HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY
FACULTY OF COMPUTER ENGINEERING

IT2022E
SEMESTER 2022.2

STOCK PRICE ANALYSIS & PREDICTION

APPLIED STATISTICS AND EXPERIMENTAL DESIGN

CLASS CODE: 141179

OUR GROUP INCLUDES FOUR MEMBERS:

Tạ Ngọc Minh	20214918	minh.tn214918@sis.hust.edu.vn
Lê Ngọc Bình	20214878	binh.ln214878@sis.hust.edu.vn
Trần Thanh Trường	20214938	truong.tt214938@sis.hust.edu.vn
Hoàng Đình Dũng	20214882	dung.hd214882@sis.hust.edu.vn

SUPERVISOR: Associate Professor Nguyễn Linh Giang

Hanoi - 2023

Vietnamese Stock Price Prediction: *Analyzing Market Trends and Forecasting Future Price*

[*]Project Report of the course IT2022E - Applied Statistics and Experimental Design under the supervision of Assoc. Prof. Giang L. NGUYEN

Ta Ngoc Minh[*]
SID No. 20214918
Email: minh.tn214918[†]
HUST-SoICT

Le Ngoc Binh
SID No. 20214878
Email: binh.ln214878[†]
HUST-SoICT

Tran Thanh Truong
SID No. 20214938
Email: truong.tt214938[†]
HUST-SoICT

Hoang Dinh Dung
SID No. 20214882
Email: dung.hd214882[†]
HUST-SoICT

***Abstract*—Stock Price Prediction is the task of forecasting future stock prices based on historical data and various market indicators. It involves using statistical models and machine learning algorithms to analyze financial data and make predictions about the future performance of a stock to help investors make informed investment decisions by providing a forecast of future stock prices. This report is a part of our work in the class 141179, introduce about predicting stock price of Vietnamese market divided into fields of industry by some simple machine learning techniques with three algorithms: ARIMA, XGBooxt and Support Vector Regression.**

***Index Terms*—Statistics, Machine Learning, Stock, Prediction, ARIMA, XGBoost, Support vector regression.**

I. Introduction

THE VIETNAMESE STOCK MARKET has emerged as a vital component of the country's financial landscape, offering investors significant growth potential and opportunities for capital appreciation. With its dynamic economy and increasing participation from both domestic and international investors, accurately predicting stock prices has become a crucial task for market participants, financial institutions, and policymakers.

The ability to predict stock prices can provide valuable insights for investors, enabling them to make informed decisions regarding buying, selling, or holding stocks. Furthermore, accurate predictions can help market regulators identify potential risks and implement appropriate measures to safeguard market stability.

Despite the importance of stock price prediction, the task remains highly challenging due to the complex nature of financial markets, the interplay of various factors, and the presence of inherent uncertainties. However, advancements in computational techniques, machine learning algorithms, and access to vast amounts of historical financial data have opened up new avenues for developing robust prediction models.

This project report aims to explore the field of stock price prediction in the context of the Vietnamese stock market. By employing data analysis, statistical modeling, and machine learning techniques, we seek to forecast the future performance of selected Vietnamese stocks and examine the factors influencing their price movements.

The primary objectives of this project are as follows:

- To analyze historical stock market data from the Vietnamese market, including stock prices, and other relevant financial indicators.
- To identify key market trends, patterns, and dependencies that can aid in understanding the underlying dynamics of stock price movements.
- To develop predictive models using machine learning algorithms and statistical techniques to forecast future stock prices accurately.
- To evaluate the performance of the prediction models based on established evaluation metrics and compare them against traditional forecasting approaches.

By the conclusion of this project, we expect to achieve the following outcomes:

- A comprehensive analysis of historical stock market data, identifying trends and patterns specific to the Vietnamese stock market.
- Development of accurate prediction models capable of forecasting future stock prices for selected Vietnamese stocks.
- Comparative analysis between the performance of the prediction models and traditional forecasting approaches.
- Insights and recommendations for investors, financial institutions, and policymakers to make informed decisions in the Vietnamese stock market.

Through this project, we aim to contribute to the growing body of research in stock price prediction while providing practical implications for stakeholders in the Vietnamese stock market.

II. Data analyzing and feature selection

A. Data crawling

When we searched for public data on Google Dataset Search, we found that the data is mostly for international market, such as NYSE or NASDAQ. However, since stock is the thing that reflect a specific market and also politics,

[†] @sis.hust.edu.vn ; [*]Team leader.

we want to getting data just only for Vietnam, from some Vietnamese markets (HOSE, HNX, UPCOM, etc.)

After analyzing stock dashboard for data crawling, we see that the TCBS Dashboard[1] of Techcombank Securities has its own API for web crawling. We investigated in that API, considered three methods of data crawling using `request`, `Selenium`, and `urllib3` and decided to use `request` based on the data volume and crawling speed. For example, for long-term stored, its API is just like that `...ticker={}...resolution={}&from={}&to={}`, where:

- `ticker`: Stock code
- `resolution`: Step of storage (for example: 1 day, 1 week, etc.)
- `from` and `to`: Starting time and ending time (converted to absolute time)

However, while crawling data, another problem appeared is that the maximum data length of JSON message is only 250 records. So, we decided to use for-loop in order to getting data from the time the company listed on stock market up to July 1st, 2023.

In conclusion, our dataset will contains data up to July 1st, 2023 with 7 attributes: Trading date, Open, High, Low, Close, Volume and Stock Code.

B. Data analysis

Another thing we must consider is that which stock code is chosen, since if we choose a random code, there is no insight will be stated. Many research in [1] and [2] has proven that the effects of macroeconomics and politics divides the stock market into some fields of industry: contruction, technology, consumer (service, F&B, etc.), finance and other industries. Here, for the sake of time and easier analysis, we decide to getting data from 4 mentioned industries with these stock code for the purpose of getting overview of stock market:

- Construction:
  - LCS: Licogi JSC.
  - PTC: ICapital Investment JSC.
- Technology:
  - VGI: Viettel Global Investment JSC.
  - ITD: Tien Phong Technology JSC.
- Consumer:
  - LSS: Lam Son Sugar JSC.
  - PLX: Vietnam National Petroleum Group.
- Finance:
  - TCB: Vietnam Technological and Commercial Joint Stock Bank.
  - VIG: Vietnam Financial Investment Securities Corporation.

Here, I will focus on showing the reason why I said the stock market is affected by macroeconomics and politics, along with some overall trends in economics. The figure 1 illustrates the price of 4 companies during 17 recent years of being listed on stock market, and I will take this is the cases as a representative for all of the market.

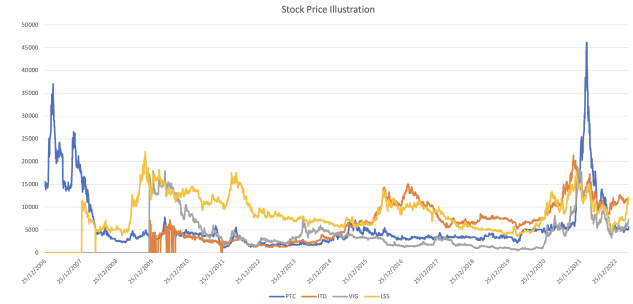

Figure 1. Stock price of PTC, ITD, LSS and VIG from the first listed date up to now

As we can see, the stock price of the period 2008 - 2012 has dropped down. This is due to international economics crisis, also affected on Vietnam. And Vietnamese government had applied some method on national economics, hence, the price had a minor increase in 2010, but it continued to fall down the day since the applied method is not applicable in a long term. [3]

Also, after COVID-19 pandemic, Vietnamese government has applied some policies to promote economic growth, hence, the stock price has witnessed a significant increase [4]. However, after the growth has reached its peak, in 2023 and 2024, Vietnam economics will suffer a dramatically decrease, according to some economics expert.

C. Feature selection

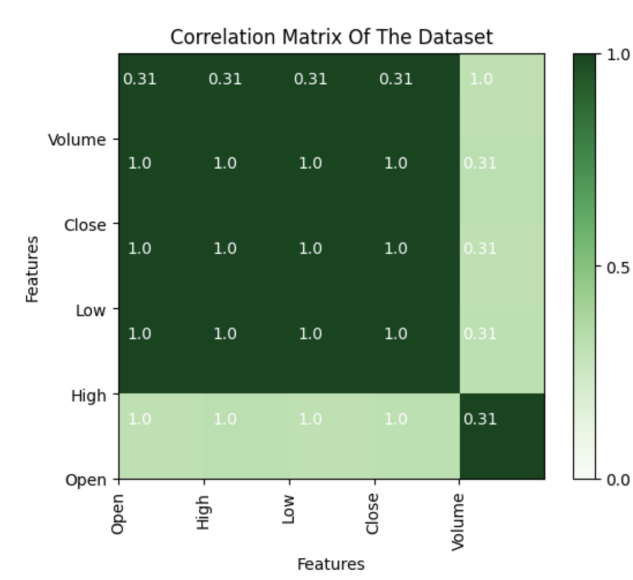

Figure 2. Correlation between features in Stock Price datasets

[1] https://www.tcbs.com.vn/dashboard

In order to select the appropriate feature for the sake of time and accuracy, we analyze the datasets and draw the heat map of correlations between the features.

As we can see, the correlation of Open, High, Low, Close is significantly higher than Volume. Moreover, the score for Open, High, Low and Close are the same, and statistically, they all are the price of the stock during a day, so we decide to use only one of them, the `Close` feature.

However, we will continue use `Volume` in the another part of our project, that is to predict transacted volume by RSI score (relative strength index score) [5].

D. RSI score and the relation to the volume

As I mentioned above, we will use the relative strength index to predict the volume. To start, we will consider about what is RSI, and how it related to the stock.

To calculate RSI, we define upward (U) and downward (D) indicators, which are:

$$U_t = \begin{cases} P_t - P_{t-1} & \text{if } P_t > P_{t-1}, \\ 0 & \text{otherwise.} \end{cases}$$

and

$$D_t = \begin{cases} -P_t + P_{t-1} & \text{if } P_t < P_{t-1}, \\ 0 & \text{otherwise.} \end{cases}$$

where t is the calculated day, and P_t is the stock price for that day.

Then up_t and down_t are average numbers of upward moves and downward moves of closing price of past n days:

$$\text{up}_t = \frac{s}{n+1} \times U_t + \left(1 - \frac{s}{n+1}\right) \times \text{up}_{t-1}$$

and

$$\text{down}_t = \frac{s}{n+1} \times D_t + \left(1 - \frac{s}{n+1}\right) \times \text{down}_{t-1}$$

where s is the smooth parameter, normally, $s = 2$.

Finally, we get the RSI score by using the formula:

$$\text{RSI}_t = 100 - \frac{100}{1 + \frac{\text{up}_t}{\text{down}_t}}$$

In calculation, we usually consider directly:

$$\text{RSI}_t = 100 \times \frac{\text{up}_t}{\text{up}_t + \text{down}_t}$$

The RSI tends to remain more static during uptrends than it does during downtrends. This makes sense because the RSI measures gains versus losses. In an uptrend, there are more gains, keeping the RSI at higher levels. In a downtrend, on the other hand, the RSI tends to stay at lower levels. During an uptrend, the RSI tends to stay above 30 and should frequently hit 70. During a downtrend, it is rare to see the RSI exceed 70, and the indicator frequently hits 30 or drops under this threshold. These guidelines can help determine trend strength and spot potential reversals. The reverse is true for a downtrend. This means that if the downtrend is unable to reach 30 or below and then rallies above 70, that downtrend is said to weaken.

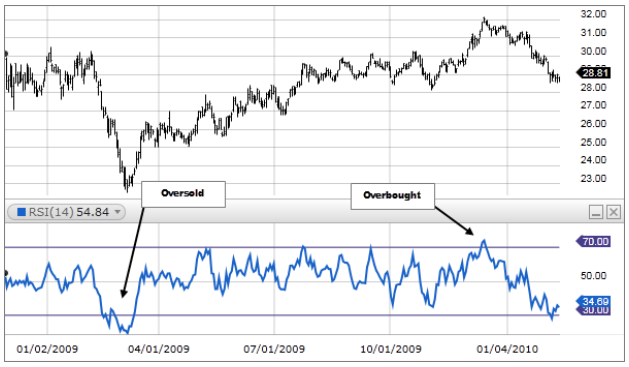

Figure 3. Example of RSI score in corresponding to buy/sell trends

Hence, we can use the RSI for volume prediction in order to get the insight of "How many stocks are exchanged over time and the what is current trend of the stock code?" From that, investor can decide the volume of their own transaction to maximize their benefits.

III. Algorithm Selection and Implementation

A. Extreme Gradient Boosting (XGBoost)

Ensemble Learning is one of the Machine Learning methods that combines some base models to give a optimal model. Instead of making a model and deploying this model with expecting to bring the predictor accurately, Ensemble methods take a lot of models into account and compute average those models in order to produce the final results.

Boosting is one of the three main methods in Ensemble Learning. Different from bagging that training many decision trees parallelly on the different samples of the same dataset and produce prediction by averaging, boosting method combines a set of weak learners into strong learners to minimize training error. In boosting, a random sample of data is selected, fitted with a model and then train sequentially - that is, each model tries to compensate for the weakness of its predecessor.

Extreme Gradient Boosting Machine (XGBM) is the latest version of boosting machines . In XGBM, trees are added sequentially (one at a time) that learn from the errors of previous trees and improve them.

Figure 4 illustrates how gradient tree boosting works.

B. Support Vector Regression (SVR)

Support Vector Machine (SVM) is a machine learning algorithm for classification problems. After the invention of SVM, it was updated to support vector regression (SVR) by incorporating a form of loss function called the ε-insensitive loss function, which penalizes data points as long as they exceed ε, SVR in a high-dimensional feature space becomes a non-linear kernel-based regression method that determines the optimal regression hyperplane with the minimum structural risk. [6]

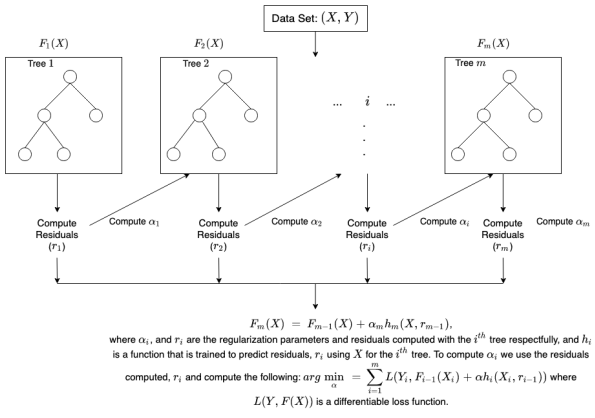

Figure 4. Illustration of XGBoost

The function of SVR method is:

$$y = f(x) = w^T \varphi(x) + b$$

where φ denotes the non-linear mapping from the input space to the feature space, w is a vector of weight coefficients and b is a bias constant. The w and b are estimated by minimizing the following optimization problem:

$$\min \frac{1}{2}\|w\|^2$$

subjects to:

$$\begin{cases} y_i - w^T \varphi(x_i) - b \leq \varepsilon, \\ b + w^T \varphi(x_i) - y_i \leq \varepsilon + \xi_i^* \end{cases}$$

Here, ξ_i and ξ_i^* are slack variables introduced to cope with training data possibly violating the condition $|f(x_i) - y_i| \leq \varepsilon$

$$\min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*)$$

subjects to:

$$\begin{cases} y_i - w^T \varphi(x_i) - b \leq \varepsilon, \\ b + w^T \varphi(x_i) - y_i \leq \varepsilon + \xi_i^*, \\ \xi_i, \xi_i^* \geq 0, i = 1, ..., n, \end{cases}$$

where C is a constant known as the penalty factor, ε is the insensitive loss parameter and the slack variables ξ_i and ξ_i^* measure the amount of difference between the estimated value and the target value beyond ε.

C. Auto Regressive Integrated Moving Average (ARIMA)

ARIMA is actually a class of models that explains a given time series based on its own past values, that is, its own lags and the lagged forecast errors, so that equation can be used to forecast future values. [7]

Any 'non-seasonal' time series that exhibits patterns and is not a random white noise can be modeled with ARIMA models.

An ARIMA model is characterized by 3 terms: p, d, q, where,

- p is the order of the AR term,,
- q is the order of the MA term
- d is the number of differencing required to make the time series stationary.

The first step to build an ARIMA model is to make the time series stationary. Because, term 'Auto Regressive' in ARIMA means it is a linear regression model that uses its own lags as predictors. Linear regression models, as you know, work best when the predictors are not correlated and are independent of each other. The most common approach is to difference it. That is, subtract the previous value from the current value. Sometimes, depending on the complexity of the series, more than one differencing may be needed.

The value of d, therefore, is the minimum number of differencing needed to make the series stationary. And if the time series is already stationary, then $d = 0$. Next, what are the p and q terms?

p is the order of the 'Auto Regressive' (AR) term. It refers to the number of lags of Y to be used as predictors. And q is the order of the 'Moving Average' (MA) term. It refers to the number of lagged forecast errors that should go into the ARIMA Model.

A pure Auto Regressive (AR only) model is one where Y_t depends only on its own lags. That is, Y_t is a function of the 'lags of Y_t'.

$$Y_t = \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + ... + \beta_p Y_{t-p} + \varepsilon_1$$

where, Y_{t-1} is the lag1 of the series, beta1 is the coefficient of lag1 that the model estimates and α is the intercept term, also estimated by the model.

A pure Moving Average (MA only) model is one where Y_t depends only on the lagged forecast errors.

$$Y_t = \alpha + \varepsilon_t + \varphi_1 \varepsilon_{t-1} + \varphi_2 \varepsilon_{t-2} + ... + \varphi_q \varepsilon_{t-q}$$

where the error terms are the errors of the autoregressive models of the respective lags. The errors E_t and E_{t-1} are the errors from the following equations :

$$Y_t = \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + ... + \beta_t Y_0 + \varepsilon_t$$

$$Y_{t-1} = \beta_1 Y_{t-2} + \beta_2 Y_{t-3} + ... + \beta_{t-1} Y_0 + \varepsilon_{t-1}$$

An ARIMA model is one where the time series was differenced at least once to make it stationary and you combine the AR and the MA terms. So the equation becomes:

$$\begin{aligned} Y_t = \quad & \alpha + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + ... + \beta_p Y_{t-p} + \varepsilon_t \\ & + \varphi_1 \varepsilon_{t-1} + \varphi_2 \varepsilon_{t-2} + ... + \varphi_q \varepsilon_{t-q} \end{aligned}$$

IV. Model tuning

A. Hyperparameter Tuning

Hyperparameter tuning is primarly finding the set of parameters' values such that this contributes to the performance of the model, maximizes the efficient but still ensures the complexity within a dataset. This process is used to not only improve the predictive power of the model but also improve the speed, make the model runs faster.

1) Choosing hyperparameters: It is probably the hardest problem in tuning hyperparameter. The reason is that it is difficult to choose the most important one to tune and the hyperparameters have interaction and affect each other within the process. Therefore, the evaluation at one time doesn't work since when changing to another hyperparameter, the evalutaion will be completely different. In this model, intuitively and by experiment, we just select the hyperparameter that likely has the most significant impact on the model for tuning test.

Support Vector Regression:

- **C**: In SVR (Support Vector Regression), the regularization parameter, denoted as C, controls the trade-off between the fitting error and the complexity of the model. It determines the penalty for data points that fall outside the margin or violate the regression tolerance. A higher value of C in SVR implies a smaller tolerance for errors, making the model try to fit the training data more closely. This can lead to overfitting, where the model becomes too specific to the training data and performs poorly on unseen data. On the other hand, a lower value of C allows for a wider margin and allows more errors, resulting in a more flexible model that can better generalize to unseen data. In the provided code, the values of C are $[0.01, 0.1, 1, 100]$, and the SVR model will be trained and tested using each of these values to find the optimal value of C for the given regression task.
- **kernel**: This parameter determines the type of kernel function used in SVR to transform the feature space. There are three commonly used kernels:
  - **linear**: Linear kernel, used when the data can be well separated by a hyperplane.
  - **rbf**: Gaussian kernel (RBF - Radial Basis Function), the most commonly used kernel, used when the data is not linearly separable in the original space.
  - **sigmoid**: Sigmoid kernel, used when the data is not linearly separable and has similar characteristics to the sigmoid function in logistic regression. The model will be trained and tested with each of these kernel types.
- γ: This parameter adjusts the influence of a training data point on other data points. A higher gamma value limits the influence to nearby points only, while a lower gamma value spreads the influence to both nearby and distant points. In the given code, the values of gamma are $[0.01, 0.001, 1]$, and the model will be trained and tested with each of these values.

These parameters allow adjusting the SVR model to find optimal separating hyperplanes in the feature space for better performance.

XGBoost:

- **n_estimators**: This parameter represents the number of individual decision trees (weak learners) to be built in the XGboost model. Increasing the number of estimators can improve the model's performance by reducing bias, but it also increases computation time. In the provided code, the values of n_estimators are $[100, 500, 1000]$, and the model will be trained and tested with each of these values.
- **max_depth**: This parameter defines the maximum depth of each decision tree in the GBM model. A higher max_depth allows the model to learn more complex relationships in the data but may lead to overfitting. On the other hand, a lower max_depth restricts the tree's complexity, reducing overfitting but potentially sacrificing some predictive power. In the provided code, the values of max_depth are $[3, 6, 10]$, and the model will be trained and tested with each of these values.
- **learning_rate**: This parameter controls the contribution of each tree in the ensemble. A smaller learning rate makes the model more conservative by reducing the impact of each tree, leading to slower learning but potentially better generalization. Conversely, a higher learning rate allows the model to learn faster but may also result in overfitting. In the given code, the values of learning_rate are $[0.01, 0.05, 0.1]$, and the model will be trained and tested with each of these values.

B. Cross Validation For Time Series

When constructing a model, it is important to assess its performance. *Cross-validation* is a statistical technique that can assist in this process. One commonly used approach is K-fold Cross-Validation, where the dataset is divided into several subsets or folds. The model is trained on all but one fold and then tested on the remaining fold. This process is repeated for each fold, allowing for a comprehensive evaluation of the model's performance.

However, We can't use this process in Time Series. Cross-validation becomes more challenging in the context of time series data. Randomly assigning samples to the test and train sets is not appropriate because it would involve using future values to predict past values, which is illogical. In another word, future-looking need to be avoided while training model.

Time Series Cross Validation: The solution is an unique way of Cross-validation for time series. Cross-validation on a rolling basis is a technique that can be applied to cross-validate the time-series model. Start with a small subset of data for training purposes, make predictions for subsequent data points, and then assess the precision of the predictions. The following training dataset has the same predicted data points, and additional data points are forecasted after that. These splits must follow the rule:

- Every test set contains unique observations
- Observations from the training set occur before their corresponding test set

In this project, we introduce two kind of **Nested Cross Validation**. We will address the case of stock with data spanning numerous days:

- Predict Second Half
- Day Forward Training

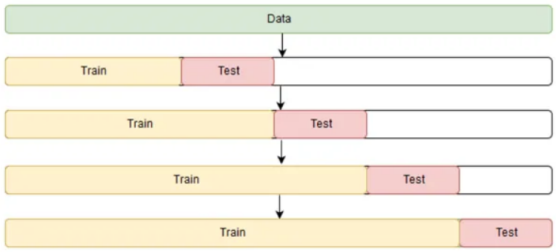

Figure 5. Cross Validation for Time Series

1) ***Predict Second Half****:* The first approach is called Predict Second Half, serves as the fundamental nested cross-validation (CV) method. The "Predict Second Half" method involves a single train/test split, where the first half of the data is used for training, and the second half is used for testing.

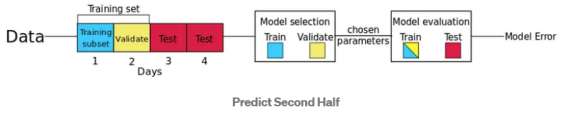

Figure 6. Illustration of Predict Second Half

In this approach, the first half of the data is allocated to the training set, and the latter half is designated as the test set, with the validation set's size varying depending on the specific problem being addressed. It is crucial to maintain chronological order, ensuring that the validation set consists of data subsequent to the training subset. Figure 6 illustrates how Predict Second Half works.

The advantage of this approach is its simplicity in implementation. However, it has a limitation in that the test set is chosen arbitrarily, which may introduce bias or lead to unreliable results. To address this, it is important to maintain the chronological order of the data, ensuring that the validation set, used for model selection or hyperparameter tuning, consists of data that comes after the training subset. This helps in evaluating the model's generalization performance on unseen future data.

2) ***Day Forward-Chaining****:* The Predict Second Half nested cross-validation method above has a limitation in that the arbitrary selection of the hold-out test set can lead to biased estimates of prediction error when evaluating the model on an independent test set. To overcome this drawback and obtain a more reliable estimate of model prediction error, a common approach is to create multiple train/test splits and average the errors across all splits. One effective technique for achieving this is known as *Day Forward-Chaining*.

Day Forward-Chaining is based on the principles of forward-chaining and rolling-origin-recalibration evaluation. This method involves systematically considering each day as the test set while assigning all previous data to the training set. For instance, if we have a dataset spanning five days, Day Forward-Chaining would generate three different training and test splits, ensuring that there is at least one day of training and validation data available. The figure 7 illustrates this process.

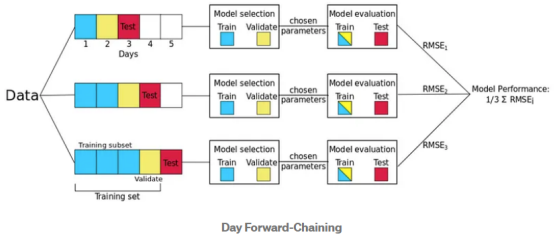

Figure 7. Illustration of DayForwardChaining

By employing this approach, we generate multiple train/test splits, allowing us to assess the model's performance across different periods of time. Each split represents a distinct evaluation scenario, where the model is trained on historical data and tested on subsequent days. This diversity of splits helps to capture the model's ability to generalize and make accurate predictions across different time periods.

To compute a robust estimate of the model error, the prediction error on each split is averaged. This averaging process smooths out any fluctuations or variations that may occur due to specific train/test splits and provides a more stable evaluation of the model's performance.

3) ***Comparison and Conclusion****:* The Predict Second Half approach relies on a single train/test split, which may not capture the full variability of the data or provide a robust estimate of model error.The Predict Second Half approach may not effectively evaluate the model's ability to generalize to unseen future data, as the test set is chosen arbitrarily.

Whereas, day-forward training generates multiple train/test splits by considering each day as a separate test set. By averaging the model's performance across these splits, a more robust estimate of the model's error can be obtained, accounting for variations and fluctuations across different time periods. These aligns with the objective of forecasting future values in time series analysis. By training the model on past data and evaluating it on subsequent days, the model is tested on unseen future data points, allowing for a better assessment of its generalization capabilities.

In summary, Day-forward training offers advantages over the Predict Second Half approach in terms of test set selection, accounting for temporal dynamics, evaluation robustness, and generalization to future unseen data. By systematically considering each day as a test set, Day-forward training provides a more representative evaluation of the model's performance

and better captures the temporal dependencies present in time series data.

V. Implementation

All the sources code and implementation are here: Repository.

Dataset is attached in the folder "Dataset", that includes some stocks in diffrent fields (construction, finance, consumer, technology).

We analyze some indicators like RSI, buy or sell signal in folder "Data Analysis". You can access this to see implementation, some properties of dataset and our prediction about buy or sell indicated RSI.

About machine learning models, we use three algorithms that are ARIMA, Support Vector Regression (SVR), Extreme Gradient Boosting (XGBoost) to train, deploy and evaluate model, then give predictions about price of stock.

Before using algorithms to train model, we created data preprocessing, cleaning, and visualization for easier implementation.

In ARIMA model, we apply some test statistics to check if data is stationary or not, and then choose the parameters to train and evaluate model.

In folder SVR and XGBoost, we try to model selection and choose optimal hyperparameters by using Nested Cross Validation Time Series (Predict-Second-Half and Day-Forward-Chaining). After that, we use that hyperparameters to retrain, evaluate model and predict price of stocks.

VI. Summaries and Improvements

A. Results and Summaries

After running several times, we get the accuracy of the chosen datasets is represented in the table below. The number in each cells are train mean squared error, train R^2 score, test mean squared error, and test R^2 score respectively.

Alg.	LCS	PTC	VGI	ITD
ARIMA	0.06	0.0015	0.04	0.007
	0.94	0.998	0.96	0.94
	0.12	0.0034	0.09	0.008
	0.90	0.801	0.9	0.93
SVR	0.005	0.004	0.016	0.002
	0.995	0.995	0.983	0.997
	0.015	0.007	0.042	0.0224
	0.984	0.992	0.957	0.977
SVR & DFC	0.006	0.0044	0.014	0.0014
	0.994	0.995	0.984	0.995
	0.001	0.0043	0.018	0.006
	0.972	0.936	0.960	0.963
SVR & PSH	0.006	0.004	0.0016	0.002
	0.993	0.995	0.983	0.997
	0.020	0.005	0.019	0.008
	0.979	0.994	0.980	0.991
XGB	0.0003	0.0003	0.013	0.0002
	0.999	0.999	0.998	0.999
	0.033	0.021	0.024	0.202
	0.966	0.978	0.975	0.801
XGB & DFC	0.0015	0.0003	0.0012	0.0004
	0.998	0.999	0.998	0.998
	0.027	0.021	0.0121	0.301
	0.519	0.978	0.875	0.201
XGB & PSH	1.630	4.612	1.569	7.28
	0.999	0.999	0.999	0.999
	0.006	0.016	0.014	0.149
	0.939	0.983	0.985	0.850

Alg.	LSS	PLX	TCB	VIG
ARIMA	0.0053	0.003	0.006	0.008
	0.9947	0.996	0.992	0.93
	0.0087	0.004	0.032	0.163
	0.982	0.945	0.984	0.87
SVR	0.0049	0.0047	0.0029	0.0026
	0.995	0.995	0.997	0.9974
	0.01	0.012	0.021	0.0089
	0.99	0.988	0.978	0.991
SVR & DFC	0.006	0.0044	0.006	0.0026
	0.994	0.996	0.994	0.996
	0.005	0.006	0.005	0.004
	0.99	0.99	0.991	0.988
SVR & PSH	0.0052	0.005	0.0206	0.003
	0.9947	0.995	0.979	0.996
	0.0077	0.008	0.01	0.004
	0.992	0.992	0.99	0.995
XGB	0.13	0.12	0.08	0.06
	0.86	0.981	0.97	0.95
	0.19	0.07	0.06	0.07
	0.8	0.93	0.94	0.96
XGB & DFC	0.12	0.1	0.002	0.09
	0.88	0.9	0.98	0.91
	0.16	0.042	0.038	0.12
	0.83	0.96	0.91	0.87
XGB & PSH	0.01	0.17	0.02	0.12
	0.99	0.82	0.91	0.87
	0.072	0.21	0.03	0.18
	0.93	0.8	0.93	0.821

In conclusion, the model has an acceptable performance in the price prediction. From that, we can have a clear insight about the towards trends of stock market, given there is no sudden changes in policies (national bankrupt, stock take-over, etc.)

B. Proposal of Improvements

Enhancing Data Integration: To improve the overall performance of the system, we recommend integrating additional relevant data sources. By incorporating financial news, social media sentiment analysis, and economic indicators, we can capture a broader range of factors that influence stock prices. This enriched dataset will enable more comprehensive analysis and prediction, resulting in enhanced decision-making capabilities for users.

User-Friendly Interface and Visualization: To ensure an intuitive and user-friendly experience, we propose developing a web-based interface that allows users to interact with the system effortlessly. The interface should provide clear visualizations of predicted stock prices, historical data, and relevant performance metrics. Additionally, incorporating features like customizable alerts and notifications will empower users to make timely and informed investment decisions.

ACKNOWLEDGMENTS

This work is supported and supervised by professor Giang L. NGUYEN under the course of Applied Statistics and Experimental Design. Also, our research and project is partially supported by other professors and students at School of Information and Communication Technology, Hanoi University of Science and Technology.

DISCLAIMERS

Our project is done just for knowledge learning and researching. It is not in-depth investigated enough to make it applicable. We are not responsible for any risks when you invest your money on what our system recommended.

Also, be careful while exchange stocks, otherwise, you may have chance to state the sentence:

"Mắt anh sáng, dáng anh hiền.
Nụ cười toả nắng, mang tiền em đi."

REFERENCES

[1] Phan, K. C., & Zhou, J. (2014). Market efficiency in emerging stock markets: A case study of the Vietnamese stock market. IOSR Journal of Business and Management, 16(4), 61-73.

[2] Ngọc, Đ. B., & Cường, N. C. (2016). Các nhân tố tác động đến dao động giá cổ phiếu của các công ty niêm yết trên thị trường chứng khoán Việt Nam. Tạp Chí Phát Triển Kinh Tế, Tạp Chí Khoa Học Kinh Tế, Tạp Chí Kinh Tế Và Tài Chính Quốc Tế, 228(6), 43-51.

[3] Nguyễn, T. T. V. (2010). Cuộc khủng hoảng kinh tế thế giới năm 2008-2009 và tác động đối với kinh tế-xã hội Việt Nam: Luận văn ThS. Quan hệ quốc tế: 60 31 40 (Doctoral dissertation, Trường Đại học Khoa học Xã hội và Nhân văn).

[4] Chương, P. H. (2020). Tác động của đại dịch Covid-19 đến nền kinh tế việt nam. Tạp chí Kinh tế và Phát triển, 274, 12.

[5] Gumparthi, S. (2017). Relative strength index for developing effective trading strategies in constructing optimal portfolio. International Journal of Applied Engineering Research, 12(19), 8926-8936.

[6] Ghosh, M., & Gor, R. (2022). STOCK PRICE PREDICTION USING SUPPORT VECTOR REGRESSION AND K-NEAREST NEIGHBORS: A COMPARISON. International Journal of Engineering Science Technologies, 6(4), 1–9. https://doi.org/10.29121/ijoest.v6.i4.2022.354

[7] Prabhakaran, S. (2023). ARIMA Model – Complete Guide to Time Series Forecasting in Python. Machine Learning Plus. https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/

[8] Shrivastava, S. (2023, May 24). Cross Validation in Time Series. Medium. https://medium.com/@soumyachess1496/cross-validation-in-time-series-566ae4981ce4