	6/13/2025 10:51 PM	Microsoft Excel Co...	2,049 KB
VNM_news.csv	6/13/2025 10:56 PM	Microsoft Excel Co...	4,571 KB
VPB_news.csv	6/13/2025 10:59 PM	Microsoft Excel Co...	3,246 KB
VRE_news.csv	6/13/2025 11:01 PM	Microsoft Excel Co...	975 KB

Figure 3.6 30 news data files for 30 stocks in updated VN30 list are stored

Attributes of the dataset:

Table 3.2 Attributes of the news data

Attribute	**Description**
Time	Time the news was posted, including date and time
Title	News headline
Link	The full URL linking to the original news article on CafeF
Content	Detailed news content

3.3. News sentiment analysis

After the data scraping and storing step was completed, a method was needed to transform the raw text of news articles into a numerical signal that could provide extra insight for the forecasting models. Historical price data alone was not enough to capture sudden market reactions caused by unexpected news or policy changes. To solve this, sentiment analysis was applied so that the qualitative meaning of each article could be converted into a score showing whether the news was positive, negative, or neutral for a specific stock.

By using these sentiment scores, the models could be adjusted not only based on past price patterns but also by considering how investor reactions might shift when new information appears. For that reason, sentiment analysis was considered a natural and necessary next step in the pipeline.

3.3.1. Model selection and preprocessing

In this study, selecting a suitable sentiment analysis model was considered an important step, as all news content was written in Vietnamese and included both short headlines and full-length articles. To process this kind of data accurately, a language model was needed, one that had been trained entirely on Vietnamese and had experience with sentiment classification tasks.

After reviewing available options, a model called PhoBERT was chosen. This model was built by the VinAI Research team using only Vietnamese text. It was trained on around twenty gigabytes of data and had performed well across a wide range of language processing tasks, including classification and sentiment prediction (Nguyen and Nguyen, 2020).

PhoBERT was first trained on about 20GB of Vietnamese text and has shown strong results for many language tasks like classification, and general sentiment tasks. However, the normal PhoBERT is a general model and not focused only on sentiment for news. So, I chose a version that supported for this purpose: phong02468/phobert-Vietnamese-newspaper-title-sentiment which is shared on Hugging Face.

This model was fine-tuned on about 35,000 Vietnamese news text data that were labeled as Positive (POS), Negative (NEG), or Neutral (NEU). This fine-tuning used standard cross-entropy loss and an 80% training and 20% validation split. The model reached a validation accuracy of about 92% on classifying titles, which is quite good for Vietnamese. This high accuracy was also confirmed by testing some sample predictions directly on Hugging Face, and the results matched human judgment well for common Vietnamese economic and business headlines.

In addition, to enhance the model's ability to interpret longer and more complex financial texts, the PhoBERT model was additionally fine-tuned using a manually labeled dataset of 1,000 full-length news articles. These articles were categorized into

POS, NEG, and NEU classes based on overall sentiment. After fine-tuning, the model was evaluated through manual inspection and sample predictions, which showed improved consistency and contextual understanding compared to the original version trained on short titles. As a result, this refined model was integrated into the pipeline to generate more reliable sentiment scores from full article content.

To apply this model in the project, the tokenizer and classification model were loaded using the Transformers library. A mapping table was also created to convert the model's output into readable sentiment labels such as POS, NEG, or NEU. Since the dataset contained not only short titles but also long-form articles, a custom preprocessing function was written to handle longer text safely by splitting it into smaller parts.

Thanks to this setup, reliable sentiment labels and confidence scores were generated in a format that matched the Vietnamese language and financial context. These results were then used as additional input features in the next step, where sentiment values were combined with daily stock price data to support more accurate forecasting.

3.3.2. Sentiment scoring and saving

After the sentiment model and preprocessing steps were set up, a method was needed to apply the model to each news article and convert the results into a meaningful numeric score. Since the final goal was to create a daily sentiment signal for each stock, it was important that every article was scored accurately and consistently.

To handle long Vietnamese articles that could not be processed by PhoBERT in one pass, each article was split into smaller parts, with each part containing no more than 256 tokens. For each chunk, the sentiment model was used to predict both the label and the confidence score. These results were collected for all chunks of the article.

For each small part, I run the sentiment model and get two things: the predicted label and the probability, which comes from the softmax layer. This probability shows how confident the model is about its prediction for that chunk. After predicting all chunks, all the labels and confidence scores are saved.

To determine the final sentiment label, majority voting was applied by selecting the label that appeared most frequently across all chunks. For example, if an article has five chunks and three of them are POS and two are NEU, then the final label is POS. This simple method helps balance small differences between parts of the same article.

Confidence was then calculated as the average of the confidence scores that matched the majority label only. This helped focus on how strongly the model supported the main sentiment of the article. The formula is:

$$Final\ Confidence = \frac{\sum_{i=1}^{N} confidence_i}{N}$$

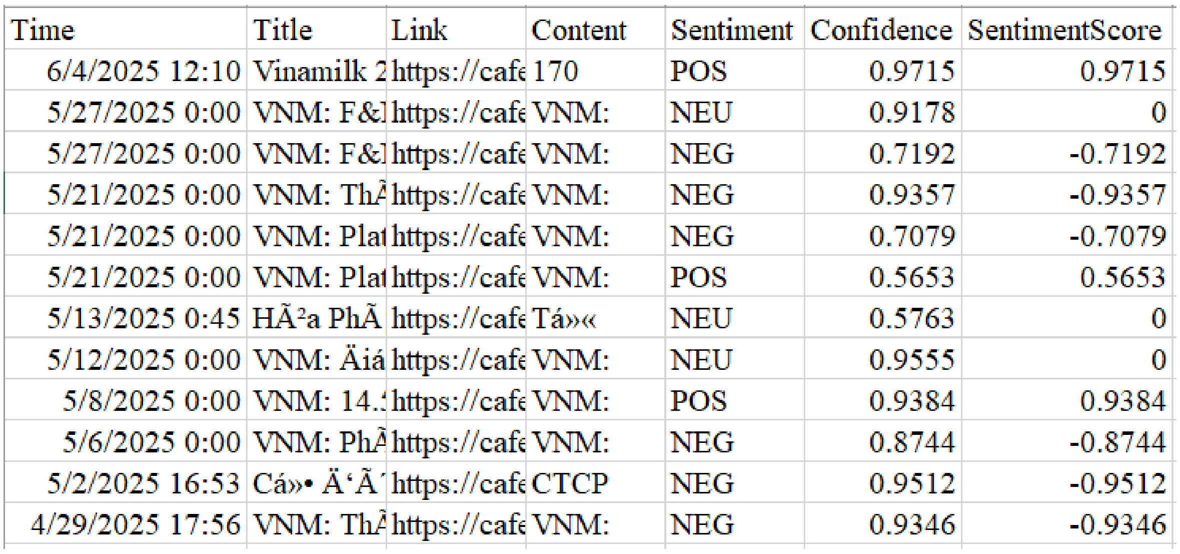
The final sentiment label and confidence score were combined into a single numeric value called SentimentScore. A positive label kept the score equal to the confidence, a negative label made it negative, and neutral or unknown labels were set to zero. This sign convention allowed daily sentiment to be easily averaged, turning raw text into usable numeric input for forecasting.

The scoring function was then applied to all new news articles for each VN30 stock. Existing sentiment files were checked before each run to avoid reprocessing old data. Only new articles with unseen links and newer timestamps were scored.

After scoring, three new columns: sentiment label, confidence, and SentimentScore were added. The new data was merged with existing sentiment files, duplicate entries were removed, and rows were sorted by date in descending order for easier tracking.

Common issues such as extra spaces in links and missing timestamps were resolved through text cleaning and flexible date parsing. Memory handling was also optimized to maintain stability when processing large files.

As a result, each stock was paired with a clean, complete sentiment dataset, ready for merging with price data in the next step.



Time	Title	Link	Content	Sentiment	Confidence	SentimentScore
6/4/2025 12:10	Vinamilk 2	https://cafe	170	POS	0.9715	0.9715
5/27/2025 0:00	VNM: F&I	https://cafe	VNM:	NEU	0.9178	0
5/27/2025 0:00	VNM: F&I	https://cafe	VNM:	NEG	0.7192	-0.7192
5/21/2025 0:00	VNM: ThÃ	https://cafe	VNM:	NEG	0.9357	-0.9357
5/21/2025 0:00	VNM: Plat	https://cafe	VNM:	NEG	0.7079	-0.7079
5/21/2025 0:00	VNM: Plat	https://cafe	VNM:	POS	0.5653	0.5653
5/13/2025 0:45	HÃ²a PhÃ	https://cafe	Tá»«	NEU	0.5763	0
5/12/2025 0:00	VNM: Äiá	https://cafe	VNM:	NEU	0.9555	0
5/8/2025 0:00	VNM: 14.:	https://cafe	VNM:	POS	0.9384	0.9384
5/6/2025 0:00	VNM: PhÃ	https://cafe	VNM:	NEG	0.8744	-0.8744
5/2/2025 16:53	Cá»• Ä‘Ã´	https://cafe	CTCP	NEG	0.9512	-0.9512
4/29/2025 17:56	VNM: ThÃ	https://cafe	VNM:	NEG	0.9346	-0.9346

Figure 3.7 Example of sentiment analysis results for Vinamilk news articles

The attributes are:

Table 3.3 Attributes of the news article sentiment analysis data

Attribute	**Description**
Time	Time the news was posted, including date and time
Title	News headline
Link	The full URL linking to the original news article on CafeF
Content	The main text content of the article
Sentiment	The predicted sentiment label for the article: POS (positive), NEG (negative), or NEU (neutral)
Confidence	The probability value showing how confident the model is about the sentiment label (range: 0 - 1)
SentimentScore	The final numeric sentiment score: equals confidence if POS, negative confidence if NEG, zero if NEU

3.4. Data mapping

3.4.1. Merging sentiment with price data to complete dataset for each stock

After the sentiment scores were calculated for all news articles, the next step was to link this information with the daily trading data of each stock. This step was essential to form a complete and structured dataset that combined both technical indicators and market sentiment in a consistent way.

First, all news articles were grouped by publishing date for each stock. The average sentiment score was then calculated for each day. To keep the dataset clean and consistent, all date and time fields were standardized. The trading dates, originally stored in day-month-year format, and the news timestamps, recorded in full datetime format, were both converted to the same date format using the pandas library in Python.

To avoid look-ahead bias, the daily sentiment score was shifted forward by one day. This ensured that each price record was only influenced by information that would have been available at the time, which reflects a more realistic trading scenario.

After that, the shifted daily sentiment was merged with the corresponding daily OHLC price data using the TradingDate as the key. For trading days without any news coverage, the missing sentiment score was filled with zero. This step prevented any null values from appearing in the final dataset, which could otherwise affect the performance of forecasting models or cause errors in data visualizations.

Each merged dataset was then sorted in descending order by date, so the most recent records appeared first. Duplicate entries were checked and removed. The final result for each stock was saved as a separate file named symbol_data.csv, which included all trading indicators and the matched daily sentiment score.

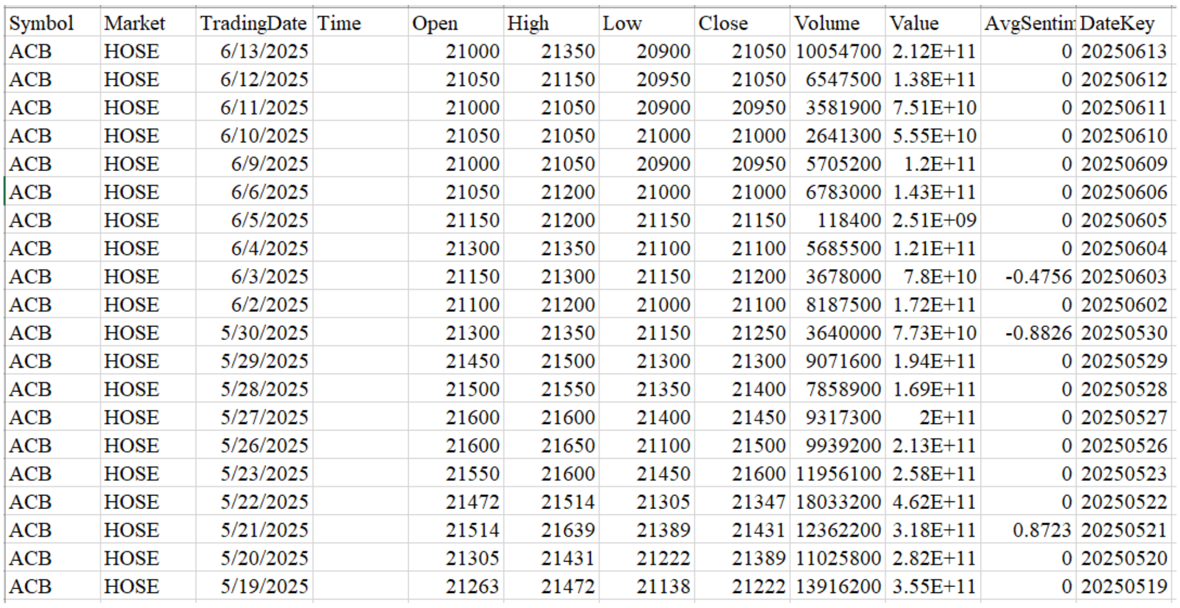
3.4.2. Building the complete fact tables

Once the merged datasets were created for all stocks, two main fact tables were built for the study. The first was the price and sentiment table, called fact_vn30_data. Each stock's merged data was loaded, and a new column named Symbol was added to identify the stock. A DateKey in the format YYYYMMDD was also created. This format follows best practices in data warehousing, making it easier and faster to perform joins in tools like Power BI. All individual stock files were then combined into a single unified table containing daily prices, technical indicators, sentiment scores, stock symbols, and clean date keys.

The second table was called fact_vn30_news. This table was built by combining all sentiment-scored news files across all VN30 stocks. Time fields were cleaned, and columns for Symbol and DateKey were added. This news table preserved the exact time, title, link, sentiment label, confidence score, and sentiment score for each article. Thanks to the DateKey, this table can be easily joined with other datasets in Power BI. From here, a semantic model can be built to support future dashboards that allow multi-dimensional analysis of sentiment and price trends across all VN30 companies.

3.4.3. Results

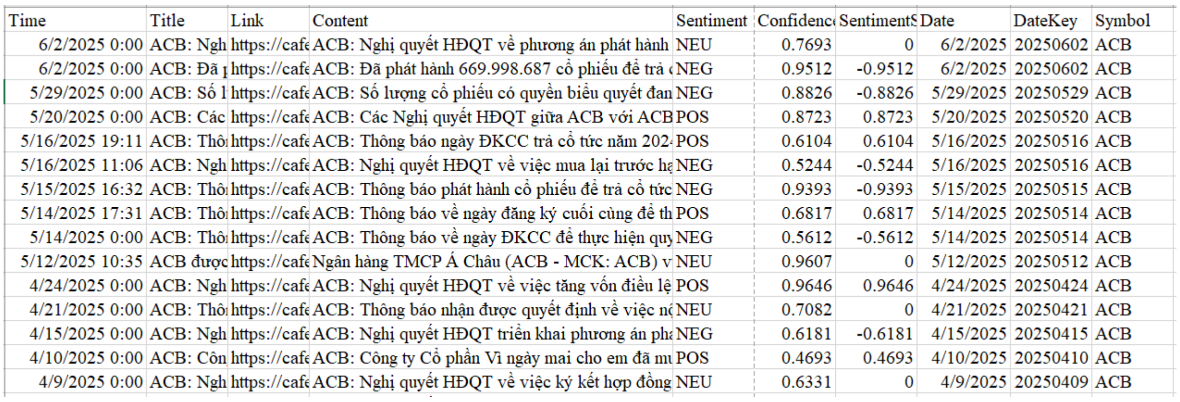
The final products in this step are two fact tables. Fact_vn30_data table combines daily trading prices with the daily average sentiment for each stock. It includes technical columns like Open, High, Low, Close, Volume, Value, the shifted sentiment score, a clear Symbol column, and a numeric DateKey.



Symbol	Market	TradingDate	Time	Open	High	Low	Close	Volume	Value	AvgSentin	DateKey
ACB	HOSE	6/13/2025		21000	21350	20900	21050	10054700	2.12E+11	0	20250613
ACB	HOSE	6/12/2025		21050	21150	20950	21050	6547500	1.38E+11	0	20250612
ACB	HOSE	6/11/2025		21000	21050	20900	20950	3581900	7.51E+10	0	20250611
ACB	HOSE	6/10/2025		21050	21050	21000	21000	2641300	5.55E+10	0	20250610
ACB	HOSE	6/9/2025		21000	21050	20900	20950	5705200	1.2E+11	0	20250609
ACB	HOSE	6/6/2025		21050	21200	21000	21000	6783000	1.43E+11	0	20250606
ACB	HOSE	6/5/2025		21150	21200	21150	21150	118400	2.51E+09	0	20250605
ACB	HOSE	6/4/2025		21300	21350	21100	21100	5685500	1.21E+11	0	20250604
ACB	HOSE	6/3/2025		21150	21300	21150	21200	3678000	7.8E+10	-0.4756	20250603
ACB	HOSE	6/2/2025		21100	21200	21000	21100	8187500	1.72E+11	0	20250602
ACB	HOSE	5/30/2025		21300	21350	21150	21250	3640000	7.73E+10	-0.8826	20250530
ACB	HOSE	5/29/2025		21450	21500	21300	21300	9071600	1.94E+11	0	20250529
ACB	HOSE	5/28/2025		21500	21550	21350	21400	7858900	1.69E+11	0	20250528
ACB	HOSE	5/27/2025		21600	21600	21400	21450	9317300	2E+11	0	20250527
ACB	HOSE	5/26/2025		21600	21650	21100	21500	9939200	2.13E+11	0	20250526
ACB	HOSE	5/23/2025		21550	21600	21450	21600	11956100	2.58E+11	0	20250523
ACB	HOSE	5/22/2025		21472	21514	21305	21347	18033200	4.62E+11	0	20250522
ACB	HOSE	5/21/2025		21514	21639	21389	21431	12362200	3.18E+11	0.8723	20250521
ACB	HOSE	5/20/2025		21305	21431	21222	21389	11025800	2.82E+11	0	20250520
ACB	HOSE	5/19/2025		21263	21472	21138	21222	13916200	3.55E+11	0	20250519

Figure 3.8 Capture of fact_vn30_data table

Fact_vn30_news table contains every news article with its publication time, headline, link, model-predicted sentiment, confidence score, numeric sentiment score, plus the Symbol and DateKey for easy joins.



Time	Title	Link	Content	Sentiment	Confidenc	SentimentS	Date	DateKey	Symbol
6/2/2025 0:00	ACB: Ngh	https://cafe	ACB: Nghị quyết HĐQT về phương án phát hành	NEU	0.7693	0	6/2/2025	20250602	ACB
6/2/2025 0:00	ACB: Đã p	https://cafe	ACB: Đã phát hành 669.998.687 cổ phiếu để trả c	NEG	0.9512	-0.9512	6/2/2025	20250602	ACB
5/29/2025 0:00	ACB: Số l	https://cafe	ACB: Số lượng cổ phiếu có quyền biểu quyết đan	NEG	0.8826	-0.8826	5/29/2025	20250529	ACB
5/20/2025 0:00	ACB: Các	https://cafe	ACB: Các Nghị quyết HĐQT giữa ACB với ACB	POS	0.8723	0.8723	5/20/2025	20250520	ACB
5/16/2025 19:11	ACB: Thôi	https://cafe	ACB: Thông báo ngày ĐKCC trả cổ tức năm 202	POS	0.6104	0.6104	5/16/2025	20250516	ACB
5/16/2025 11:06	ACB: Ngh	https://cafe	ACB: Nghị quyết HĐQT về việc mua lại trước hạ	NEG	0.5244	-0.5244	5/16/2025	20250516	ACB
5/15/2025 16:32	ACB: Thôi	https://cafe	ACB: Thông báo phát hành cổ phiếu để trả cổ tức	NEG	0.9393	-0.9393	5/15/2025	20250515	ACB
5/14/2025 17:31	ACB: Thôi	https://cafe	ACB: Thông báo về ngày đăng ký cuối cùng để th	POS	0.6817	0.6817	5/14/2025	20250514	ACB
5/14/2025 0:00	ACB: Thôi	https://cafe	ACB: Thông báo về ngày ĐKCC để thực hiện quy	NEG	0.5612	-0.5612	5/14/2025	20250514	ACB
5/12/2025 10:35	ACB được	https://cafe	Ngân hàng TMCP Á Châu (ACB - MCK: ACB) v	NEU	0.9607	0	5/12/2025	20250512	ACB
4/24/2025 0:00	ACB: Ngh	https://cafe	ACB: Nghị quyết HĐQT về việc tăng vốn điều lệ	POS	0.9646	0.9646	4/24/2025	20250424	ACB
4/21/2025 0:00	ACB: Thôi	https://cafe	ACB: Thông báo nhận được quyết định về việc nợ	NEU	0.7082	0	4/21/2025	20250421	ACB
4/15/2025 0:00	ACB: Ngh	https://cafe	ACB: Nghị quyết HĐQT triển khai phương án phá	NEG	0.6181	-0.6181	4/15/2025	20250415	ACB
4/10/2025 0:00	ACB: Côn	https://cafe	ACB: Công ty Cổ phần Vì ngày mai cho em đã mu	POS	0.4693	0.4693	4/10/2025	20250410	ACB
4/9/2025 0:00	ACB: Ngh	https://cafe	ACB: Nghị quyết HĐQT về việc ký kết hợp đồng	NEU	0.6331	0	4/9/2025	20250409	ACB

Figure 3.9 Capture of fact_vn30_news table

All missing values were handled properly, date formats were standardized, and duplicate rows were removed. These technical steps ensure the data is ready for building forecasting models and professional BI dashboards. By connecting price movements with daily market sentiment in this way, the thesis provides a well-structured and rich dataset that helps better explain and predict how news can move stock prices.

3.5. Predictive models training

After have completed and well-organized dataset, move to the next phase of the research is to build machine learning models to solve the main problem: forecast the price movements of stocks in VN30 list.

3.5.1. Model selection

Choosing the right models is a very important step in this study because different models can capture different patterns in the stock market data. In my literature review, I saw that many researchers have tested both traditional machine learning and deep learning models for forecasting stock prices. For this reason, I decided to test three different types of models in my pipeline: Ridge Regression, BiGRU, and LSTM.

Firstly, Ridge Regression is a good representative of a classic machine learning method. It is a linear regression model with L2 regularization. The main reason for using Ridge is that it is simple, fast, and works well when the relationship between features and the target is mostly linear but can have noise. It also helps control overfitting by shrinking less important feature weights. In stock forecasting, many studies have used Ridge or other linear models as a baseline to compare with more complex models. So, in my project, Ridge acts as a solid benchmark to see how much improvement deep learning can bring.

Secondly, I chose BiGRU, which stands for Bidirectional Gated Recurrent Unit. This model is a type of Recurrent Neural Network (RNN). RNNs are popular for time series because they can learn patterns that happen over time. The GRU is an improved version of the standard RNN because it solves the vanishing gradient problem better. The bidirectional setup lets the network learn from both past and future context in the training data. In finance, many papers have shown that BiGRU can model stock prices well because it can handle both trend and sudden jumps. This inspired me to include BiGRU to see how it can handle the sequential and sentiment-rich VN30 data.

Thirdly, I added the LSTM model. LSTM stands for Long Short-Term Memory and is another advanced type of RNN. LSTM can remember long-term patterns by using memory cells and gates, which makes it very strong for long sequences. Many studies, like (Fischer and Krauss, 2018), have shown that LSTM can forecast stock prices better than traditional models because it learns both short-term and long-term dependencies in the data. This makes LSTM a good choice for VN30 stocks, which often have complex price movements influenced by both old trends and new events. By using LSTM in my system, I want to compare if its memory power brings a clear benefit over Ridge and BiGRU for my dataset.

In short, my model selection covers three levels: a simple linear model (Ridge), a smart sequential model (BiGRU), and a powerful deep learning model for long-term patterns (LSTM). All three models use both technical indicators and my sentiment scores as input. This way, I can test if adding sentiment helps each model and find out which one is the best choice for forecasting VN30 stock prices.

3.5.2. Ridge Regression

In this study, Ridge is the first model to be trained. It works well when there is a mostly linear relationship between the input features and the target price but some randomness or small shocks may exist. For stock prices, which often change suddenly, this is a safe choice to check if more complex models really add extra value.

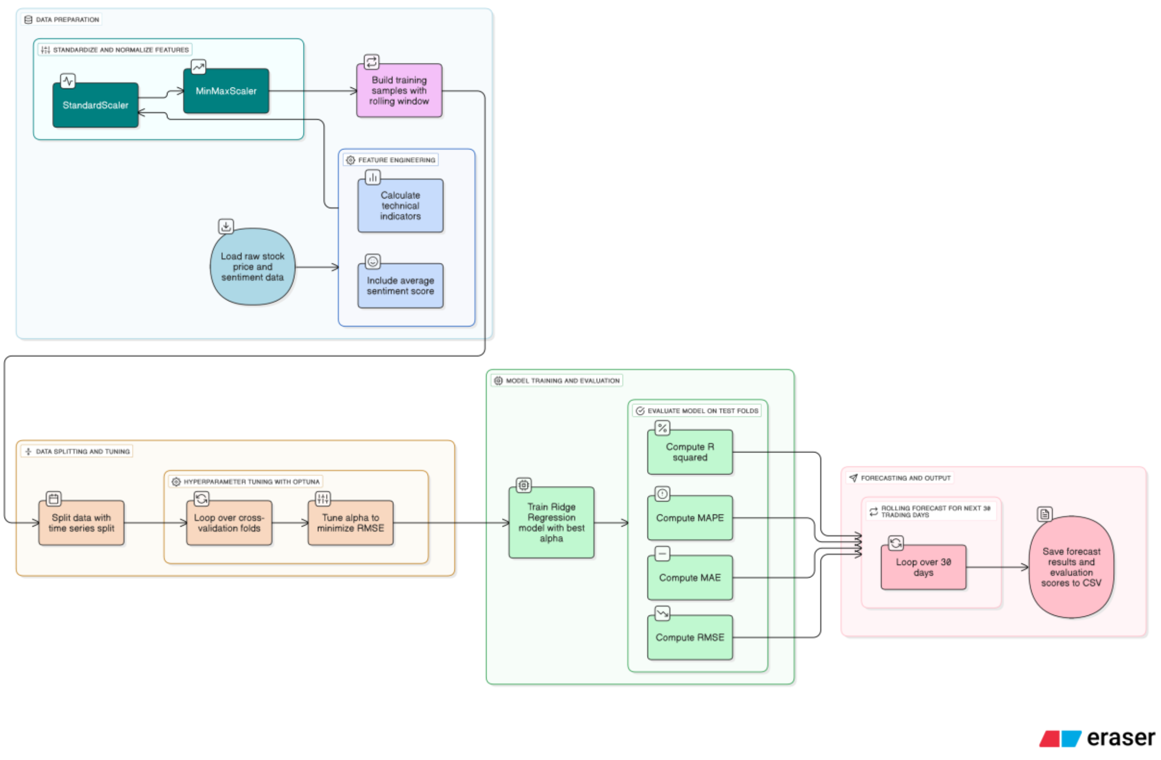

DATA PREPARATION
STANDARDIZE AND NORMALIZE FEATURES
StandardScaler
MinMaxScaler
Build training samples with rolling window
FEATURE ENGINEERING
Calculate technical indicators
Load raw stock price and sentiment data
Include average sentiment score
DATA SPLITTING AND TUNING
Split data with time series split
HYPERPARAMETER TUNING WITH OPTUNA
Loop over cross-validation folds
Tune alpha to minimize RMSE
MODEL TRAINING AND EVALUATION
Train Ridge Regression model with best alpha
EVALUATE MODEL ON TEST FOLDS
Compute R squared
Compute MAPE
Compute MAE
Compute RMSE
FORECASTING AND OUTPUT
ROLLING FORECAST FOR NEXT 30 TRADING DAYS
Loop over 30 days
Save forecast results and evaluation scores to CSV
eraser

Figure 3.10 Ridge Regression processes flowchart

Before training Ridge, a lot of time spent designing the input features. I did not just use basic columns like Open, High, Low, Close, and Volume. I added more features to help the model see more patterns.

Table 3.4 Engineered features description

Feature	Formula	Description
PriceChange	Close – Open	Shows the intraday price movement, indicating whether the stock gained or lost value during the trading day.
High_Low_Spread	High – Low	Measures the volatility of the trading day by capturing the range between the highest and lowest prices.
RollingMean_5	5-day rolling mean of Close	Smooths short-term fluctuations and helps identify the short-term trend direction.
Momentum_10	Close – Close.shift(10)	Measures the price momentum over the past 10 days, indicating upward or downward acceleration.
VolumeChange	(Current Volume - Previous Volume) / Previous Volume	Shows the percentage change in trading volume, which may reflect changes in investor interest or sentiment.
EMA_12	12-period Exponential Moving Average of Close	Captures recent price trends with more weight on recent prices, useful for identifying short-term momentum.
EMA_26	26-period Exponential Moving Average of Close	Provides a longer-term trend signal compared to EMA_12.
MACD	EMA_12 - EMA_26	A momentum indicator that shows the difference between two EMAs to signal trend strength and direction.
AvgSentimentScore	Average Sentiment Score of News per Day	Reflects the daily investor mood based on news sentiment, where positive values suggest optimism and negative values suggest concern.

By combining both technical signals and sentiment, I made sure Ridge has enough information.

One limitation of Ridge Regression is its sensitivity to feature scales. For example, Volume can have very large values, while sentiment scores typically range between minus one and one. To address this, double scaling was applied. First, StandardScaler was used to standardize the data, then MinMaxScaler was applied to bring all features into the [0, 1] range. This helped the Ridge model train more stably and prevented any single feature from dominating the learning process.

Training samples were created using a rolling window of 60 days, with the target defined as the average closing price over the following few days. Averaging helped smooth out sharp price jumps and reduced noise from outliers. Since time series data cannot be randomly shuffled, TimeSeriesSplit was used to divide the dataset in chronological order.

To tune the regularization strength, Optuna was used to search for the best alpha value. A wide range from very small (1e-5) to large (10.0) was explored, and the value that produced the lowest average RMSE across folds was selected. Once the best alpha was found, Ridge was retrained and tested. Predictions were inverse transformed to actual price units, and four evaluation metrics were calculated: R^2, RMSE, MAE, and MAPE.

For forecasting, a dynamic rolling approach was applied. Starting with the last 80 days of data, the next day's price was predicted, added to the input, and the process was repeated for 30 trading days. This method closely follows how predictions are updated in real-world trading. The final forecasts were saved to vn30_forecast_next30days.csv, and evaluation results were stored in vn30_ridge_evaluation.csv.

In short, Ridge Regression served as a solid baseline. With careful feature engineering, scaling, and step-by-step forecasting, its performance was maximized before being compared with more advanced deep learning models like BiGRU and LSTM.

3.5.3. Bidirectional Gated Recurrent Units (BiGRU)

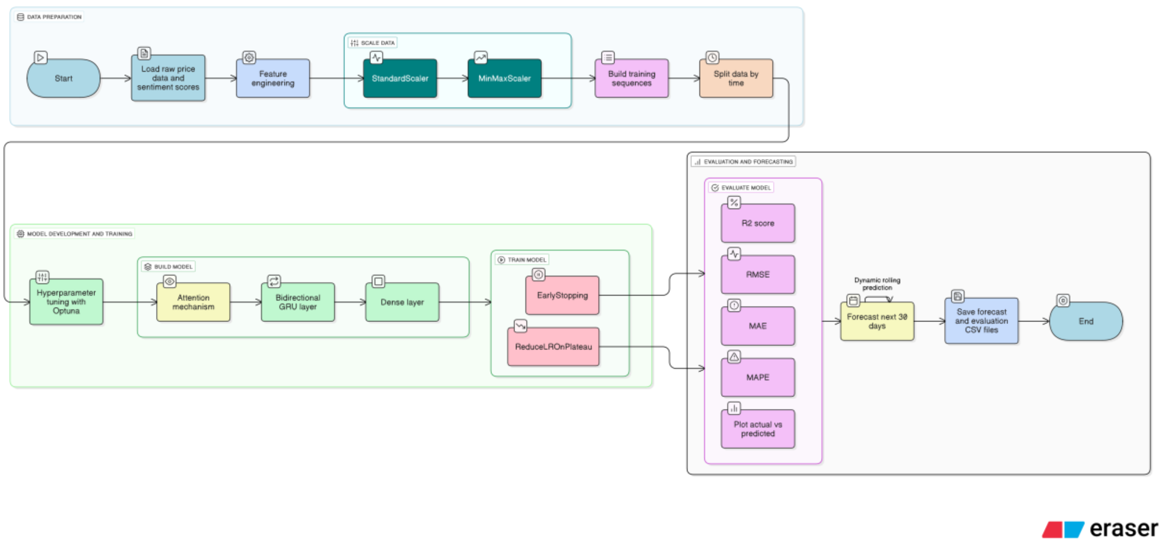


Figure 3.11 BiGRU processes flowchart

In this thesis, a BiGRU model with an Attention mechanism was trained to evaluate how deep learning could improve forecasting for VN30 stocks.

To highlight which time steps contributed most to the prediction, an Attention layer was added. This mechanism assigned weights to the BiGRU outputs and formed a context vector, which was then passed to a Dense layer to predict the next closing price.

The same engineered features used in Ridge were applied, including both technical indicators and sentiment scores. StandardScaler and MinMaxScaler were used in sequence to normalize the data and keep training stable.

Input sequences were generated using a sliding window of 60 time steps to predict the closing price one day ahead. The data was split chronologically, with 80 percent for training and 20 percent for testing.

To tune performance, Optuna was used to search for optimal values of GRU units, dropout rate, and learning rate. Training was managed with EarlyStopping and ReduceLROnPlateau to prevent overfitting and adjust learning rate when needed.

After training, predictions were inverse transformed into real prices. The model was evaluated using R^2, RMSE, MAE, and MAPE. An actual versus predicted chart was also created to compare trends visually.

For future forecasts, a rolling prediction loop was applied for 30 trading days. Starting with the last 60 days, the next day's price was predicted, added to the sequence, and the process was repeated. Volume was adjusted slightly with random noise to reflect market variation. Forecasts were saved in vn30_biGRU_tuned_forecast.csv, and evaluation scores in vn30_biGRU_tuned_evaluation.csv.

Overall, BiGRU with Attention provided a more advanced approach than Ridge by modeling sequence context and highlighting important signals, leading to better accuracy when combined with rich features.

3.5.4. Long Short-Term Memory (LSTM)

For the third model, LSTM was selected due to its strong performance on sequential data. LSTM is capable of learning both short-term and long-term patterns, which makes it suitable for forecasting stock prices that are influenced by both recent events and longer market trends.

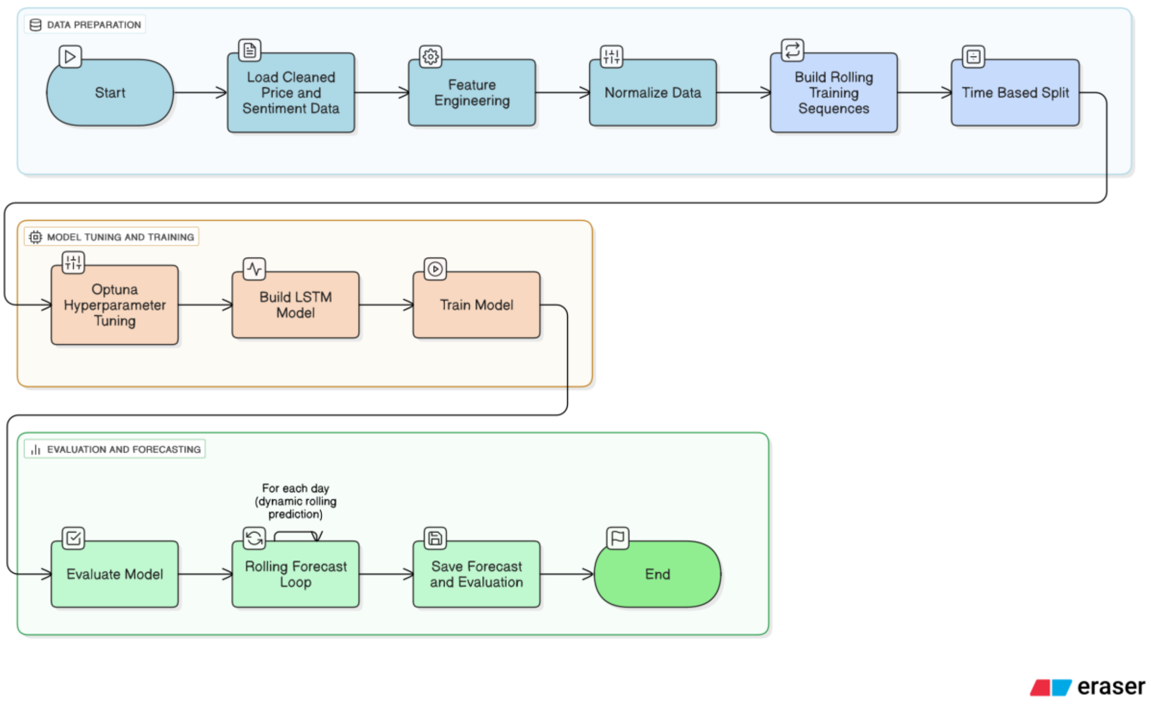

DATA PREPARATION
Start
Load Cleaned Price and Sentiment Data
Feature Engineering
Normalize Data
Build Rolling Training Sequences
Time Based Split
MODEL TUNING AND TRAINING
Optuna Hyperparameter Tuning
Build LSTM Model
Train Model
EVALUATION AND FORECASTING
For each day (dynamic rolling prediction)
Evaluate Model
Rolling Forecast Loop
Save Forecast and Evaluation
End
eraser

Figure 3.12 LSTM processes flowchart

The same engineered features used in Ridge and BiGRU were applied to ensure fairness in comparison. Data was scaled in two steps: first using StandardScaler for standardization, then MinMaxScaler to fit all values into the [0, 1] range. This double scaling improved stability during training and kept the input balanced across features.

Input sequences were prepared using a rolling window of 60 days to predict the closing price one day ahead. Data was split in time order, with 80% used for training and 20% for testing, to reflect real-world forecasting conditions.

Hyperparameters were tuned using Optuna. The number of LSTM units, dropout rate, and learning rate were optimized within practical ranges to balance model complexity and training efficiency. The final network consisted of an LSTM layer, a Dropout layer to prevent overfitting, and a Dense layer to output the predicted price. More units mean the model can learn more complex patterns, but too many can cause overfitting and slower training. Meanwhile, dropout tells how much of the LSTM's outputs are randomly dropped during training. This helps the model generalize better and prevents it from memorizing noise.

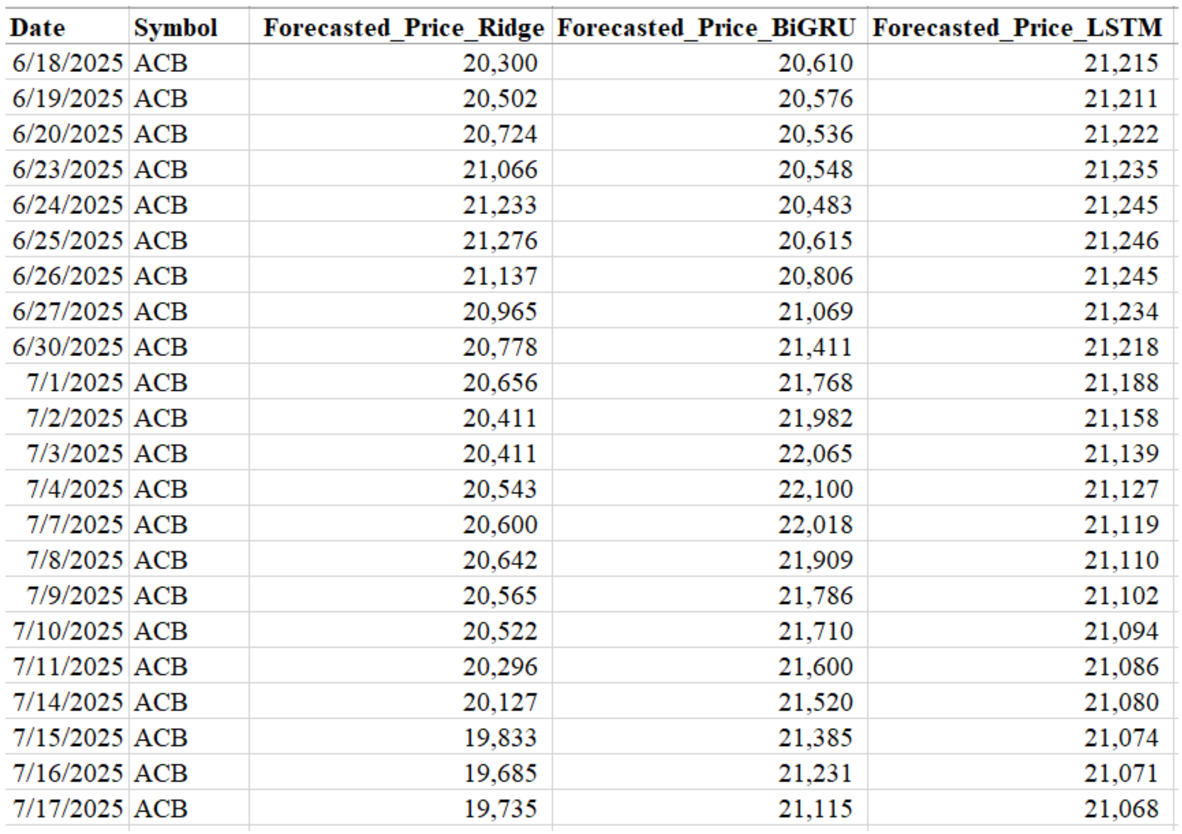
Training was supported by two callbacks: EarlyStopping to halt training when validation loss stopped improving, and ReduceLROnPlateau to lower the learning rate when progress slowed. After training, predictions were inverse transformed to VNĐ prices and evaluated using R^2, RMSE, MAE, and MAPE.

The forecasting phase used a dynamic rolling method for 30 future trading days. Starting from the last 60 known days, the next day was predicted, then added back into the sequence. Features were recalculated, including a slight random variation on Volume to mimic real market behavior. This process was repeated until all 30 forecasts were generated.

Final results were saved in vn30_lstm_tuned_forecast.csv and vn30_lstm_tuned_evaluation.csv. These outputs made it easy to compare LSTM with Ridge and BiGRU. Overall, each part of the LSTM pipeline from feature preparation to step-by-step forecasting was carefully designed to test how deep learning performs in the VN30 market.

3.5.5. Models results evaluation and select the best model

3.5.5.1. Model results example



Date	Symbol	Forecasted_Price_Ridge	Forecasted_Price_BiGRU	Forecasted_Price_LSTM
6/18/2025	ACB	20,300	20,610	21,215
6/19/2025	ACB	20,502	20,576	21,211
6/20/2025	ACB	20,724	20,536	21,222
6/23/2025	ACB	21,066	20,548	21,235
6/24/2025	ACB	21,233	20,483	21,245
6/25/2025	ACB	21,276	20,615	21,246
6/26/2025	ACB	21,137	20,806	21,245
6/27/2025	ACB	20,965	21,069	21,234
6/30/2025	ACB	20,778	21,411	21,218
7/1/2025	ACB	20,656	21,768	21,188
7/2/2025	ACB	20,411	21,982	21,158
7/3/2025	ACB	20,411	22,065	21,139
7/4/2025	ACB	20,543	22,100	21,127
7/7/2025	ACB	20,600	22,018	21,119
7/8/2025	ACB	20,642	21,909	21,110
7/9/2025	ACB	20,565	21,786	21,102
7/10/2025	ACB	20,522	21,710	21,094
7/11/2025	ACB	20,296	21,600	21,086
7/14/2025	ACB	20,127	21,520	21,080
7/15/2025	ACB	19,833	21,385	21,074
7/16/2025	ACB	19,685	21,231	21,071
7/17/2025	ACB	19,735	21,115	21,068

Figure 3.13 ACB stock price forecasted by 3 models

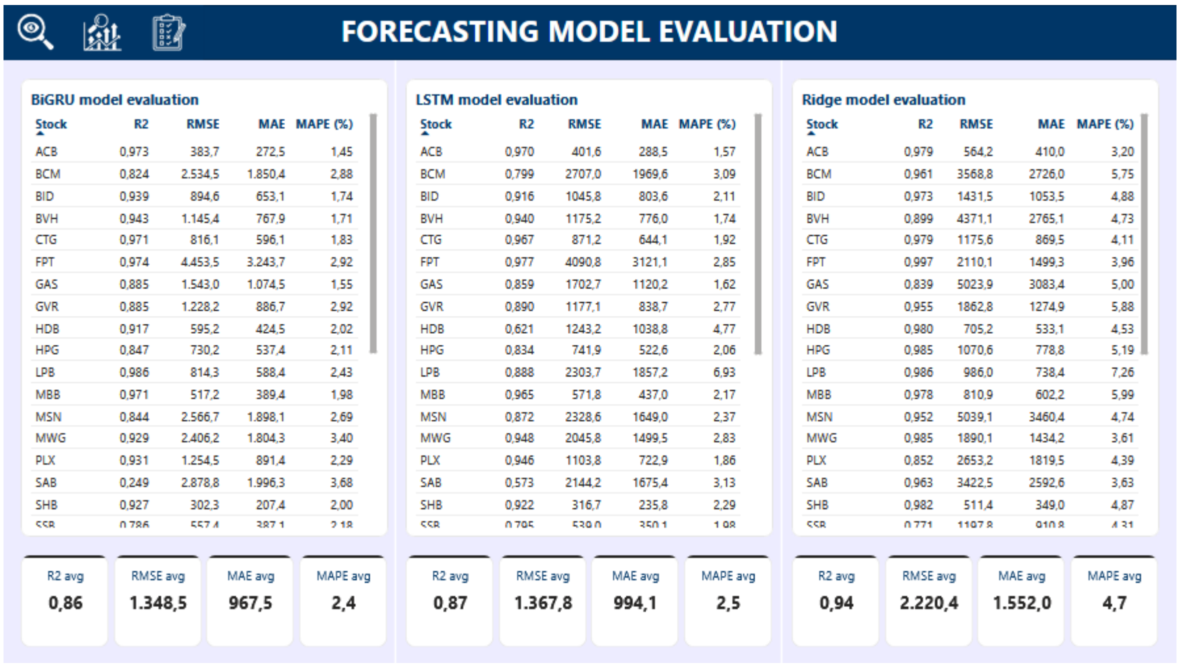
The figure above shows the forecasted closing prices of ACB stock from June 18 to July 17, 2025, generated by Ridge, BiGRU, and LSTM. The results show clear differences in prediction style, sensitivity, and signal interpretation across the three models.

Ridge Regression produced stable and smooth forecasts, but its limited flexibility made it less responsive to short-term market shifts. The model showed a downward bias in the later period, potentially missing recovery signals.

BiGRU with Attention captured sharp movements and reacted strongly to recent patterns. While it showed high sensitivity, the forecasts appeared slightly over-volatile, which may reduce reliability in calmer market conditions.

LSTM offered a balanced performance, maintaining smooth transitions while still adapting to new trends. Its predictions remained consistent across the forecast window, making it a practical choice for daily stock forecasting.

3.5.5.2. Best model selection


FORECASTING MODEL EVALUATION

BiGRU model evaluation

Stock	R2	RMSE	MAE	MAPE (%)
ACB	0,973	383,7	272,5	1,45
BCM	0,824	2.534,5	1.850,4	2,88
BID	0,939	894,6	653,1	1,74
BVH	0,943	1.145,4	767,9	1,71
CTG	0,971	816,1	596,1	1,83
FPT	0,974	4.453,5	3.243,7	2,92
GAS	0,885	1.543,0	1.074,5	1,55
GVR	0,885	1.228,2	886,7	2,92
HDB	0,917	595,2	424,5	2,02
HPG	0,847	730,2	537,4	2,11
LPB	0,986	814,3	588,4	2,43
MBB	0,971	517,2	389,4	1,98
MSN	0,844	2.566,7	1.898,1	2,69
MWG	0,929	2.406,2	1.804,3	3,40
PLX	0,931	1.254,5	891,4	2,29
SAB	0,249	2.878,8	1.996,3	3,68
SHB	0,927	302,3	207,4	2,00
SSB	0,786	557,4	387,1	2,18

R2 avg	RMSE avg	MAE avg	MAPE avg
0,86	1.348,5	967,5	2,4

LSTM model evaluation

Stock	R2	RMSE	MAE	MAPE (%)
ACB	0,970	401,6	288,5	1,57
BCM	0,799	2707,0	1969,6	3,09
BID	0,916	1045,8	803,6	2,11
BVH	0,940	1175,2	776,0	1,74
CTG	0,967	871,2	644,1	1,92
FPT	0,977	4090,8	3121,1	2,85
GAS	0,859	1702,7	1120,2	1,62
GVR	0,890	1177,1	838,7	2,77
HDB	0,621	1243,2	1038,8	4,77
HPG	0,834	741,9	522,6	2,06
LPB	0,888	2303,7	1857,2	6,93
MBB	0,965	571,8	437,0	2,17
MSN	0,872	2328,6	1649,0	2,37
MWG	0,948	2045,8	1499,5	2,83
PLX	0,946	1103,8	722,9	1,86
SAB	0,573	2144,2	1675,4	3,13
SHB	0,922	316,7	235,8	2,29
SSB	0,795	539,0	350,1	1,98

R2 avg	RMSE avg	MAE avg	MAPE avg
0,87	1.367,8	994,1	2,5

Ridge model evaluation

Stock	R2	RMSE	MAE	MAPE (%)
ACB	0,979	564,2	410,0	3,20
BCM	0,961	3568,8	2726,0	5,75
BID	0,973	1431,5	1053,5	4,88
BVH	0,899	4371,1	2765,1	4,73
CTG	0,979	1175,6	869,5	4,11
FPT	0,997	2110,1	1499,3	3,96
GAS	0,839	5023,9	3083,4	5,00
GVR	0,955	1862,8	1274,9	5,88
HDB	0,980	705,2	533,1	4,53
HPG	0,985	1070,6	778,8	5,19
LPB	0,986	986,0	738,4	7,26
MBB	0,978	810,9	602,2	5,99
MSN	0,952	5039,1	3460,4	4,74
MWG	0,985	1890,1	1434,2	3,61
PLX	0,852	2653,2	1819,5	4,39
SAB	0,963	3422,5	2592,6	3,63
SHB	0,982	511,4	349,0	4,87
SSB	0,771	1197,8	910,8	4,31

R2 avg	RMSE avg	MAE avg	MAPE avg
0,94	2.220,4	1.552,0	4,7

Figure 3.14 Models metrics comparison

Among the three models, Ridge Regression achieved the highest R² score at 0.94, showing strong trend fitting on paper. However, its large average errors RMSE of 2,220.4, MAE of 1,552.0, and MAPE of 4.7%, revealed a major weakness. In practice, these errors are too high for daily use, making Ridge unreliable for short-term forecasting despite its good fit to the overall trend.

BiGRU with Attention brought significant improvements. Although its R² dropped to 0.86, both RMSE (1,348.5) and MAE (967.5) were sharply lower than Ridge. Its MAPE of 2.4% suggests that BiGRU predictions are far more precise and stable on a daily basis. However, some forecasts appeared overly sensitive to short-term noise, which may reduce consistency in volatile conditions.

LSTM offered the best overall balance. With an R² of 0.87, slightly higher than BiGRU, and similarly low error scores (RMSE: 1,367.8, MAE: 994.1, MAPE: 2.5%), it matched BiGRU's accuracy while delivering smoother and more stable forecasts. Unlike BiGRU, LSTM avoided sudden jumps and followed the market trend more naturally. This made it easier to interpret and more reliable in real trading scenarios.

According to (Nguyen, Dang and Vu, 2025), the ARIMA model, a traditional method used to predict stock price, achieved an average MAPE of approximately 7.90% when applied to five VN30 constituent stocks. In comparison, the LSTM model developed in this thesis achieved a significantly lower average MAPE of 2.5% across all 30 VN30 stocks.

This result indicates that the LSTM model produced more accurate forecasts in relative terms. The improvement in MAPE highlights LSTM's advantage in capturing nonlinear patterns and incorporating sentiment features, making it more suitable for short-term stock price prediction under real market conditions.

Taken together, while Ridge was the fastest to train, its high errors limited its practicality. BiGRU captured complex patterns but was prone to overreaction.

LSTM, on the other hand, delivered consistent results with stable predictions and realistic trends, making it the most suitable model for forecasting VN30 stock prices in this study.

While the LSTM model demonstrated strong overall performance and stability across most stocks, its forecasting errors were mainly observed during periods of high volatility or unexpected price shifts. In general, LSTM tends to learn well from stable sequential patterns, but may struggle to react promptly when faced with sharp market movements, fast reversals, or event-driven anomalies. This reflects a limitation in the model's sensitivity to sudden behavioral changes, especially when such changes are not well represented in the training data or input features.

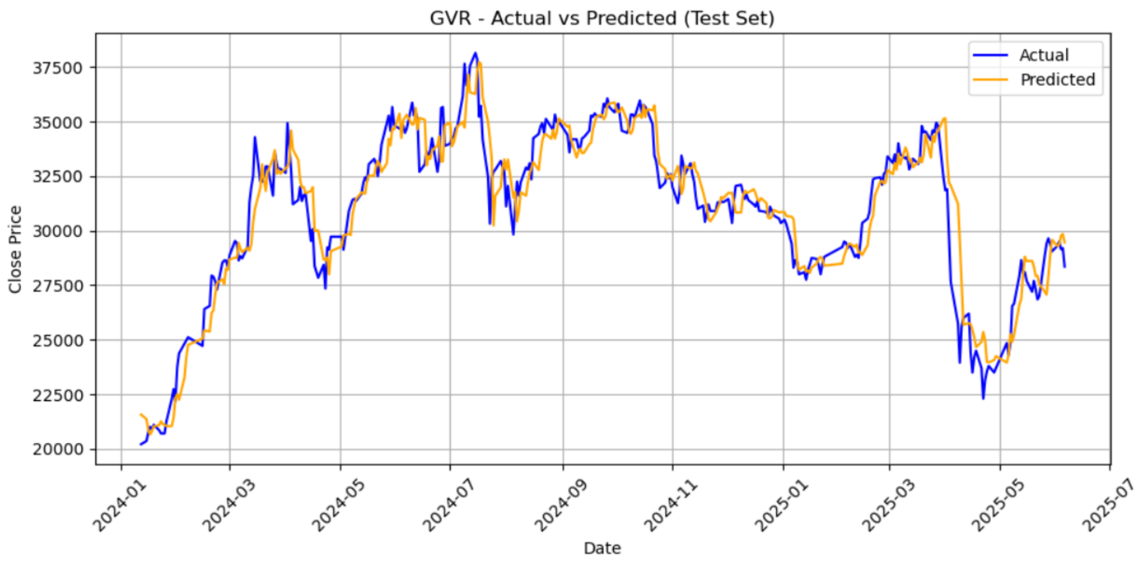


Figure 3.15 Actual vs Predicted Closing Prices for GVR using LSTM model

A clear example can be seen with the GVR stock. Around May 2025, the actual price experienced a sharp drop followed by a quick recovery. While the LSTM model successfully captured the overall direction, it underestimated the speed and scale of this movement, resulting in a noticeable lag. Similarly, in early 2024, GVR saw a strong upward trend over a short period, but the model produced a smoother curve that failed to fully reflect the sharp rise. These deviations suggest that LSTM, despite its robustness in trend prediction, may require further enhancement to better capture extreme or abrupt price dynamics, such as through volatility-based features or event-sensitive signals.

3.6. Deploying the results to an interface

After finishing all training and evaluation steps, I needed to put my results into an interactive format so users can check and understand them easily. For this purpose, I chose Power BI as the tool to build a simple but effective dashboard. Power BI helps turn my final data files into a clear visual report that users can view online, filter by stock, and follow daily.

3.6.1. Data model in Power BI

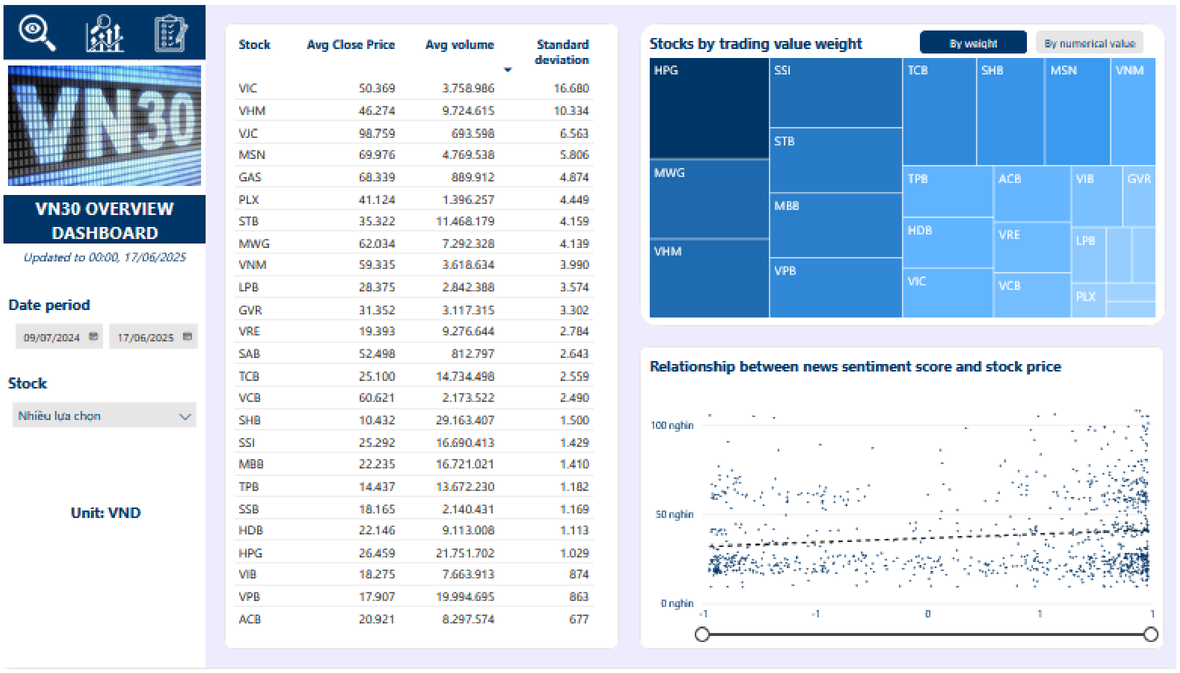
In Power BI, a clean data model was created to connect all parts of the pipeline. The main fact table, fact_vn30_data, stores daily prices and sentiment scores for each VN30 stock. It is linked to dim_date and dim_company through DateKey and Symbol, allowing filters by time and company.

The fact_vn30_news table was added to include headlines, content, sentiment labels, and confidence scores. It uses the same dimensions to display related news by trading day. Forecast files from LSTM, BiGRU, and Ridge were also imported, each containing Date, Symbol, and Forecasted_Price, and connected to the date and company tables.

Finally, evaluation tables with R^2, RMSE, MAE, and MAPE were linked, enabling quick comparison of model performance. Proper relationships were established among all tables, making it possible to build interactive dashboards from multiple analysis perspectives.

3.6.2. Showing the results

In the dashboard, the first page is called VN30 Overview. This page shows a quick view of all stocks. On the left side, I put slicers for Date period and Stock, so users can choose any time range and pick one or more stocks.



Stock	Avg Close Price	Avg volume	Standard deviation
VIC	50.369	3.758.986	16.680
VHM	46.274	9.724.615	10.334
VJC	98.759	693.598	6.563
MSN	69.976	4.769.538	5.806
GAS	68.339	889.912	4.874
PLX	41.124	1.396.257	4.449
STB	35.322	11.468.179	4.159
MWG	62.034	7.292.328	4.139
VNM	59.335	3.618.634	3.990
LPB	28.375	2.842.388	3.574
GVR	31.352	3.117.315	3.302
VRE	19.393	9.276.644	2.784
SAB	52.498	812.797	2.643
TCB	25.100	14.734.498	2.559
VCB	60.621	2.173.522	2.490
SHB	10.432	29.163.407	1.500
SSI	25.292	16.690.413	1.429
MBB	22.235	16.721.021	1.410
TPB	14.437	13.672.230	1.182
SSB	18.165	2.140.431	1.169
HDB	22.146	9.113.008	1.113
HPG	26.459	21.751.702	1.029
VIB	18.275	7.663.913	874
VPB	17.907	19.994.695	863
ACB	20.921	8.297.574	677

Figure 3.16 VN30 Overview Dashboard page

The main visuals are:

- A summary table showing each stock's risk, average closing price and volume. This helps users compare which stocks are more stable or more volatile.
- A tree map shows stocks by trading value weight or shows in numerical value as well.
- A scatter plot showing the relationship between news sentiment score and stock price over the last one year. In this chart, when the sentiment score is more positive, the price often moves slightly higher too. But the pattern is not very strong, so it suggests that sentiment can affect the market but is not the only factor.

The second page is the Stock Results page, which is the main part for deeper insight. This page also has slicers for Date period and Stock, so the user can look at each stock in detail.

Key visuals on this page include:

- A set of cards showing the current open, high and close prices and how they change compared to the start of the selected period.
- A candlestick chart showing the stock's historical daily prices.
- A column chart of daily trading volume.
- A table listing recent news headlines for that stock, along with sentiment labels and a simple icon to suggest if the news is positive or negative.

- Most importantly, a line chart displaying the predicted closing price for the next 30 trading days. This uses the LSTM forecast which is the best model as selected above. This visual helps the user quickly see the trend the model expects, which can support their planning or decision-making.

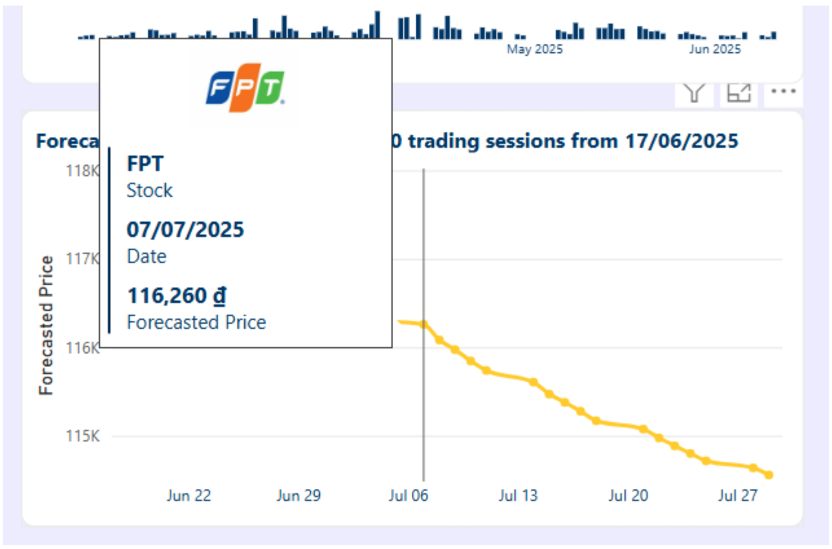


Figure 3.17 FPT stock forecasted price chart

This makes the dashboard interactive and practical for daily use. By combining actual prices, news impact and predicted trend, this report provides a clear and realistic view of the VN30 market based on both technical data and sentiment information.

3.7. Automated pipeline

After finished building all the previous steps, the next step is to set up everything to run by itself every day. So, I set up an automatic pipeline using Apache Airflow and made sure that my Power BI dashboard always shows the newest results.

3.7.1. Setup automated pipeline in Apache Airflow

Airflow was installed using Docker, as it allowed for easier setup and compatibility across different systems. The original code, which was first written in Jupyter Notebook, was later refactored into separate Python files. Each function, including price crawling, news crawling, sentiment analysis, data mapping, and LSTM training, was saved in a clearly named script within an organized folder structure.

A main pipeline script was created to define the DAG. Within this DAG, tasks were managed using PythonOperator and arranged in logical order: from data collection to sentiment scoring, data integration, and model training. Retry policies were configured to re-execute failed tasks after 10 minutes, and email alerts were enabled to support timely monitoring and debugging.

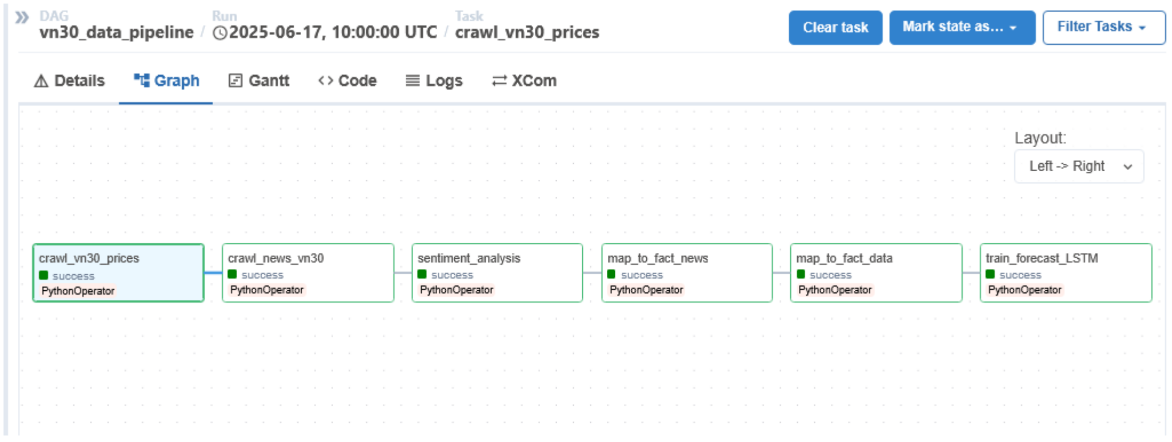

DAG
vn30_data_pipeline
Run
2025-06-17, 10:00:00 UTC
Task
crawl_vn30_prices
Clear task
Mark state as...
Filter Tasks
Details
Graph
Gantt
Code
Logs
XCom
Layout:
Left -> Right
crawl_vn30_prices success PythonOperator
crawl_news_vn30 success PythonOperator
sentiment_analysis success PythonOperator
map_to_fact_news success PythonOperator
map_to_fact_data success PythonOperator
train_forecast_LSTM success PythonOperator

Figure 3.18 Apache Airflow DAGs pipeline

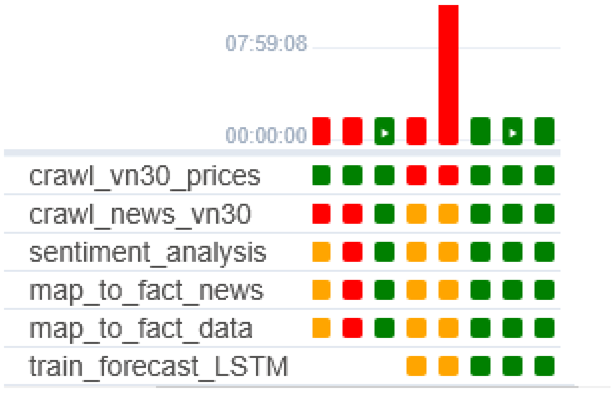


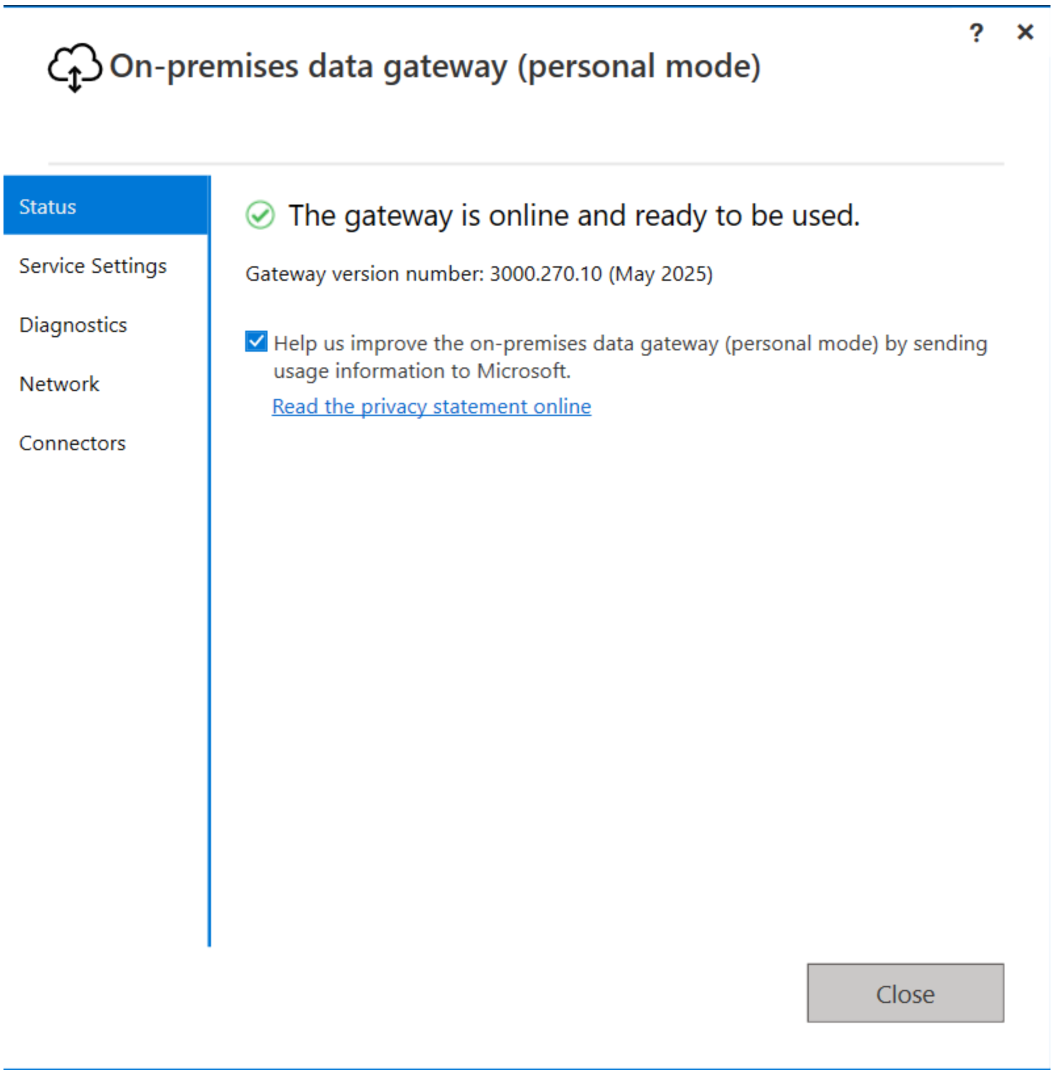
Figure 3.19 Task execution status of the VN30 forecasting pipeline in Airflow (Green means success)

3.7.2. Setup auto refresh for dashboard using Gateway

After the pipeline finishes, output files are automatically overwritten and linked to Power BI reports. To keep the dashboard updated without manual uploads, Power BI Gateway was installed in Personal Mode and connected to the Power BI Service account.

The Gateway acts as a secure bridge between local data files and the Power BI cloud. It allows the Power BI Service to access and refresh reports using files stored on the local machine, even when the report is opened online. A daily refresh schedule was set to run right after the Airflow pipeline completes.

Thanks to this setup, the dashboard always reflects the most recent stock prices, sentiment scores, and forecasted values, making the system fully automated and practical for daily monitoring.



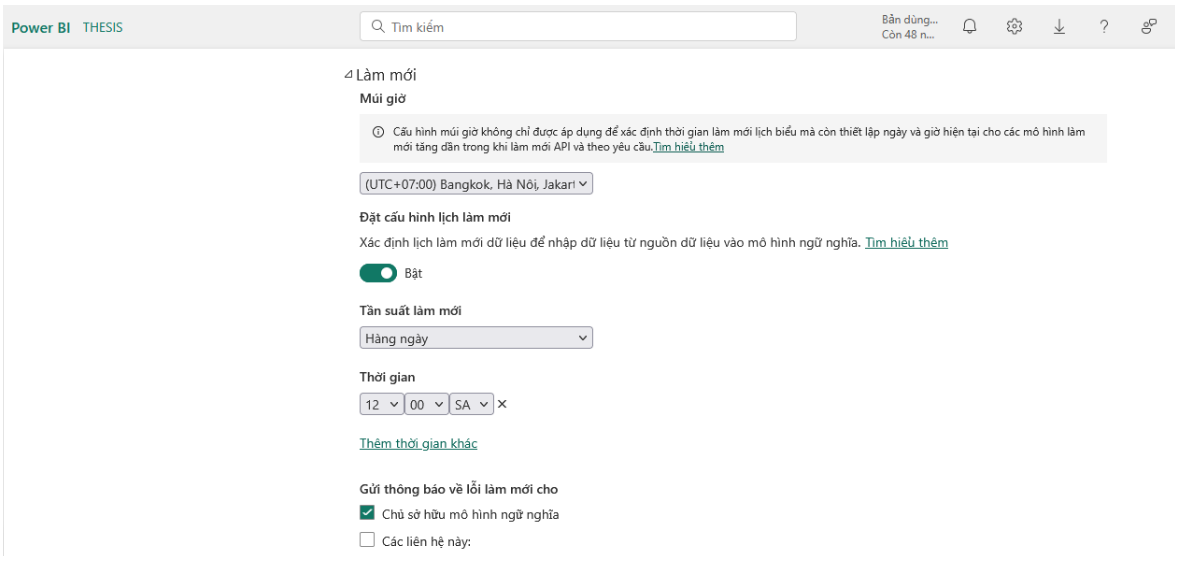
The gateway is online and ready to be used.

Gateway version number: 3000.270.10 (May 2025)

☑ Help us improve the on-premises data gateway (personal mode) by sending usage information to Microsoft.

Read the privacy statement online

Figure 3.20 On-premises data gateway set up



Làm mới

Múi giờ

Cấu hình múi giờ không chỉ được áp dụng để xác định thời gian làm mới lịch biểu mà còn thiết lập ngày và giờ hiện tại cho các mô hình làm mới tăng dần trong khi làm mới API và theo yêu cầu.Tìm hiểu thêm

(UTC+07:00) Bangkok, Hà Nội, Jakart

Đặt cấu hình lịch làm mới

Xác định lịch làm mới dữ liệu để nhập dữ liệu từ nguồn dữ liệu vào mô hình ngữ nghĩa. Tìm hiểu thêm

Bật

Tần suất làm mới

Hàng ngày

Thời gian

12 : 00 SA

Thêm thời gian khác

Gửi thông báo về lỗi làm mới cho

☑ Chủ sở hữu mô hình ngữ nghĩa

☐ Các liên hệ này:

Figure 3.21 Set up auto refresh in Power BI Service

CONCLUSION

Summary

In this thesis, a complete system was designed and built to forecast stock prices of VN30 companies. The system used two main types of data: daily historical trading prices and news sentiment scores.

First, the data was collected and cleaned using trusted APIs and web scraping scripts. Then, a sentiment analysis model based on PhoBERT was used to label each news article as positive, negative, or neutral. These sentiment labels were then matched with each trading day, so the forecasting models could learn both price trends and market sentiment.

Three models were developed and tested: Ridge Regression, BiGRU, and LSTM. All models were trained and evaluated on the same dataset. Their performance was compared using R^2, RMSE, MAE, and MAPE scores. The results showed that deep learning models performed better than Ridge Regression. Among them, LSTM was the most stable and accurate for this dataset.

To help users view and use the results easily, an interactive dashboard was created with Power BI. This dashboard showed historical price trends, daily sentiment, and predicted stock prices for the next 30 trading days. An automatic pipeline was also set up using Airflow and Power BI Gateway, so the system could update itself every day without manual work.

Limitations

Although my system runs well and shows useful results, there are still some points that need to improve. One limitation is about the input data. I only used a basic set of technical indicators and common features. To get even higher prediction accuracy, the system should include more complex technical factors like advanced momentum signals, volatility measures or market depth data.

Next, the sentiment model I used is good for short news titles but might not fully understand the deep context in very long or detailed articles. This means some subtle information may be missed when the sentiment score is calculated.

Finally, training the LSTM model every day for all VN30 stocks uses a lot of computer resources. To apply stronger optimization techniques or train bigger models, a more powerful machine with better CPU and GPU would need to be deployed. This could be a barrier if I want to expand the project in the future.

Future development

To make this system better in the future, I plan to add more advanced technical attributes to the input data. For example, I want to include features like ATR, RSI, Bollinger Bands or other signals that traders often use to read market movements.

I also plan to use a better sentiment analysis model. A version of PhoBERT fine-tuned for full news bodies or even a newer Transformer model could help the system understand the meaning of long articles more deeply and label sentiment more correctly.

Another important plan is to optimize the LSTM model further. I want to test ways to speed up training, use GPU more effectively, or try other architectures like BiLSTM with attention or hybrid models. This can help the system forecast prices even better without using too much computer power.

Finally, I hope to improve the Power BI dashboard. I want to add alerts and automatic signals so that users can get notified when the forecast shows a big change. This can help investors make decisions faster and use my system as a daily tool to track and plan their investments.

In short, this thesis shows a clear way to mix technical data, news sentiment, machine learning, and dashboards into one working system for the VN30 stocks. I believe this work can be a good base for future improvements and real-world use in Vietnam’s stock market.

REFERENCES

1. Akiba, T. *et al.* (2019) 'Optuna: A Next-generation Hyperparameter Optimization Framework', in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Available at: https://doi.org/10.1145/3292500.3330701.
2. Aston Zhang (2021) *Dive Into Deep Learning*, *Dive Into Deep Learning*. Available at: https://d2l.ai/chapter_recurrent-modern/lstm.html (Accessed: 10 June 2025).
3. Azman, S., Pathmanathan, D. and Balakrishnan, V. (2025) 'A two-stage forecasting model using random forest subset-based feature selection and BiGRU with attention mechanism: Application to stock indices', *Plos* [Preprint]. Available at: https://doi.org/https://doi.org/10.1371/journal.pone.0323015.
4. Ba, J.L. and Kingma, D.P. (2015) 'Adam: A method for stochastic optimization', in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*.
5. Fischer, T. and Krauss, C. (2018) 'Deep learning with long short-term memory networks for financial market predictions', *European Journal of Operational Research* [Preprint]. Available at: https://doi.org/10.1016/j.ejor.2017.11.054.
6. Gu, W. *et al.* (2024) 'Predicting Stock Prices with FinBERT-LSTM: Integrating News Sentiment Analysis', *ACM Digital Library* [Preprint]. Available at: https://doi.org/https://doi.org/10.48550/arXiv.2407.16150.
7. Hongyu, Z. (2025) 'Vietnam V30 Closing Price Forecast Based on ARIMA and ETS', *EWA Publishing* [Preprint]. Available at: https://doi.org/https://doi.org/10.54254/2754-1169/2024.GA19104.
8. Hyndman, R.J. *et al.* (2002) 'A state space framework for automatic forecasting using exponential smoothing methods', *International Journal of Forecasting* [Preprint]. Available at: https://doi.org/10.1016/S0169-2070(01)00110-8.
9. Jamdar, S. (2024) *Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) Networks: A Deep Dive*, *Medium*. Available at:

https://medium.com/@sarthakjamdar9561/recurrent-neural-networks-rnns-and-long-short-term-memory-lstm-networks-a-deep-dive-85817329d0a6 (Accessed: 10 June 2025).

10. Khalid Nazarov (2023) *Overview of the Steps in a Machine Learning Pipeline*, *Linkedin*. Available at: https://www.linkedin.com/pulse/overview-steps-machine-learning-pipeline-khalid-nazarov/ (Accessed: 1 June 2025).
11. Krauss, C., Do, X.A. and Huck, N. (2017) 'Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500', *European Journal of Operational Research* [Preprint]. Available at: https://doi.org/10.1016/j.ejor.2016.10.031.
12. *ML Introduction with scikit-learn* (2025) *Codefinity*. Available at: https://codefinity.com/courses/v2/a65bbc96-309e-4df9-a790-a1eb8c815a1c/d7c3c2cd-8cd0-4572-90df-64edaeefe42e/98249160-c222-423b-b5d3-f0f7257f92cb.
13. Nguyen, D.K. *et al.* (2021) 'Does short-term technical trading exist in the Vietnamese stock market?', *Borsa Istanbul Review* [Preprint]. Available at: https://doi.org/10.1016/j.bir.2020.05.005.
14. Nguyen, D.Q. and Nguyen, A.T. (2020) 'PhoBERT: Pre-trained language models for Vietnamese', in *Findings of the Association for Computational Linguistics Findings of ACL: EMNLP 2020*. Available at: https://doi.org/10.18653/v1/2020.findings-emnlp.92.
15. Nguyen, H.L. and Tetiana, P. (2024) 'Forecasting the VN30 Index: An Empirical Comparison Between ARIMA and ETS Models', *SIAM Journal of Applied Economics* [Preprint]. Available at: https://doi.org/http://dx.doi.org/10.31031/siam.2024.04.000598.
16. Nguyen, T.T.L., Dang, N.H. and Vu, T.T. Van (2025) 'Application of ARIMA model and deep learning in forecasting stock price in Vietnam', *Salud, Ciencia y Tecnología - Serie de Conferencias*, 4. Available at: https://doi.org/https://doi.org/10.56294/sctconf20251320.
17. Oak, O. *et al.* (2024) 'A Novel Multivariate Bi-LSTM model for Short-Term Equity Price Forecasting', *IEEE* [Preprint]. Available at: https://doi.org/https://doi.org/10.48550/arXiv.2409.14693.

18. Pascanu, R., Mikolov, T. and Bengio, Y. (2013) ‘On the difficulty of training recurrent neural networks’, in *30th International Conference on Machine Learning, ICML 2013*.
19. Prechelt, L. (1998) ‘Automatic early stopping using cross validation: Quantifying the criteria’, *Neural Networks* [Preprint]. Available at: https://doi.org/10.1016/S0893-6080(98)00010-0.
20. Sayed Fahim Ali Dawdye (2024) *Time Series Prediction with GRU*, *Code360*. Available at: https://www.naukri.com/code360/library/time-series-prediction-with-gru.
21. Smith, L.N. (2017) ‘Cyclical learning rates for training neural networks’, in *Proceedings - 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017*. Available at: https://doi.org/10.1109/WACV.2017.58.
22. Tun, N.S. *et al.* (2021) ‘Stock article title sentiment-based classification using PhoBERT’, in *CEUR Workshop Proceedings*.
23. Vu, L.T. *et al.* (2023) ‘Sentiments Extracted from News and Stock Market Reactions in Vietnam’, *International Journal of Financial Studies* [Preprint]. Available at: https://doi.org/10.3390/ijfs11030101.
24. Vuong, P.H. *et al.* (2022) ‘Stock-price forecasting based on XGBoost and LSTM’, *Computer Systems Science and Engineering* [Preprint]. Available at: https://doi.org/10.32604/CSSE.2022.017685.