Real time stock predcition using Kafka and Spark

Phan.T.Nhan, Phan.H.B.Khang, Vo.T.T.Phuc, Doan.V.Bach
University of Information and Technology,
Ho Chi Minh city, VN
{20520258,20520570,20520280,20520041}gm.uit.edu.vn

Abstract

Stock market prediction has been a subject of extensive research due to its potential implications for investors and financial institutions. This paper presents a comparative study of machine learning techniques for predicting stock prices. We explore the performance and effectiveness of various algorithms, including support vector machines, random forests, long short-term memory (LSTM) networks, and gradient boosting models.

Overall, this study contributes to the growing body of research in stock market prediction by offering a comprehensive comparison of machine learning techniques. The results highlight the efficacy of LSTM networks in capturing complex stock price patterns, while also underscoring the potential of ensemble methods and feature engineering for enhanced prediction accuracy. These insights can assist investors, financial analysts, and decision-makers in making more informed investment decisions and managing market risks.

1 Introduction

The ability to accurately predict stock market movements has long been a pursuit of researchers, investors, and financial institutions. Accurate predictions can provide valuable insights for making informed investment decisions, managing risk, and optimizing portfolio performance. Traditional approaches to stock market prediction, such as fundamental analysis and technical indicators, have limitations in capturing the complex dynamics of financial markets. In recent years, the emergence of machine learning techniques has shown promise in improving the accuracy and robustness of stock market predictions.

This paper presents a comprehensive comparative study of machine learning techniques for stock market prediction. We explore the performance and effectiveness of various algorithms, including support vector machines, random forests, long short-term memory (LSTM) networks, and gradient boosting models. By conducting a systematic evaluation of these techniques, we aim to shed light on their capabilities, limitations, and potential for advancing stock market prediction.

The study leverages a rich dataset comprising historical stock prices, financial indicators, and potentially alternative data sources, such as sentiment analysis of news articles and social media data. By incorporating a diverse set of features, our models can capture both the intrinsic characteristics of individual stocks and the broader market trends. This comprehensive approach allows us to examine the predictive power of machine learning models under varying market conditions and across different time horizons.

One of the key focal points of our investigation is the LSTM network, a type of recurrent neural network known for its ability to model sequential data. LSTM networks have gained considerable attention in recent years due to their capacity to capture temporal dependencies and nonlinear patterns in time series data. We delve deeper into the performance of LSTM networks and compare them against other machine learning algorithms to assess their effectiveness in stock market prediction.

In this paper, we will introduce a real time prediction system with data provided by VnStock. Our pipeline consit of Kafka as message broker, Spark as data process system and a web UI to display predicted data.

2 Related works

The first step in addressing problems related to the stock market is to gain knowledge about it. Our team has spent an amount of time to dive deeper into the stock market, especially the Vietnamese stock market, where the main topic of this research is about the price of VIC shares.

The next step of this process is all about the related works that we have learned in order to apply our knowledge to manage data streams as well as handle errors during this process using our research on technologies.

- Apache Kafka is one of the most popular open-source distributed messaging platform and is often used for real-time streaming and processing data. Kafka proved its advantages in many fields which need real-time features such as services, production, social networks, finance and banking, etc,.. based on the ability to transmit an enormous number of messages in real-time along with error-control capability. It can also ensure that our information does not get lost. (Apache Software Foundation, 2022)

  The story of Kafka is based on the message sender as we named as The Producer and the message receiver is The Consumer. The related messages (our data) will be categorized into Topics. Kafka Clusters include Kafka Servers or we can call as Brokers, which is a bridge between the Producers and the Consumers in order to transfer messages. These Brokers are managed by the Zookeeper.

- Apache Spark is a famous open-source unified analytics engine for large-scale data processing. This engine supports multi languages with flexible programming model, which helps developers and data scientists to maximize their capabilities. In this project, our team decided to use the PySpark library, a Python interface for Spark to perform Spark SQL queries. We also use Spark Streaming, one of Spark's components to process real-time data. Finally, we will use Spark's MlLib machine learning library to process the data and to learn and make predictions. Our goal is to explore the power of Apache Spark and apply it to this project. (Zaharia et al., 2010)

- Learning models for time-series data : LSTM, ARIMA, Prophet, TCNForecasting (Google Brain Team, 2015)

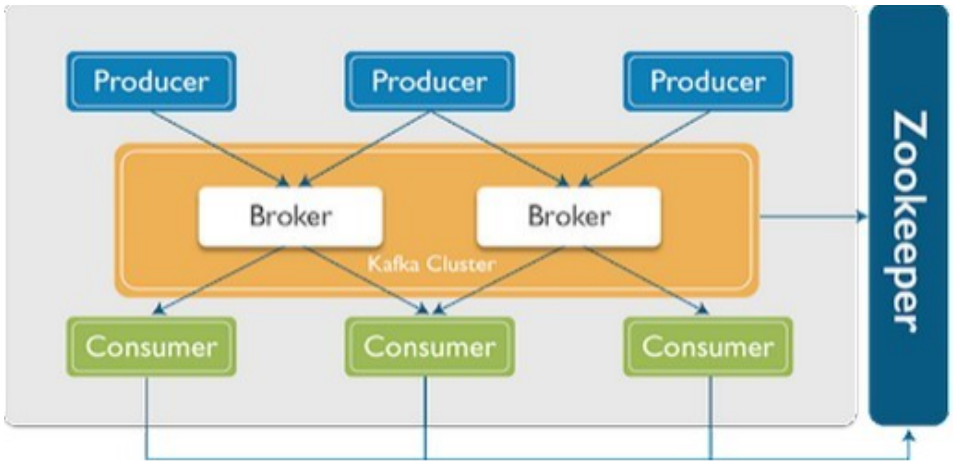

Figure 1: Kafka Architecture

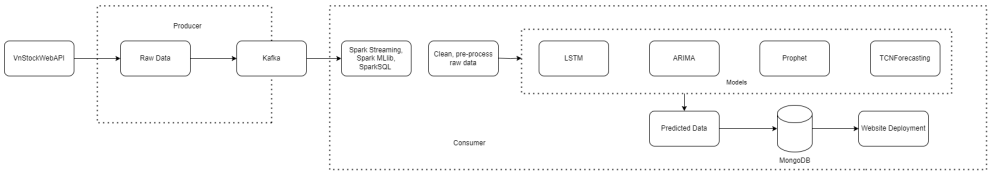

Figure 2: Our System's Architecture

3 Data

3.1 Collecting data

This dataset is collected form VnStockAPI, a Python-based library designed to support researches on the Vietnamese stock market. Because our data came from this library so the information is verified to be accurate and reliable for use. Based on the raw data obtained, our team designed some steps to pre-process the data in order to make it suitable for our learning models.

The dataset was collected over a period from January 1, 2013 to December 31, 2023 with daily information about VIC shares price including the following columns:

- time: the date

- open: opening price of that day
- high: highest price of that day
- low: lowest price of that day
- close: closing price of that day
- volume: the transaction volume of that day
- ticker: the name of the company where the shares belong to

3.2 Data pre-processing

With the shares price, we will focus on the "time" and "close" column to train the models because closing price is the last price of that day so it is more valuable. In this step, our team decided to use the Min Max Scaler from 0 to 1, and our data will be separated into 2 part : 1 for training (from 01/01/2013 to 31/12/2022) and 1 for testing (from 01/01/2923 to 31/12/2023)

	time	open	high	low	close	volume	ticker
0	2013-01-02	16876	17407	16770	16982	106720	VIC
1	2013-01-03	16982	16982	16770	16982	118830	VIC
2	2013-01-04	16876	17088	16770	16982	162880	VIC
3	2013-01-07	17088	17831	17088	17725	370650	VIC
4	2013-01-08	17937	18044	17725	17937	363210	VIC

Figure 3: Enter Caption

4 Experiment

4.1 System Architecture

Our system consists of some main parts, the sender (The Producer), the receiver (The Consumer) and models. The data, after being received form the VnStock API, is transferred from the Producer to the Consumer via KafkaProducer, which serves as a bridge. The data then undergoes preprocessing, cleaning and normalization. After that, our data is fed into the model to train and generate the final results. These results are then used for display on a simple website created by our team in order to represent the real-time analysis.

In more details, the 'Producer' is responsible for collecting and sending data to the 'Consumer'. In this case, the Producer receives data from the VnStock API. The 'Consumer' retrieves this data and processes it. This process includes cleaning (remove irrelevant rows), normalization (using Scalers to scale numerical data to a standard range, in this case is from 0 to 1, and some other preprocessing steps to make the data suitable for our learning models.

The 'Model' as defined before is the machine learning algorithm that we use to train and make predictions based on the processed data. The result from the Model is then displayed on our website, which provides users with valuable insights about VIC shares price.

4.2 Models

4.2.1 LSTM

LSTM model (Long Short-Term Memory) is a type of Recurrent neural network (RNN), which is designed to address the problem named the 'long-term dependency problem' which often encountered in traditional RNNs when processing long data sequences. LSTM has the ability to learn long-term dependencies and handle variable-length sequence data. This model is not only capable of processing single data points for example images but also entire data sequences such as speech or video inputs. LSTM has proven its effectiveness in time series prediction, text classification, machine translation and speech recognition. Main modules of a LSTM :

- Forget Gate : This gate decides which information from the past will be "forgotten" and not be fowarded to the current cell state. This gate helps the model manage information and handle long-term sequences.
- Input gate : This gate decides which information from the current input will be added to the cell state. It helps determine which updates should be made to the cell state.
- Output Gate : This gate decides which information from the cell state will become the model's output. It helps determine the output of the predictive model.
- Cell State: This object stores long-term information and is the main place for processing and retaining information throughout the sequence process.
- Hidden State : This object is the output of the model at each time step and contains short-term information from the cell state.

LSTM model have the ability to store information over a long period of time, which is very useful when process time series or sequential data. An importtant feature of LSTM is its ability to decide what information will be stored and what will be discarded, which provides flexibility in the learning and predicting process.

4.2.2 ARIMA model

The ARIMA model (AutoRegressive Integrated Moving Average) is a method of forecasting and modeling time series in statistics and forecasting. The ARIMA model is designed to handle types of time series that do not follow a simple linear model.

George Box and Gwilym Jenkins (1976) conducted research on the ARIMA model (Autoregressive Integrated Moving Average), and they are often known as the Box-Jenkins method. This model is widely applied in the analysis and forecasting of time series.

This method includes 4 important parts:

- Model Identification: Determine the ARIMA model that fits the data through a process of experimentation and analysis.
- Estimation: Estimate the parameters of the ARIMA model based on observed data.
- Diagnostic Checking: Test and diagnose the accuracy of the model using statistical methods and graphs.
- Forecasting: Use the constructed ARIMA model to forecast future values.

There are many different forecasting methods, but the ARIMA model stands out for its flexibility and high reliability in forecasting, especially in the case of short-term forecasting. Some advantages of ARIMA include the ability to apply to different types of time series without requiring much linear theory.

For ARIMA, the minimum necessary number of observations is 50, and the forecasting environment is better when there is less volatility. ARIMA is often preferred in short-term forecasting, and from ARIMA, forecasting methods such as ARCH and GARCH can be expanded, especially when considering risk factors and market volatility.

4.2.3 Prophet model

Prophet is a robust forecasting tool developed by the Facebook's Core Data Science Team. This is well-known for analyzing time-series data as well as forecasting sales, user growth,... in the business field. Prophet also has the ability to handle missing data, shifts in trend, large outliers, which are common in business's time-series data.

This tool use an addictive regression model with logistic growth curve trends, it includes a yearly seasonality component modeled by a Fourier series and a weekly seasonality component modeled by dummy variables.

Prophet is also very easy to use and does not require experts in statistical knowledge, which makes it accessible to our team. We decided to choose Prophet because it is a reliable and robust choice for predicting a field such as stock prices.

4.2.4 TCN model

The TCN (Temporal Convolutional Network) model is a type of neural network used in time series processing. It is a variant of convolutional neural networks (CNNs) designed specifically to handle time series data that traditional models like recurrent neural networks (RNNs) or Long Short-Term Memory (LSTM) networks may struggle with.

TCN uses 1D convolutional layers to automatically learn hidden features of time series data at both local and global levels. Unlike RNN models, TCN can handle sequences of varying lengths and can capture long-range dependencies without using mechanisms such as gates as in LSTMs.

Advantages of TCN include:

- Ability to learn local and global features: TCN can automatically learn features at various levels of complexity in time series data.
- Parallelism in training: Convolution operations can be computed in parallel, speeding up the training process compared to some RNN models.
- Scalability: TCN can easily scale to handle sequences of varying lengths without changing the network structure.
- Performance: In some cases, TCN has been shown to be more effective than traditional models like LSTMs in predicting and processing time series data.

TCN has been widely applied in various fields such as time series prediction, natural language processing (NLP), speech recognition, and many other applications requiring efficient time series data processing.

5 Results

5.1 Evaluation Methods

To completely analyze the results of our models and predictions, our team will need some benchmark metrics as the standard to evaluate the success of our models as well as to do more research if there are cases of overfitting and underfitting, which reduce the performance of our models

These metrics includes MSE, MAE, RMSE, R^2 and MAPE.

5.1.1 MSE

MSE (Mean Square Error) is a measure that finds the average squared error between the predicted and actual values. Since it is squared, this value is always non-negative and the values are best when they are close to 0.

$$\text{MSE} = \frac{1}{n}\sum(Y_i - \hat{Y}_i)^2$$

Figure 4: MSE formula

5.1.2 MAE

MAE (Mean Absolute Error) is a measure of accuracy between the model's predictions and reality by measuring the average difference between the predicted and actual values. The smaller the MAE, the more accurate the prediction, and vice versa.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n} |\boldsymbol{x}_i - \boldsymbol{x}|$$

Figure 5: MAE formula

5.1.3 RMSE

RMSE (Root Mean Square Error) is a measure of error between the predicted and actual values by calculating the square root of the average squared error between the predicted and actual values. Because it involves a square root, it allows for the assessment of the magnitude of the error. Like MSE, the smaller this value, the better the model, and vice versa.

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

Figure 6: RMSE formula

5.1.4 MAPE

MAPE (Mean Absolute Percentage Error) is a measure used to compute the accuracy of a forecasting model in statistics. It calculates the average absolute percentage difference between the actual and predicted values. The smaller the MAPE, the higher the level of accuracy of the forecast. It's particularly useful because it presents the error in terms of percentage, making it easy to understand and interpret the forecasting accuracy

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{Y_i - \hat{Y}_i}{Y_i} \right|$$

Figure 7: MAPE formula

Metrics	MSE	RMSE	MAE	R^2	MAPE
Prophet	243415924.204	15601.79	13125.06	0.4808	17.59
LSTM	3823363.17	1955.34	1588.16	0.933	3.166
ARIMA	47727679.044	6908.52	5054.67	-0.09	11.2
TCN Forecasting	3047087.09	1745.59	1381.0256	0.9446	2.59

Table 1: Results

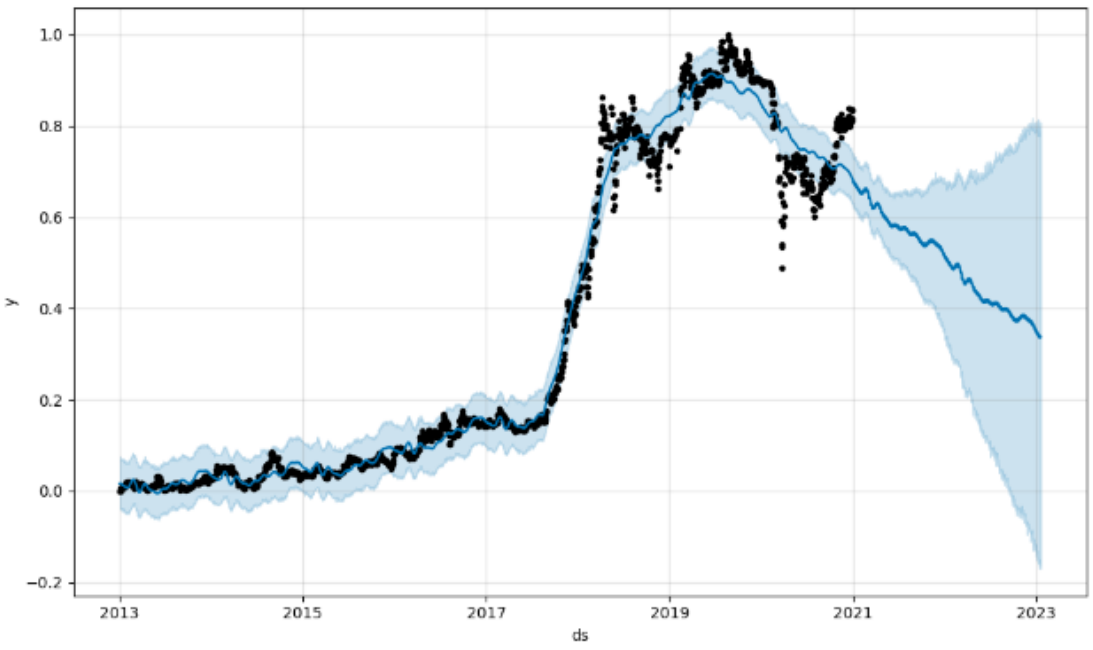

Figure 8: Prophet model's result

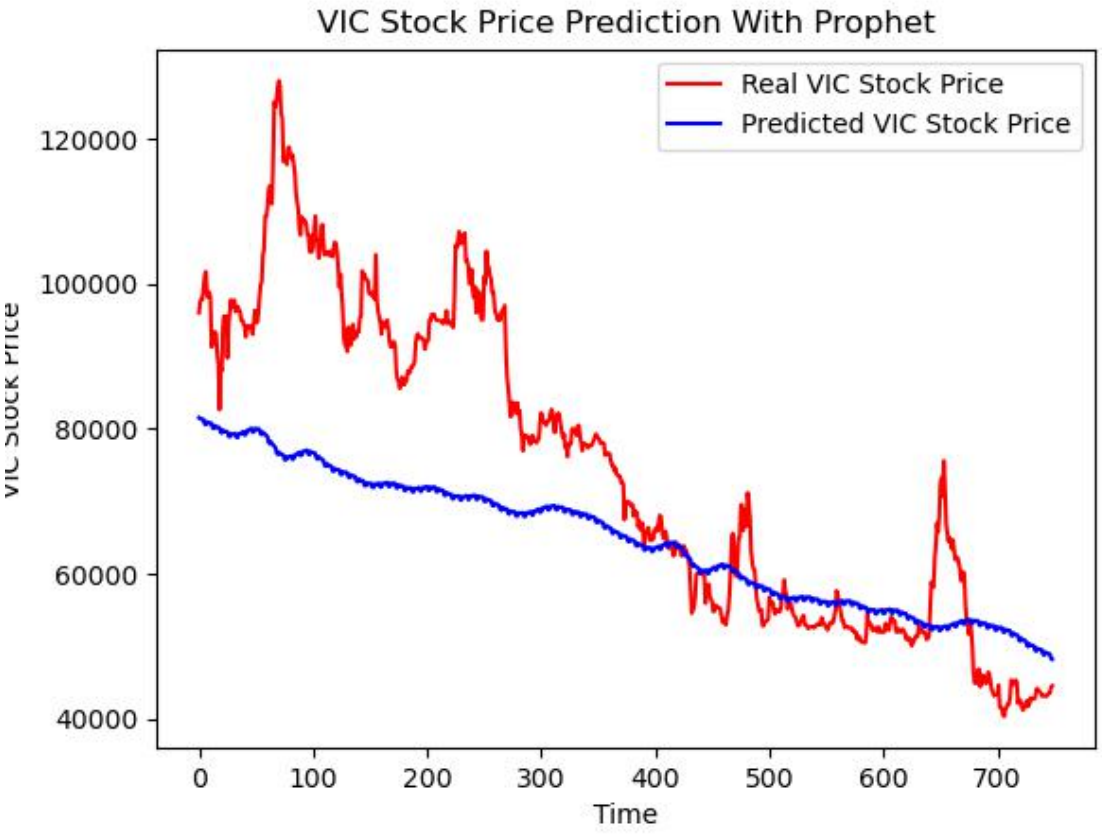

Figure 9: Prophet model's result

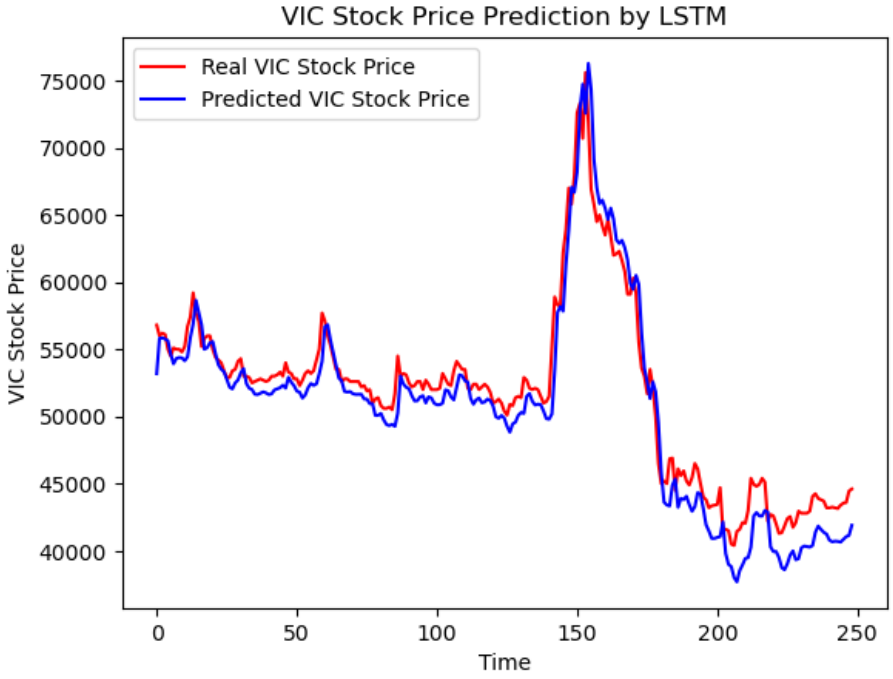

Figure 10: LSTM model's result

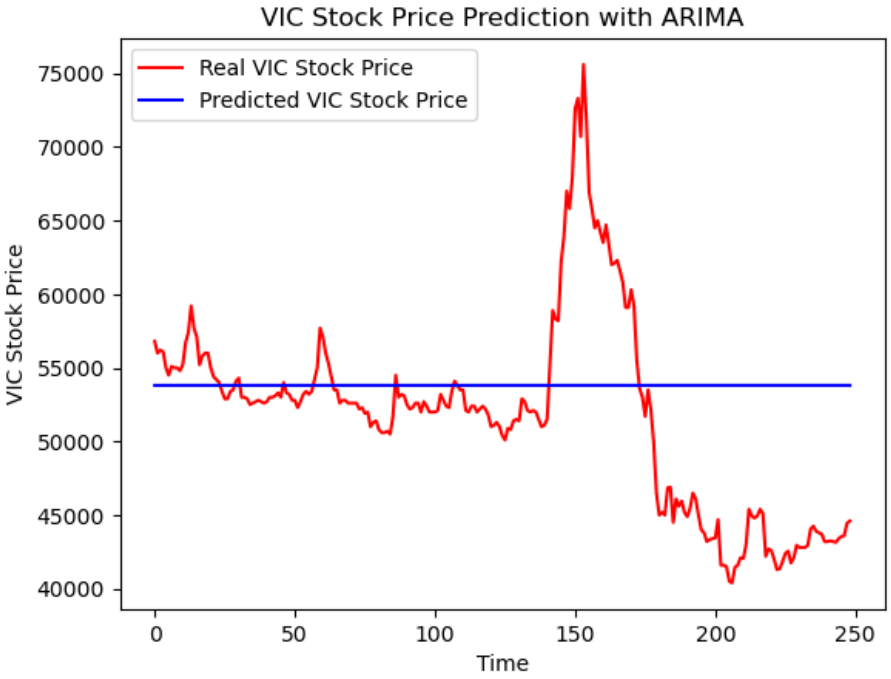

Figure 11: ARIMA model's result

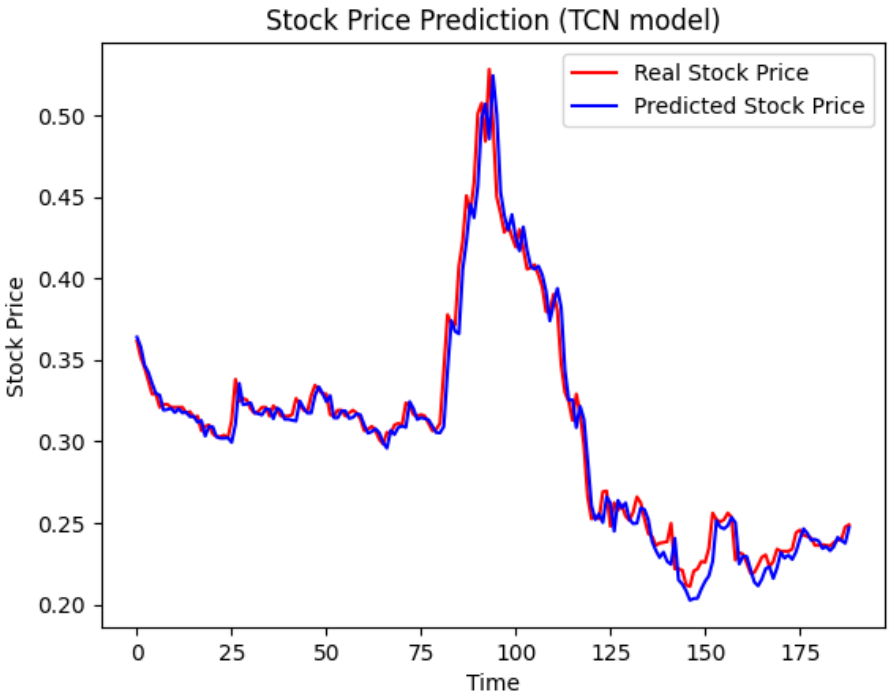

Figure 12: TCNForecasting model's result

5.2 Demo

Out demo include 3 stages: collect data, process data and display data.

In the first stage, we collect data from site likes VnStock ... Then, the second stage is when the data are processed by our model and return the predicted data based on the processed data. Finally, the predicted data are display on the web page with a real time plot that show the data which are updated every 2 seconds.

Figure 13: Our Web User Interface

6 Summary

From our team's research, based on the fluctuation of the of the stock market, analyzing and predicting is one of the most important part to support investment decisions. In this report, our team has made technical analyzes in order to predict trends of the stock market. With that predicted values, we used MongoDB to store these values and build a small website to present real-time analysis, which brings visualization to our research.

Based on learning and using suitable models like LSTM, ARIMA, Prophet, TCNForecasting and support platforms like Apache Kafka, Apache Spark with the knowledge gained from this subject, our team can make some predictions about the stock market in the recent future. Nevertheless, these models may not bring accurate and complete result, which need more research and addition in the future.

While using those models, we discovered that LSTM model and especially TCNForecasting model proved their effectiveness in predicting stock market with excellent score measured for example the LSTM got 3.166 in MAPE score while 2.59 is for the TCNForecasting Model, respectively. Prophet and ARIMA proved their advantages in capturing linear trends and seasonal patterns. However, they may not perform well with complex data like stock prices, where nonlinear factors and long-term dependencies between data points can play a significant role.

On the other hand, LSTM (Long Short-Term Memory) and TCN (Temporal Convolutional Networks) are deep learning models capable of learning nonlinear relationships and long-term dependencies in data. LSTM addresses the vanishing gradient problem that traditional recurrent neural networks face, enabling it to learn long-term dependencies in time series data. This makes LSTM and TCN good choices for stock price prediction. Otherwise, we know that the stock market is an unpredictable environment, where anything happens in this world can be an enormous impact on the stock price, which means the models are not always true and precise. The crucial thing is remain flexible and adaptative to changes in order to update the models to deal with the stock market's fluctuation.

Acknowledgements

Our team expresses sincere gratitude to all the members who have contributed to the successful completion of this project. We also extend our deepest thanks to Dr.Do Trong Hop, who has encouraged us and provided constructive feedback that has improved our understanding and approach to resolving the knowledge needed to create the most complete course project.

Without the collective efforts of the individuals and organizations mentioned above, this project would not have been able to achieve its current level of completion.

References

Apache Software Foundation. 2022. Apache kafka documentation.

Google Brain Team. 2015. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. Software Library.

Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica. 2010. Spark: Cluster computing with working sets. In *Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing (HotCloud)*.