ANALYSIS AND FORECAST OF NVIDIA STOCK PRICES

Yuchen Gu
57126260
yuchengu4-c@my.cityu.edu.hk

Yulin Cai
57126990
yulincai2-c@my.cityu.edu.hk

Yiyao Jin
57124923
yiyaojin2-c@my.cityu.edu.hk

Yang Li
56843298
yli669-c@my.cityu.edu.hk

March, 2024

ABSTRACT

This study provides a comprehensive analysis and forecast of NVIDIA Corporation's stock prices, utilizing historical data from March 2019 to March 2024. We applied advanced forecasting models—XGBoost, ARIMA, LSTM, and VAR—to identify the most effective method for predicting future stock prices. Our findings indicate that the XGBoost model surpasses others in terms of accuracy, due to its robust handling of non-linear data relationships and sophisticated feature engineering. This research highlights the potential of using advanced models to enhance the accuracy of stock price predictions, offering valuable insights for future financial market analysis and forecasting.

1 Motivation and Background

Amid rapid technological progress, NVIDIA stands out for its innovations and leadership in artificial intelligence and data centers. Its FY2024 Q3 and Q4 reports show impressive revenue and net profit surges, with data center business growth at 279% and 409% year-over-year, respectively[1]. These achievements underscore NVIDIA's strengths in tech innovation, AI computing, and data processing, and its wide-ranging product applications. Our analysis of NVIDIA is driven by its stellar financial results and its critical influence on the tech sector's future, providing a valuable perspective for understanding market trends and forecasting future directions.

2 Objectives

The primary objective of our project is to perform a comprehensive analysis of NVIDIA Corporation's stock returns and forecast its prices. Our analysis aims to identify the most accurate model for predicting NVIDIA's stock adjusted close prices, enhancing our understanding of financial data analytics and guiding investment decision-making processes related to NVIDIA Corporation.

3 Methods

Our methodology starts with a preliminary analysis using histograms, time series plots, QQ plots, and box plots to visualize the distribution, trends, deviations, and outliers in stock returns. Following this, we employ four advanced forecasting models to predict NVIDIA's stock adjusted close prices: XGBoost(eXtreme Gradient Boosting), ARIMA(Autoregressive Integrated Moving Average), LSTM (Long Short-Term Memory networks), and VAR (Vector Auto Regression). These models are selected for their proven effectiveness in handling financial time series data and for facilitating a robust comparative analysis. To evaluate their forecasting accuracy, we use two key performance metrics: Root Mean Square Error (RMSE) and Mean Absolute Percentage Error (MAPE). RMSE measures the average size of forecasting errors across all predictions, while MAPE, as a percentage, provides an intuitive measure of the model's performance in comparison to actual observed values.

4 Data Collection and Data Prepossessing

4.1 Data Collection

The historical data used in this report was retrieved from Yahoo Finance, covering the period from March 18, 2019 to March 18, 2024. The dataset includes opening and closing prices, highest and lowest prices, adjacent closing prices, and stock trading volumes, across a total of 1509 effective trading days.

4.2 Data Prepossessing

To calculate daily returns, we applied the formula $R_n = \frac{C_{n+1}-C_n}{C_n}$, where R_n represents the return on day n, and C_n denotes the closing price on day n. In our XGBoost and LSTM model, we firstly split the whole data set into a 70% training set, 25% validation set, and a 5% test set. We used the training set to train our models, then used the validation set to tune the hyper parameters for lower RMSE and MAPE. After selecting the best hyper parameters, we fed the training set and validation set into the model, and finally used the test set to test our predictions and compare the results. In our ARIMA and VAR model, we split the data set into a 90% training set and a 10% test set, for there are not so many complex hyper parameters as XGBoost and LSTM to tune.

After splitting the data set, we scaled the "Adj Close" column in each set respectively to make the observations in each set follow a standard normal distribution (mean=0 and standard deviation=1).

5 Exploratory analysis for return

5.1 Histogram

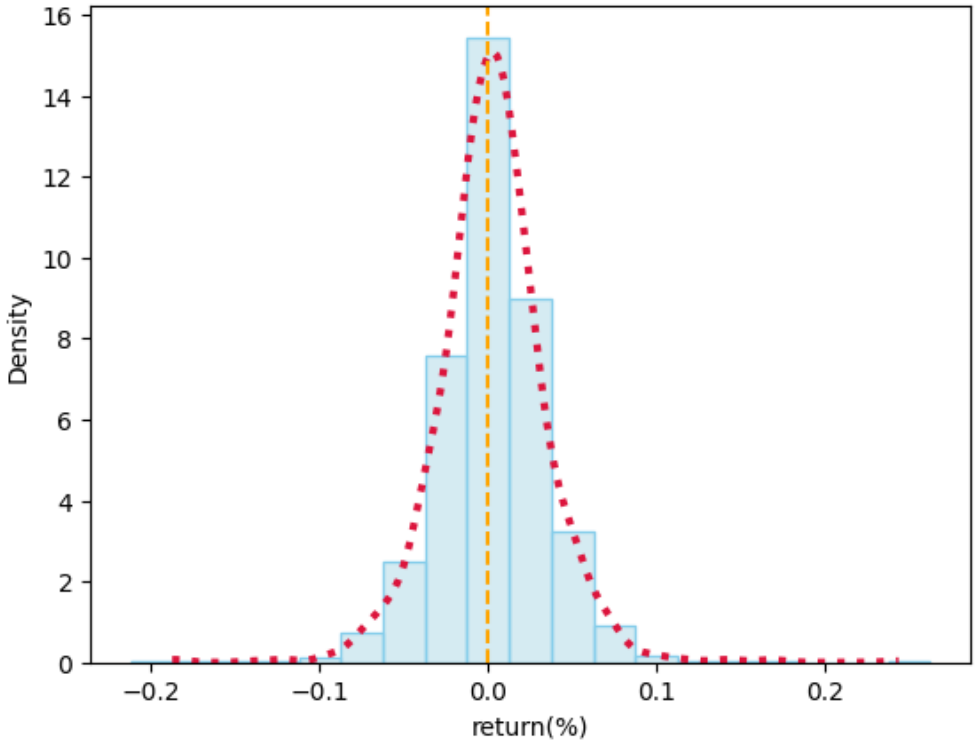

Figure 1: Time Series Plot of Daily Return

Figure 1 shows the density distribution of daily returns from 2018 March 20th to 2024 March 15th. The red line is the kernel density estimation. The return data fit well to a normal distribution with mean equals to 0, consistent with the assumption of CAPM that returns on a portfolio are normally distributed. Most returns are between -0.1 and 0.1.

5.2 Time Series Plot

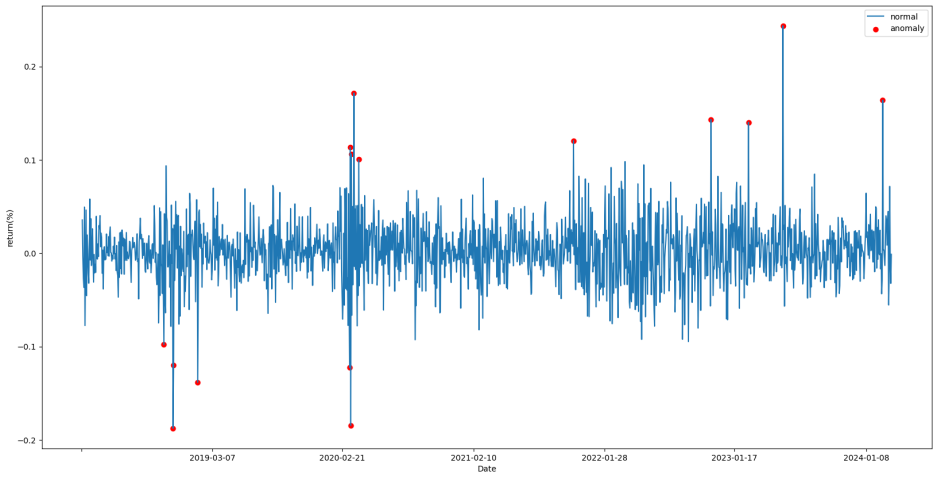

Figure 2: Histogram of Daily Return's Distribution

	Date	return
153	2018-10-24	-0.097937
170	2018-11-16	-0.187559
171	2018-11-19	-0.119990
216	2019-01-28	-0.138245
499	2020-03-12	-0.122368
500	2020-03-13	0.113402
501	2020-03-16	-0.184521
502	2020-03-17	0.106263
507	2020-03-24	0.171564
516	2020-04-06	0.100406
916	2021-11-04	0.120423
1172	2022-11-10	0.143293
1242	2023-02-23	0.140214
1306	2023-05-25	0.243696
1492	2024-02-22	0.164009

Figure 3: Daily Returns and Outliers over Six Years

Figure 2 visualizes the change of daily returns over past six years. Red points denote the outliers based on 3 sigma-rule. As the figure shows, the largest volatility occurred in March 2020, whose daily returns fluctuated wildly between -0.185 and 0.172. This may be because Nvidia acquired Mellanox Technologies in March 11th . Extremely low values generally appeared before this period, and extremely high values appeared after this period. In 2023 May 25th, the highest return occurred, which may result form the Nvidia's financial report for February-April 2023and the stock price rose immediately. Except for these 15 outliers, the rest of the returns fluctuated steadily back and forth between +-0.1. It can be concluded that the company has experienced minimal share price fluctuations during the pandemic. These data show random variation. There are no patterns or cycles.

5.3 QQ Plot

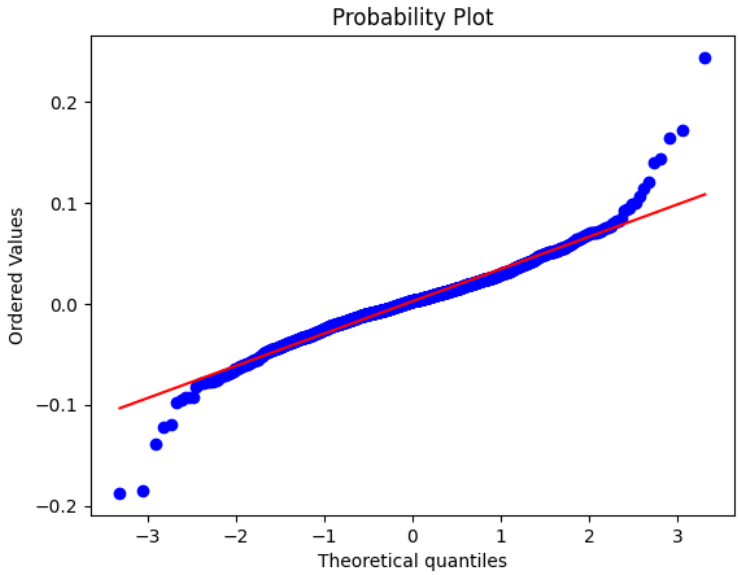

Figure 4: QQ Plot of Return

Figure 4 shows the quantiles of two distributions – daily returns and normal distribution, against each other. It can be inferred that the daily returns plausibly came from theoretical normal distribution according to the roughly straight line. There are a few points obviously deviated from the line, indicating there are more extremums compared to the normal distribution, and they mainly gather at tails.

5.4 Box Plot

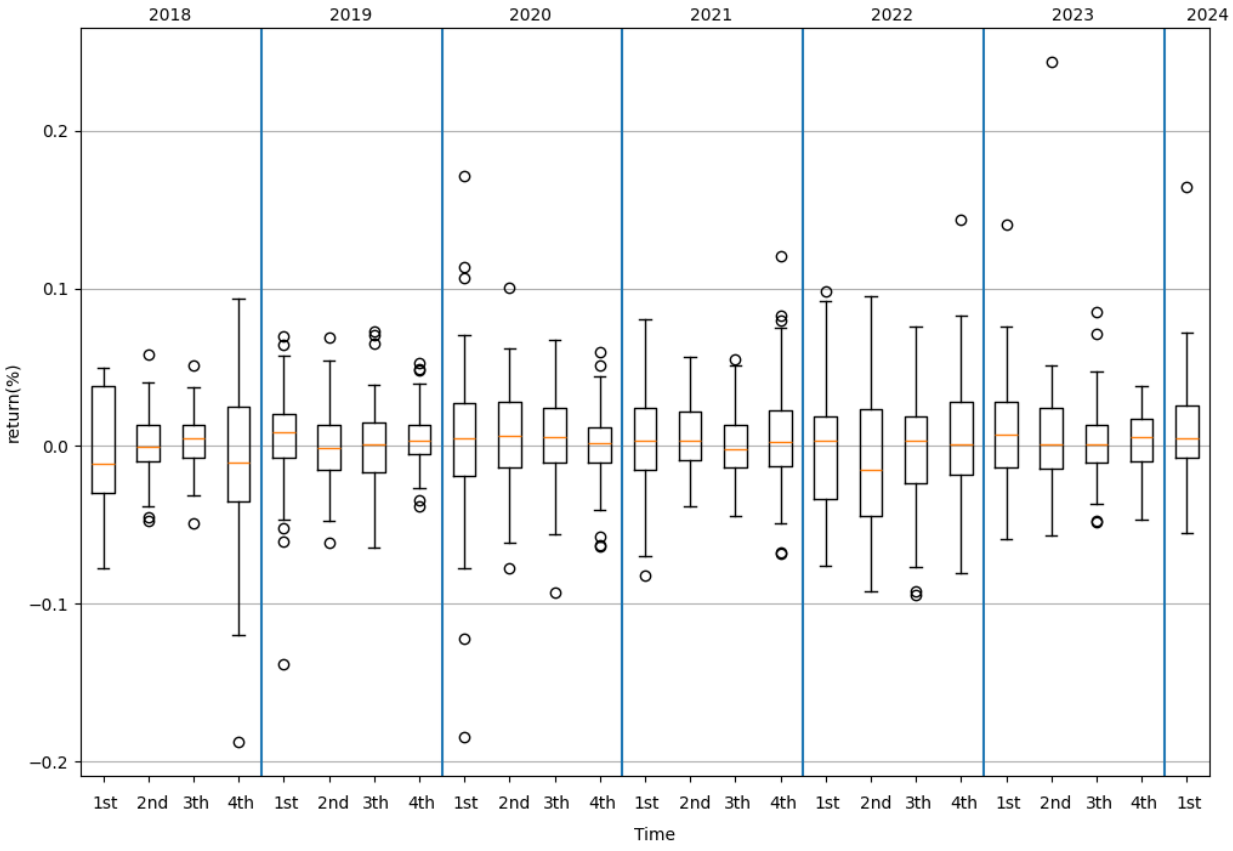

Figure 5: Box Plot of Return

Figure 5 summarizes daily returns for each quarter in past six years. There is not much difference in the median of returns in every quarter, fluctuating around 0. 2018 1st quarter has the most unstable daily returns for its largest IQR, followed by 2022 2nd quarter and 2018 4th quarter. 2018 1st quarter is most unstable mainly because it only contains the data from 2018 March 20th to 29th.

6 Prediction of Return by Forecasting Regression

6.1 XGBoost

XGBoost (eXtreme Gradient Boosting) is a scalable, portable and distributed gradient-boosted decision tree (GBDT) model. It provides parallel tree boosting and is the leading machine learning library for regression problems. The workflow of XGBoost is shown in the figure below.

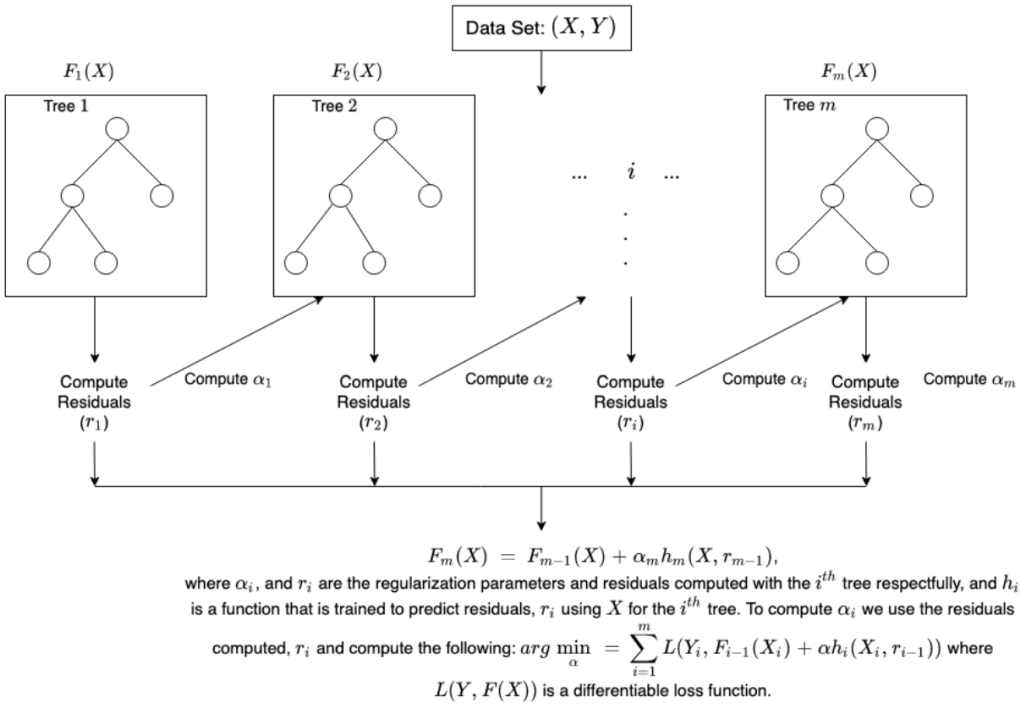

Figure 6: XGBoost Algorithm Process with Regularization and Residuals Computation[2]

For our XGBoost model, before the data preprocessing mentioned above, we also performed feature engineering. We generated the following new features:

- Mean 'Adj Close' of each month
- Difference between 'High' and 'Low' of each day
- Difference between 'Open' and 'Close' of each day
- Mean 'Volume' of each month

Then we dropped the 'High', 'Low', 'Open', and 'Close' columns, and added other columns indicating lag information. We used a total of 28 features in our XGBoost model to predict the 'Adj Close' of each day. Then we tuned the hyper parameters.

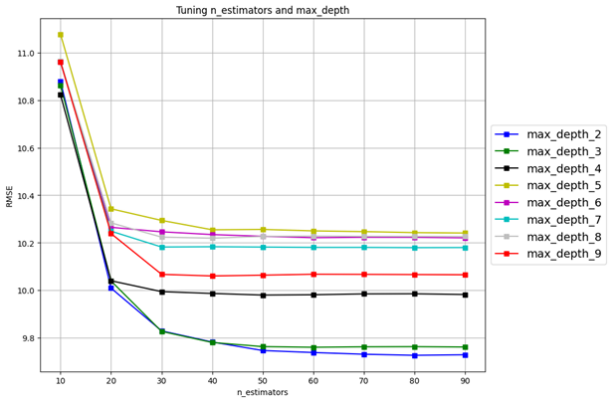

Figure 7: Hyperparameter Tuning Curve for XGBoost

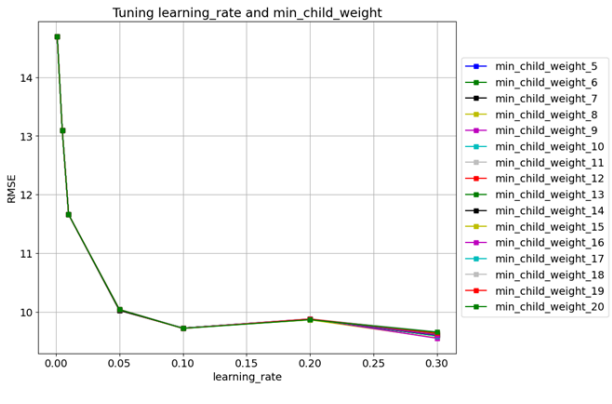

Figure 8: Learning Rate and Minimum Child Weight Tuning for XGBoost

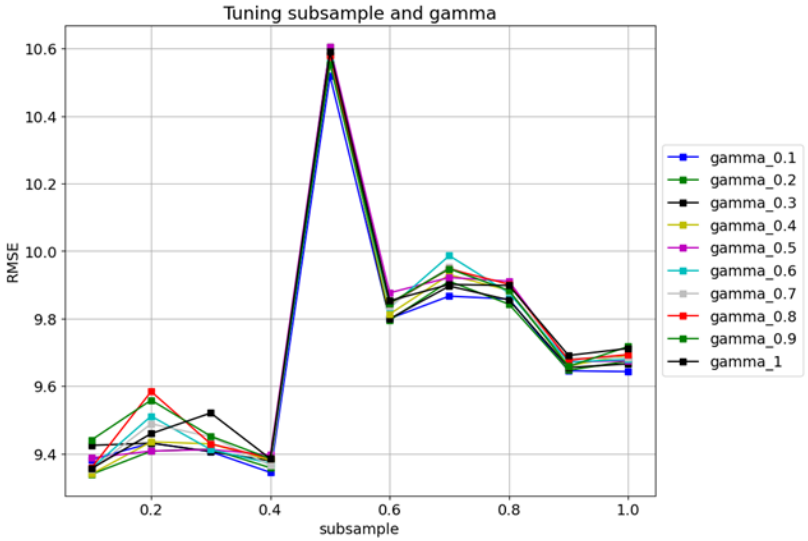

Figure 9: Subsample and Gamma Parameter Tuning for XGBoost

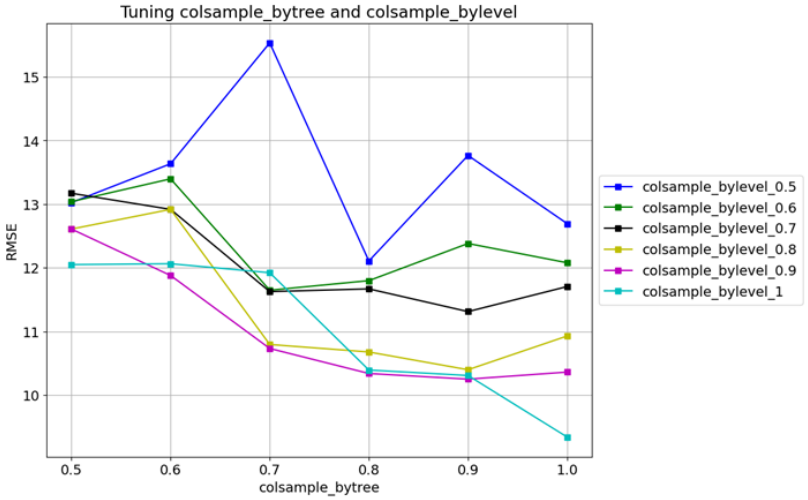

Figure 10: Column Sample by Tree and Level Tuning for XGBoost

	param	original	after_tuning
0	n_estimators	100.000	80.000
1	max_depth	3.000	2.000
2	learning_rate	0.100	0.300
3	min_child_weight	1.000	9.000
4	subsample	1.000	0.100
5	colsample_bytree	1.000	1.000
6	colsample_bylevel	1.000	1.000
7	gamma	0.000	0.200
8	rmse	9.762	9.339
9	mape_pct	2.808	2.700

Figure 11: XGBoost Hyperparameter Values Before and After Tuning

After tuning hyper parameters, our final model yielded a relatively good result:

```
RMSE on test set = 21.366
MAPE on test set = 2.355%
```

Figure 12: XGBoost Model Test Set Performance Metrics

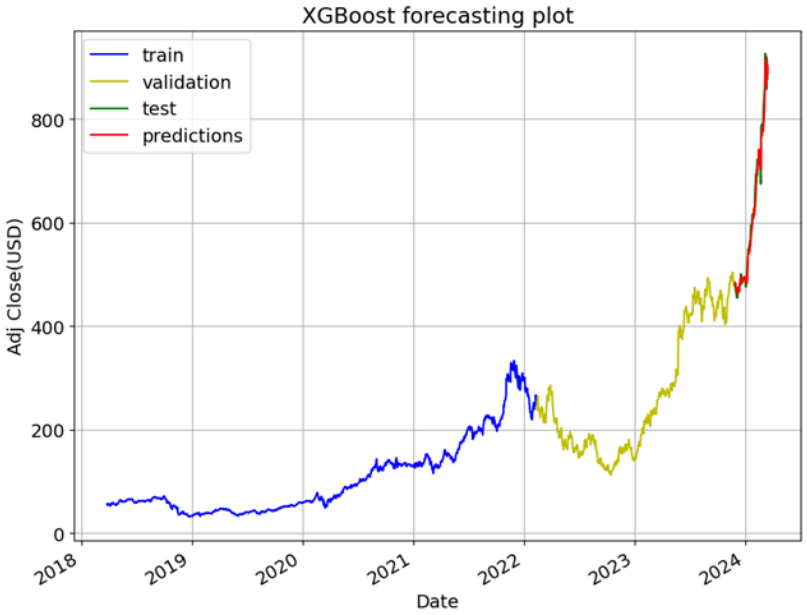

Figure 13: XGBoost Model Forecasting Results

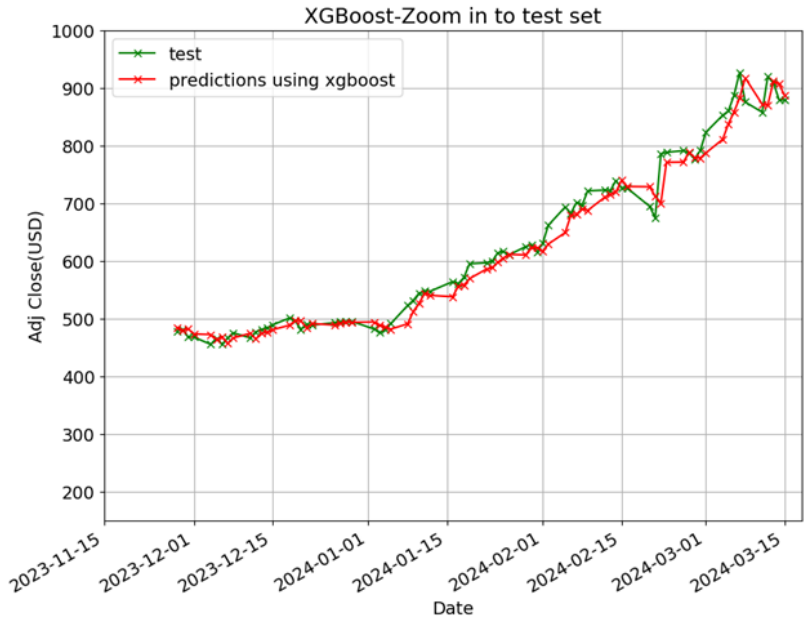

Figure 14: Detailed XGBoost Prediction Performance on Test Set

6.2 ARIMA

ARIMA (Autoregressive Integrated Moving Average) is a classic time series forecasting model used to analyze and predict data with temporal dependencies. It combines three components: autoregression (AR), differencing (I), and moving average (MA).

Autoregression captures the relationship between an observation and a certain number of lagged observations. It assumes that the current value of a variable can be predicted using its own previous values. The order of autoregression, denoted as AR(p), specifies the number of lagged terms considered in the model.

The differencing component is used to make the time series data stationary, which means removing any trend or seasonality. Differencing involves subtracting the previous observation from the current observation. The order of differencing, denoted as I(d), indicates the number of times differencing is applied to achieve stationarity.

Moving average considers the dependency between an observation and a residual error from a moving average model applied to lagged observations. It helps to capture the short-term fluctuations in the data. The order of the moving average, denoted as MA(q), specifies the number of lagged forecast errors considered in the model.

The ARIMA model combines these three components as ARIMA(p, d, q). It represents a linear regression model where the dependent variable is differenced to achieve stationarity and is then regressed on the lagged values of itself and the lagged forecast errors.

In our ARIMA forecasting model, in addition to data preprocessing mentioned above, we do the ADF (Augmented Dickey-Fuller) test to check whether the data is stationary. ADF test shows that p-value is about 0.995 which is much greater than 0.05, indicating the data is not stationary. So we do the differencing.

After differencing, p-value from ADF test became very close to 0. We also examined the ACF (autocorrelation function) and PACF (partial autocorrelation function) plots after differencing.

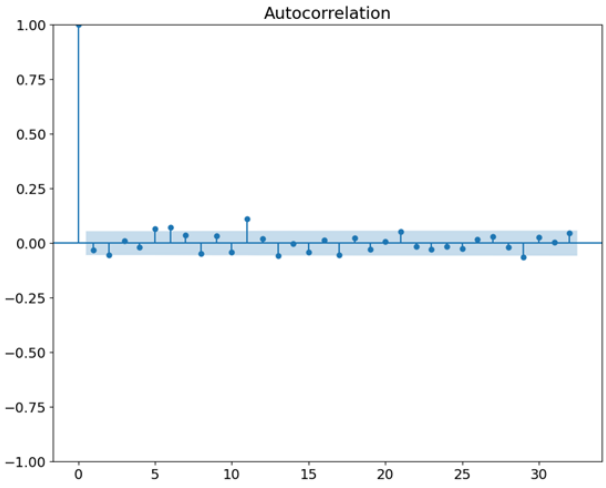

Figure 15: Autocorrelation Plot

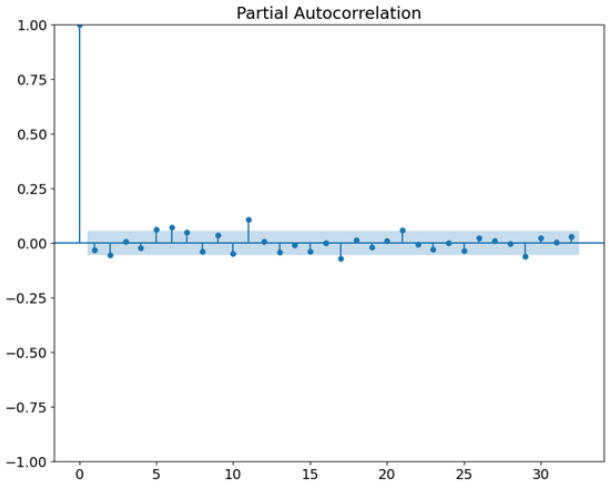

Figure 16: Partial Autocorrelation Plot

From the above results, we can determine that the order of differencing (d) equal to 1.

The parameters p and q also need to be tuned based on the characteristics of the time series data. This is done automatically using the `auto_arima` function, which fits the model multiple times to find the best model that yields the lowest Akaike Information Criterion (AIC).

Final ARIMA model result:

SARIMAX Results

Dep. Variable:	y	No. Observations:	1283
Model:	ARIMA(4, 1, 3)	Log Likelihood	1700.821
Date:	Sat, 23 Mar 2024	AIC	-3385.642
Time:	19:33:49	BIC	-3344.393
Sample:	0	HQIC	-3370.155
	- 1283		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.1404	0.093	-1.510	0.131	-0.323	0.042
ar.L2	-0.4163	0.074	-5.644	0.000	-0.561	-0.272
ar.L3	-0.7875	0.092	-8.523	0.000	-0.969	-0.606
ar.L4	-0.0923	0.020	-4.639	0.000	-0.131	-0.053
ma.L1	0.0970	0.091	1.065	0.287	-0.081	0.275
ma.L2	0.3629	0.072	5.054	0.000	0.222	0.504
ma.L3	0.7718	0.088	8.754	0.000	0.599	0.945
sigma2	0.0041	8.76e-05	47.289	0.000	0.004	0.004

Ljung-Box (L1) (Q):	0.09	Jarque-Bera (JB):	1561.32
Prob(Q):	0.76	Prob(JB):	0.00
Heteroskedasticity (H):	30.25	Skew:	0.41
Prob(H) (two-sided):	0.00	Kurtosis:	8.34

Figure 17: SARIMAX Results

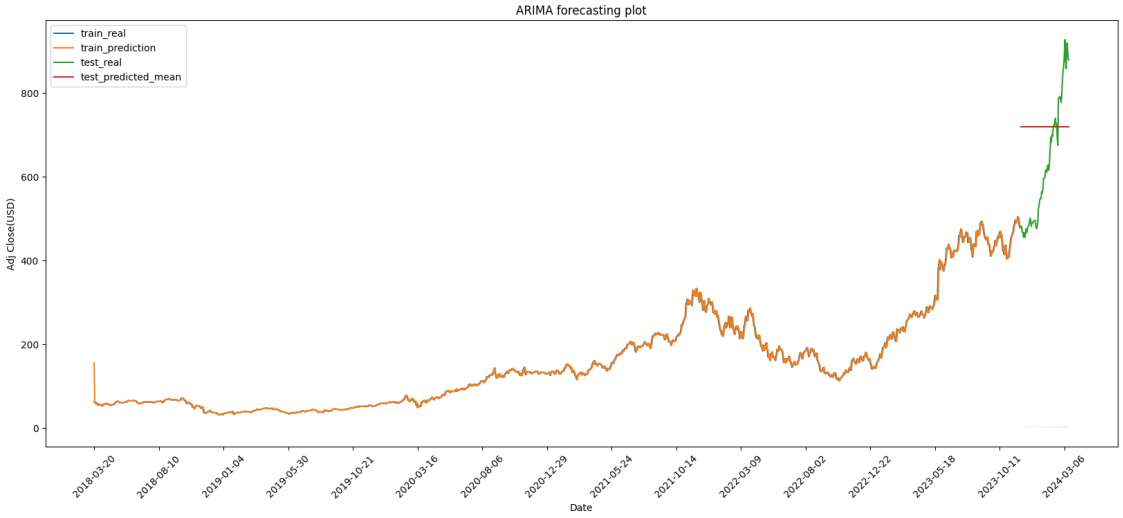

Figure 18: ARIMA Model Forecasting of Adjusted Close Prices

RMSE on test set = 430.646
MAPE on test set = 73.176%

Figure 19: ARIMA Model Prediction Accuracy

6.3 LSTM

LSTM (Long Short-Term Memory networks) is a variant of recurrent neural networks (RNNs) that is more powerful than RNNs because it can overcome the vanishing gradient problem (Hochreiter, 1998). Due to its ability to "remember" previous information, LSTM performs well in predicting data with long-term dependencies, such as time series, text, audio, etc. (Dave et al., 2021). An LSTM layer consists of memory blocks, which include an input gate, forget gate, output gate, and hidden state. The structure is illustrated in the figure below:

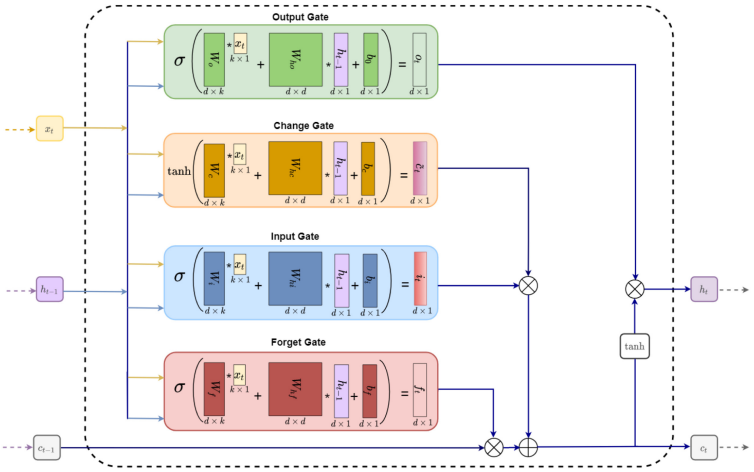

Figure 20: Long Short-Term Memory (LSTM) Cell Architecture with Gate Mechanisms[3]

The input shape of the model is (number of samples, window size, number of features). The optimal window size of the LSTM model cannot be obtained directly, so we set the window size to 6 based on the real-world information. This is because there is no stock data over the weekend, but there may be external factors that interfere with the stock price during the two days that are vacant. Window size=6 will ensure that the historical observations considered by the model at each time step are across weeks.

The model includes one LSTM layer and two Dense layers. The first layer is an LSTM layer with an output dimension of 32. It uses the ReLU function as the activation function to help the model capture nonlinear relationships. To control overfitting, dropout is set to 0.01.

The output dimensions of Dense layers are 32 and 4. The activation function in dense layers is linear because it is a regression prediction problem. The optimizer used to optimize the model weights is RMSprop, and the loss function is mean squared error (MSE).

The prediction results are shown in the figure below:

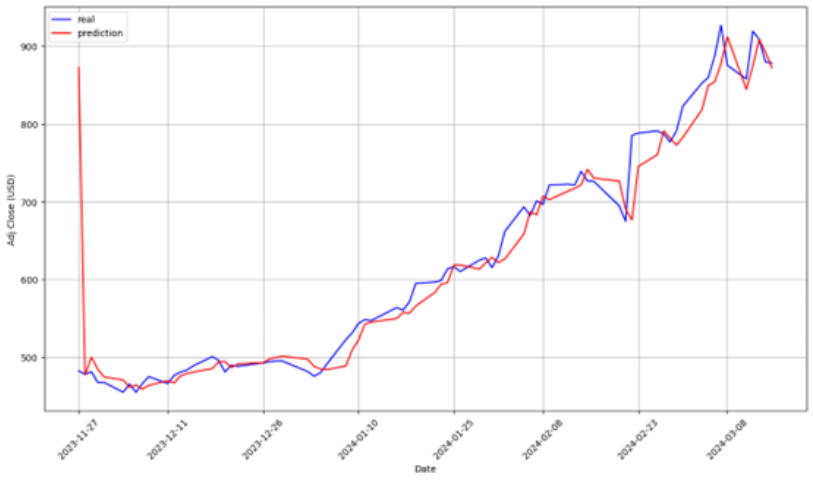

Figure 21: Time Series Forecast Comparison

RMSE on test set = 51.451
MAPE on test set = 3.204%

Figure 22: Model Performance Metrics on Test Set

6.4 VAR (Vector Auto Regression)

The VAR model is an extension of the univariate autoregressive model to dynamic multivariate time series. It has long been proven useful in forecasting financial time series.

The VAR (p), where p represents the order of model, is given by[6]:

$$y_t = A_1 y_{t-1} + A_2 y_{t-2} + \cdots + A_p y_{t-p} + \varepsilon_t, \quad t = 0, \pm 1, \pm 2, \ldots, \tag{1}$$

where $\{y_t\}$ is the time series, $A_1, \ldots, A_p$ are constant coefficient matrices. $\{\varepsilon_t\}$ is a multivariate white noise with mean vector 0 and variance-covariance matrix.

The VAR model requires the input data to be stationary, but by checking the unit root of each variable, it can be found that only the variable 'Volume' in the normalized data satisfies the input requirements of the model.

Table 1: Unit Root Test P-Values

Variable	P-Value for Unit Root Test
Open	0.9963045267850447
High	0.9958952761342585
Low	0.9917048656222253
Close	0.9916231146052235
Adj Close	0.991700676998527
Volume	$5.230864123455103e-24$

After performing a first-order difference on a stationary variable, a picture of a stationary Adj Close can be drawn as follows:

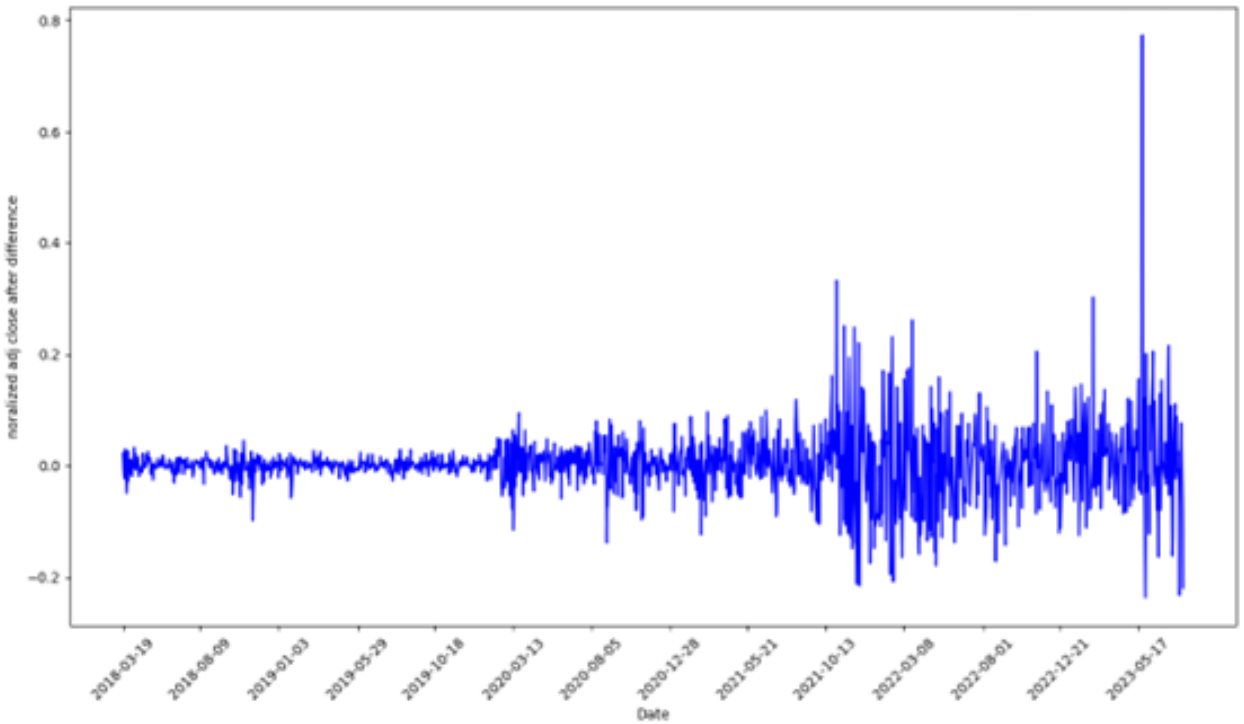

Figure 23: Stationary Time Series of First-Order Differenced Adjusted Close Values

There are various methods for parameter estimation in the VAR model, and in this project, ordinary least squares (OLS) is used. To estimate the VAR model, it is necessary to first determine the lag order of the endogenous variables, denoted as p. We used criteria AIC, where smaller values are preferred.

	P	AIC
0	1	-46.938776
1	2	-47.146798
2	3	-46.509605
3	4	-44.722255
4	5	-41.613994
5	6	-36.878798
6	7	-30.184242
7	8	-21.273357
8	9	-9.725205

Figure 24: VAR Model Lag Order Selection Based on AIC Values

The minimum values for AIC are around -47, and p is set to 2. We use the VAR(2) model of OLS to estimate the parameters for prediction and restore the predicted results back to their own data range. The prediction results are shown in the figure below:

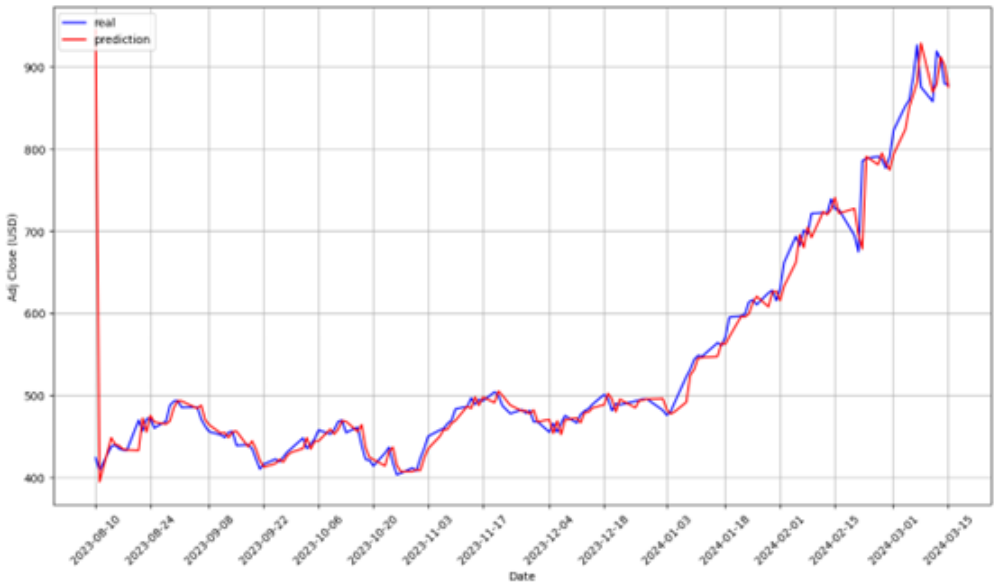

Figure 25: Comparison of Actual vs. VAR(2) Predicted Values Over Time

```
RMSE on test set = 45.433
MAPE on test set = 2.760%
```

Figure 26: VAR(2) Model Prediction Accuracy

6.5 Results

Based on models' performance on the same test set, we evaluated the RMSE and MAPE. For these two metrics, the lower they are, the better the model performs in predicting NVIDIA's adjusted close price. After comparing the RMSE and MAPE of the four models, we found that although there was some lag in the forecast, XGBoost preformed well in this task, with the lowest RMSE and MAPE. Three main reasons for XGBoost's good performance are: Firstly, we did detailed and reasonable feature engineering when choosing predictors. Secondly, XGBoost can handle non-linear relationship in time series data. By combining multiple decision trees, it can capture intricate interactions and nonlinearity in the data, enabling it to handle complex problems where linear models may not be sufficient. Thirdly, it uses regularization techniques such as L1 (LASSO) and L2 (Ridge) regularization, which control the complexity of the model and prevent overfitting. Regularization helps to generalize the model and reduce the impact of irrelevant or noisy features, improving the model's robustness and performance on unseen data. VAR, LSTM, and ARIMA didn't perform well in this task. The main reason may be that these three models cannot handle nonlinear relationships well. Also, LSTM and ARIMA models cannot effectively capture the dependence of stock prices on longer time scales, resulting in inaccurate prediction results. ARIMA model did well in fitting the training set, but performed poorly on the test set. One possible reason for this is that ARIMA can only get the mean value of predictions in the whole test set.

7 Conclusion

This study conducted a comprehensive analysis of NVIDIA Corporation, aimed at exploring stock returns and predicting stock prices. Through an in-depth exploratory analysis, we noted that despite significant fluctuations in certain periods, median returns remained stable over time. In the forecasting phase, we utilized four advanced models: XGBoost, ARIMA, LSTM, and VAR, to precisely predict the adjusted closing prices.

The results demonstrated that the XGBoost model surpassed its counterparts in prediction accuracy, thanks to its effective feature engineering, capability to manage non-linear relationships, and the use of regularization techniques to prevent overfitting. However, VAR, LSTM, and ARIMA models showed limitations in capturing the non-linear dynamics and long-term dependencies characteristic of stock price movements, leading to less precise forecasts.

Our analysis was limited by the available historical data, which may not fully account for future market conditions or the impact of unforeseen external factors. Additionally, the inherent unpredictability of stock prices presents an ongoing challenge to predictive modeling.

Future research should aim to expand the dataset to incorporate a more diverse range of information sources and explore additional predictive models that could yield new insights into stock price movements. Enhancing feature engineering and further tuning of model parameters are also likely to improve forecast accuracy.

8 Appendix

Data source: https://finance.yahoo.com/quote/NVDA?.tsrc=fin-srch
Our `Python` source code can be found here:
`https://github.com/Fiona1224/XGBoost-NVIDIA-stock-price-prediction`
`https://github.com/YOLANDALI/Stock3022`

References

[1] Tianfeng Securities Co., Ltd., *FY2024 Q4 review: Q4 and annual performance exceed expectations, strong growth in data center and gaming business, substantial increase in net profit. Research Report*, Tianfeng Securities Co., Ltd., 2024. Accessed on: 2024-03-09. URL: `https://www.hangyan.co/reports/3322705861360485589`.

[2] Amazon Web Services, Inc., *How XGBoost Works*, Available at: `https://docs.aws.amazon.com/sagemaker/latest/dg/xgboost-HowItWorks.html` (pic source).

[3] Hum Nath Bhandari, Binod Rimal, Nawa Raj Pokhrel, Ramchandra Rimal, Keshab R. Dahal, Rajendra K.C. Khatri, *Predicting stock market index using LSTM*, Machine Learning with Applications, Volume 9, 2022, 100320, ISSN 2666-8270, https://doi.org/10.1016/j.mlwa.2022.100320.

[4] E. Dave, A. Leonardo, M. Jeanice, N. Hanafiah, Forecasting Indonesia exports using a hybrid model ARIMA-LSTM, Procedia Comput. Sci., 179 (2021), pp. 480-487.

[5] S. Hochreiter, The vanishing gradient problem during learning recurrent neural nets and problem solutions, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 6 (02) (1998), pp. 107-116.

[6] X. Li and J. Yuan, *DeepVARwT: Deep Learning for a VAR Model with Trend*, ArXiv, abs/2209.10587, 2022.