BTP - II

Final Report
Volatility Forecasting: Evidence from India

Aaryan (19D070001)

May 3, 2023

Guided by **Prof. Piyush Pandey & Prof. V Rajbabu**
Indian Institute of Technology Bombay

Contents

1 Abstract

Volatility forecasting is an essential area of research in finance that aims to predict the level of uncertainty in financial markets, as measured by the variability of asset prices. The accurate estimation of volatility is crucial for various applications, including risk management, portfolio optimization, and pricing of financial instruments. In recent years, there has been a growing interest in the use of advanced machine learning (ML) methods for volatility forecasting, due to their ability to capture complex non-linear relationships in financial data.

This paper presents an analysis of the performance of several ML models, including random forest (RF), artificial neural network (ANN), recurrent neural network (RNN), and long short-term memory (LSTM), for volatility forecasting. The models are evaluated using a dataset of daily stock returns for a selection of major global stock indices, and their performance is compared to that of the benchmark autoregressive integrated moving average (ARIMA) model.

The results show that the ML models generally outperform the ARIMA model in terms of forecasting accuracy, with the LSTM model achieving the best results overall. The study also highlights the importance of selecting appropriate input variables and model parameters for ML-based volatility forecasting. These findings have important implications for practitioners in finance, who can use ML methods to improve their predictions of volatility and make better-informed investment decisions.

2 Introduction

Volatility, as defined in finance, refers to the degree of variation in the price of a financial instrument over a specific period. It is a measure of the level of uncertainty in the market and is commonly used as an indicator of risk. Volatility can be estimated using various statistical methods, and accurate forecasts of volatility are essential for a wide range of financial applications, including portfolio optimization, risk management, and derivative pricing.

In recent years, there has been an increased demand for volatility trading strategies as more investors recognize the potential benefits of investing in assets with high volatility. For instance, volatility trading can provide a hedge against market risks or generate profits from trading volatility itself. Consequently, there is a growing need for accurate and reliable volatility forecasting methods that can be used to develop and implement effective trading strategies.

Machine learning (ML) methods have become popular in finance due to their ability to handle large and complex datasets and capture non-linear relationships between variables. In particular, ML methods have shown promise in improving the accuracy of volatility forecasting, which is crucial for effective volatility trading strategies. This paper presents a comparative analysis of several advanced ML models for volatility forecasting, including random forest (RF), artificial neural network (ANN), recurrent neural network (RNN), and long short-term memory (LSTM), with a benchmark autoregressive integrated moving average (ARIMA) model.

The paper aims to evaluate the performance of these models in forecasting volatility specifically in the context of the Indian markets and to highlight the importance of selecting appropriate input variables and model parameters for ML-based volatility forecasting. No research in this regard have been performed in case of emerging markets.

3 Data

The study used a dataset comprising daily data of Nifty30d volatility for the period of April 2008 to April 2023. Nifty30d volatility is a forward-looking measure of volatility in the Indian stock market, which is derived from the implied volatility of the next expiring options on the Nifty index. The dataset was obtained from Bloomberg.

In addition to Nifty30d volatility, the study also utilized several other variables that are believed to have an economic significance and are potential factors that could affect the Indian market volatility. These variables include:

- Nifty close prices: The daily closing prices of the Nifty 50 index were used as a proxy for the overall performance of the Indian stock market. Nifty close prices are considered a crucial factor in predicting future volatility as they reflect the behavior of the market participants.
- Nifty volume: The daily trading volume of the Nifty 50 index was included as a factor in the study as it is an important indicator of the level of market activity and liquidity. Higher trading volumes are often associated with higher volatility.
- CBOE VIX: The Chicago Board Options Exchange (CBOE) Volatility Index (VIX) was included in the study as a measure of the market's expectation of future volatility of the $S\&P$ 500 index. The VIX is often used as a proxy for global risk sentiment and can impact the Indian market volatility.
- US 10-year Treasury yield: The daily yield on US 10-year Treasury bonds was included in the study as a proxy for the global risk-free rate. The yield on the US Treasury bonds can impact the cost of capital for investors and hence affect the volatility of emerging market assets.
- LIBOR: The daily London Interbank Offered Rate (LIBOR) was included as a measure of the global interbank lending rate. Changes in LIBOR rates can affect the borrowing costs for market participants and impact the volatility of the Indian stock market.
- WTI crude oil futures: The daily prices of West Texas Intermediate (WTI) crude oil futures were included as a proxy for the global oil market. Changes in oil prices can impact the profitability and cost structure of firms in various sectors and can affect market volatility.
- Gold: The daily prices of gold were included in the study as a proxy for the global safe-haven asset. Changes in gold prices can reflect shifts in the global risk sentiment and can impact the volatility of emerging market assets.

4 Time Series Forecasting

Time series forecasting is a crucial task for many industries, including finance, economics, and marketing. It involves using statistical models to predict future values of a variable based on its past behavior. ARIMA models are one of the most popular statistical models used for time series forecasting due to their ability to capture trends and seasonality in the data. In this article, we will explore what ARIMA models are, how they work, and how to use them for time series forecasting

4.1 The AR Model

An Autoregressive (AR) model is a time series model that uses past values of the variable to predict future values. The AR model assumes that the current value of the variable is a linear combination of its past values, with the coefficients of the past values determined by the model.

The AR model can be written mathematically as:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + \epsilon_t \tag{1}$$

where y_t is the value of the variable at time t, c is a constant, $\phi_1, \phi_2, ..., \phi_p$ are the parameters of the model, and ϵ_t is the error term at time t.

The error term ϵ_t is assumed to be normally distributed with a mean of 0 and a constant variance. The values of the parameters $\phi_1, \phi_2, ..., \phi_p$ are estimated using a method called maximum likelihood estimation, which seeks to find the values of the parameters that maximize the likelihood of the observed data.

The order of the AR model is the number of lagged values used to predict the current value. A higher-order AR model will use more past values to predict the current value and may capture more complex patterns in the time series data. However, a higher-order model may also result in over-fitting the data, which can lead to poor out-of-sample performance.

To estimate the parameters of the AR model, we first need to check if the time series data is stationary. A stationary time series has a constant mean and variance over time, and the autocorrelation between any two points in time only depends on the distance between the two points and not on the absolute time. If the time series is not stationary, we can transform it into a stationary time series using techniques such as differencing.

Once we have a stationary time series, we can estimate the parameters of the AR model using maximum likelihood estimation. This involves finding the values of the parameters that maximize the likelihood of the observed data, given the assumptions of the model.

To make predictions using the AR model, we first estimate the values of the parameters using the observed data. We then use the estimated parameters and the past values of the variable to predict future values. For example, to predict the value of the variable at time t+1, we would use the estimated parameters and the values of the variable at times t, t-1, ..., t-p to predict the value at time t+1.

4.2 The MA Model

The Moving Average (MA) model is a time series model that uses past prediction errors to predict future values of a variable. The model assumes that the current value of the variable is a function of the past prediction errors.

The MA model can be written mathematically as:

$$y_t = c + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + ... + \theta_q \epsilon_{t-q} + \epsilon_t \tag{2}$$

where y_t is the value of the variable at time t, c is a constant, $\theta_1, \theta_2, ..., \theta_q$ are the parameters of the model, and ϵ_t is the error term at time t.

The error term ϵ_t is assumed to be normally distributed with a mean of 0 and a constant variance. The values of the parameters $\theta_1, \theta_2, ..., \theta_q$ are estimated using a method called maximum likelihood

estimation, which seeks to find the values of the parameters that maximize the likelihood of the observed data.

The prediction errors ϵ_t are calculated as the difference between the actual value of the variable at time t and the predicted value of the variable at time t based on the previous q prediction errors. For example, to calculate the prediction error ϵ_t for an MA(1) model, we would first use the estimated parameter θ_1 and the prediction error ϵ_{t-1} to predict the value of the variable at time t, and then calculate the prediction error as the difference between the actual value of the variable at time t and the predicted value.

The MA model is useful for modeling time series data that exhibit a moving average pattern, where the values of the variable at different points in time are influenced by past prediction errors. For example, the stock prices of a company may be influenced by the prediction errors of its competitors or the overall performance of the stock market.

The order of the MA model is the number of lagged prediction errors used to predict the current value. A higher-order MA model will use more past prediction errors to predict the current value and may capture more complex patterns in the time series data. However, a higher-order model may also result in overfitting the data, which can lead to poor out-of-sample performance.

4.3 The ARMA Model

An Autoregressive Moving Average (ARMA) model is a time series model that combines the concepts of the Autoregressive (AR) model and the Moving Average (MA) model. The ARMA model is used to capture both the trend and the random variation in the time series data.

The ARMA model is defined by two parameters, p and q, which represent the number of lags used in the AR and MA components of the model, respectively. The AR component of the model captures the trend in the data, while the MA component of the model captures the random variation.

The ARMA model can be written mathematically as:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + ... + \phi_p y_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + ... + \theta_q \epsilon_{t-q} \tag{3}$$

where y_t is the value of the variable at time t, c is a constant, $\phi_1, \phi_2, ..., \phi_p$ are the parameters of the AR component, $\theta_1, \theta_2, ..., \theta_q$ are the parameters of the MA component, ϵ_t is the error term at time t, and p and q are the orders of the AR and MA components, respectively.

The error term ϵ_t is assumed to be normally distributed with a mean of 0 and a constant variance. The values of the parameters $\phi_1, \phi_2, ..., \phi_p$ and $\theta_1, \theta_2, ..., \theta_q$ are estimated using maximum likelihood estimation, which seeks to find the values of the parameters that maximize the likelihood of the observed data.

The order of the AR and MA components of the ARMA model is typically determined using the autocorrelation function (ACF) and partial autocorrelation function (PACF) of the time series data. The ACF measures the correlation between the variable and its lagged values, while the PACF measures the correlation between the variable and its lagged values after controlling for the effect of intermediate lags. The ACF and PACF plots can help identify the orders of the AR and MA components.

However, in some cases, the time series data may also exhibit non-stationarity, which means that the mean and/or variance of the data changes over time. To model non-stationary time series data, we can use a method called differencing. Differencing involves taking the difference between consecutive values of the time series data to make it stationary. The differenced data can then be modeled using the ARMA model.

4.4 The ARIMA Model

When dealing with time series data, it is common to encounter patterns such as trend and seasonality. A trend refers to a long-term increase or decrease in the values of the time series data, while seasonality refers to a pattern that repeats itself over a fixed time interval.

To capture both trend and seasonality in the time series data, we can use a more complex version of the ARMA model called the Autoregressive Integrated Moving Average (ARIMA) model. The ARIMA model includes an additional parameter, d, which represents the degree of difference needed to make the time series data stationary. The "I" in ARIMA stands for "integrated", which refers to the differencing operation.

The ARIMA model can be written mathematically as:

$$y'_t = c + \phi_1 y'_{t-1} + \phi_2 y'_{t-2} + ... + \phi_p y'_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + ... + \theta_q \epsilon_{t-q} \tag{4}$$

where y'_t is the differenced value of the variable at time t, c is a constant, $\phi_1, \phi_2, ..., \phi_p$ are the parameters of the autoregressive (AR) component, $\theta_1, \theta_2, ..., \theta_q$ are the parameters of the moving average (MA) component, ϵ_t is the error term at time t, and p and q are the orders of the AR and MA components, respectively.

The differenced data is obtained by taking the difference between consecutive values of the time series data. Differencing is used to make the time series data stationary, which means that the mean and/or variance of the data does not change over time. Stationarity is important for time series modeling because it allows us to apply statistical techniques that assume the data has a constant mean and variance.

To determine the degree of differencing needed to make the time series data stationary, we can use a method called the Augmented Dickey-Fuller (ADF) test. The ADF test is a statistical test that tests for the presence of a unit root in the time series data, which indicates non-stationarity. If the ADF test indicates the presence of a unit root, then differencing can be applied to the data to make it stationary.

Once the differenced data is stationary, we can use the ACF and PACF plots to determine the orders of the AR and MA components of the ARIMA model. The ACF and PACF plots can help identify the lagged correlations in the data, which can be used to determine the orders of the AR and MA components.

To fit the ARIMA model to the data, we can use the method of maximum likelihood estimation. Maximum likelihood estimation seeks to find the values of the parameters $\phi_1, \phi_2, ..., \phi_p$, $\theta_1, \theta_2, ..., \theta_q$, and c that maximizes the likelihood of the observed data.

Once the ARIMA model is fitted to the data, we can use it to make forecasts for future values of the time series data. The forecasting process involves using the estimated model parameters to generate forecasts for future values of the time series data.

4.5 SARIMA Model

The Seasonal Autoregressive Integrated Moving Average (SARIMA) model is an extension of the ARIMA model that can handle time series data with both trend and seasonality. The SARIMA model includes additional parameters to capture the seasonal patterns in the data.

Seasonality refers to patterns in the data that repeat themselves over fixed time intervals, such as daily, weekly, or monthly. These seasonal patterns can have a significant impact on the time series data and can be difficult to capture using traditional ARIMA models.

The SARIMA model extends the ARIMA model by including seasonal differences and seasonal AR and MA terms. The seasonal differences refer to the differences between values at the same time point in different seasonal periods. For example, the difference between the value at time t and the value at time t-12 for monthly data.

The seasonal AR and MA terms capture the seasonal patterns in the data by including lagged values of the seasonal differences in the model. The seasonal AR and MA terms are denoted as SAR and SMA, respectively.

The SARIMA model can be written mathematically as:

$$y'_t = c + \phi_1 y'_{t-1} + \phi_2 y'_{t-2} + ... + \phi_p y'_{t-p} + \Phi_1 y'_{t-s} + \Phi_2 y'_{t-2s} + ... + \Phi_P y'_{t-Ps} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + ... + \theta_q \epsilon_{t-q} + \Theta_1 \epsilon_{t-s} + \Theta_2 \epsilon_{t-2s} + ... + \Theta_Q \epsilon_{t-Qs} \quad (5)$$

where y'_t is the differenced value of the variable at time t, c is a constant, $\phi_1, \phi_2, ..., \phi_p$ are the parameters of the autoregressive (AR) component, $\Phi_1, \Phi_2, ..., \Phi_P$ are the parameters of the seasonal autoregressive (SAR) component, $\theta_1, \theta_2, ..., \theta_q$ are the parameters of the moving average (MA) component, $\Theta_1, \Theta_2, ..., \Theta_Q$ are the parameters of the seasonal moving average (SMA) component, ϵ_t is the error term at time t, s is the length of the seasonal cycle, and p, q, P, and Q are the orders of the AR, MA, SAR, and SMA components, respectively.

The orders of the AR, MA, SAR, and SMA components can be determined using the ACF and PACF plots of the differenced and seasonally differenced data. The orders can be selected based on the significant lags in the ACF and PACF plots.

The SARIMA model can be fitted to the data using the method of maximum likelihood estimation. Once the model is fitted to the data, it can be used to make forecasts for future values of the time series data.

The SARIMA model is important because it can handle time series data with both trend and seasonality. The seasonal patterns in the data can have a significant impact on the time series data and can be difficult to capture using traditional ARIMA models. The SARIMA model provides a more robust approach to modeling seasonal time series data and can lead to more accurate forecasts.

4.6 Implementing SARIMA Model

4.6.1 Order of the Model

In time series analysis, selecting the optimal order of the ARIMA or SARIMA model is a critical step in the forecasting process. One common method for selecting the order is to use the Akaike Information Criterion (AIC).

The Akaike Information Criterion (AIC) is a statistical measure of the quality of a model that estimates the relative amount of information lost by the model in comparison to the true data generating process. It is defined as AIC = -2log(L) + 2k, where L is the likelihood of the data given the model and k is the number of parameters in the model. The goal is to minimize the AIC, which balances the goodness of fit of the model (measured by the likelihood function) and the complexity of the model (measured by the number of parameters). By minimizing the AIC, we can select the order that provides the best trade-off between model fit and model complexity.

The Bayesian Information Criterion (BIC) is a similar measure that places a stronger penalty on models with more parameters. It is defined as BIC = -2log(L) + klog(n), where n is the number

of observations in the time series. Like AIC, the goal is to minimize the BIC to select the optimal order of the SARIMA model.

Determining the optimal values for these parameters can be a challenging task. However, the auto arima function in Python's statsmodels library uses an iterative approach to automatically find the optimal values for these parameters based on minimizing the AIC (Akaike Information Criterion) or BIC (Bayesian Information Criterion). The auto arima function saves a significant amount of time and effort by automating the process of selecting the order of the SARIMA model.

For our data, the order is (2,1,2)(0,0,0,20).

4.6.2 Seasonality

In Python, we can use the seasonal decomposition command from the statsmodels library to check for seasonality in a time series. The seasonal decomposition command separates the time series into its three components: trend, seasonality, and noise. We can then visualize the seasonality component to see if there is any clear pattern or seasonality present in the data. If there is a repeating pattern in the seasonality component, it suggests that the time series data is seasonal.

In our data, we have checked for seasonality using m = 20 (trading days in a month), so as to capture monthly repeating patterns.

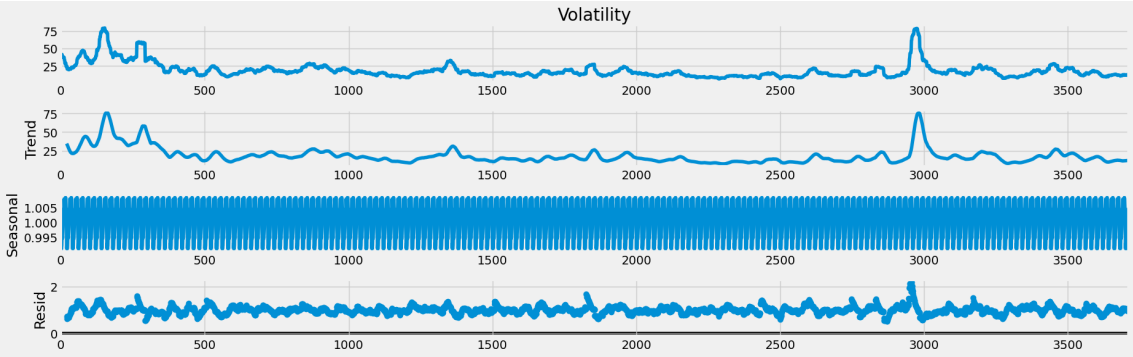


Figure 1: Seasonal decomposition of time-series data

4.6.3 Model Performance

Using a train-test-split of 0.9, the model has been trained on the train-data and used to forecast the test-data. The fitted diagnostics have been shown in Fig 2 and the forecast plot has been shown in Fig 3. The R2 score obtain was **96%** with an MSE of **1.07**.

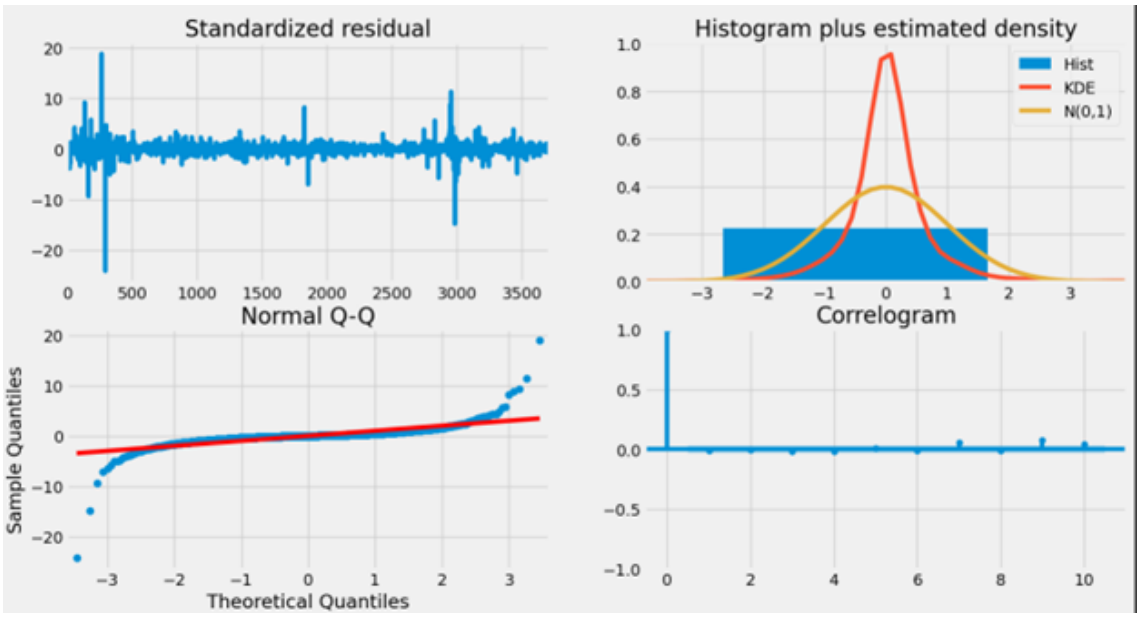


Figure 2: Diagnostics of the trained Model

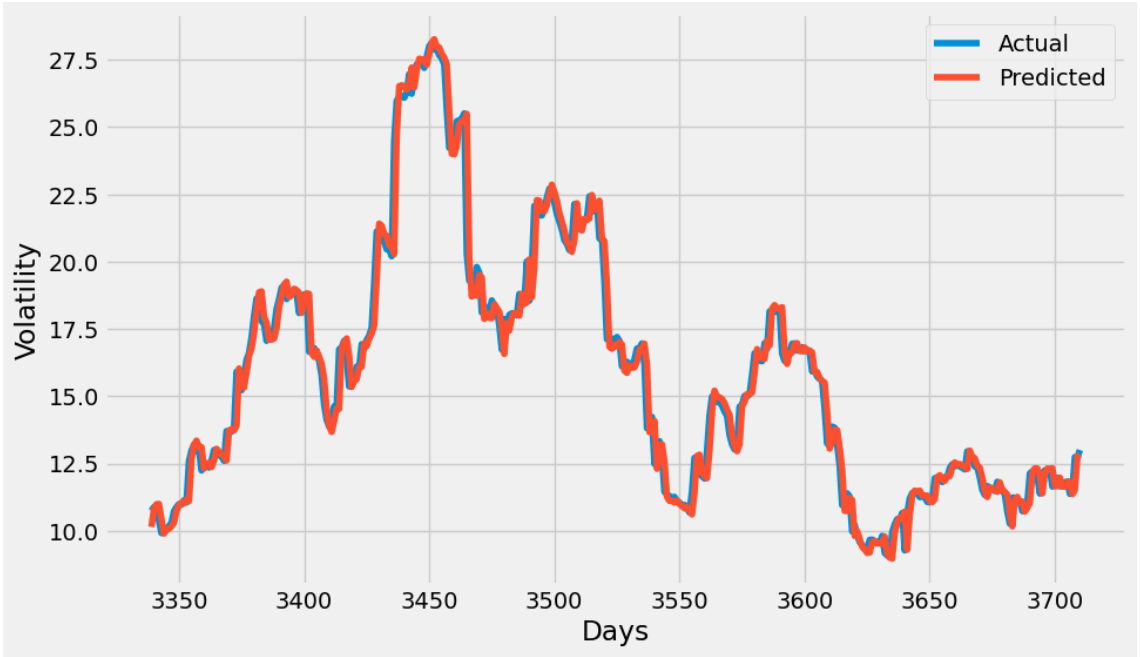


Figure 3: Forecast results

4.7 The SARIMAX Model

The Seasonal Autoregressive Integrated Moving Average with eXogenous variables (SARIMAX) model is an extension of the SARIMA model that can handle time series data with both trend,

seasonality, and exogenous variables. The SARIMAX model includes additional parameters to capture the impact of exogenous variables on the time series data.

Exogenous variables refer to external factors that can influence the time series data, such as weather patterns, economic indicators, or marketing campaigns. The inclusion of exogenous variables in the model can lead to more accurate forecasts by accounting for the impact of these external factors.

The SARIMAX model extends the SARIMA model by including exogenous variables in the model. The impact of the exogenous variables on the time series data is captured through the use of regression coefficients. The regression coefficients are denoted as β and represent the change in the time series data for a one-unit change in the exogenous variable.

The SARIMAX model can be written mathematically as:

$$\begin{aligned} y'_t = c + \phi_1 y'_{t-1} + \phi_2 y'_{t-2} + ... + \phi_p y'_{t-p} + \Phi_1 y'_{t-s} + \Phi_2 y'_{t-2s} + ... + \Phi_P y'_{t-Ps} \\ + \beta_1 x_{1,t} + \beta_2 x_{2,t} + ... + \beta_k x_{k,t} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + ... + \theta_q \epsilon_{t-q} \\ + \Theta_1 \epsilon_{t-s} + \Theta_2 \epsilon_{t-2s} + ... + \Theta_Q \epsilon_{t-Qs} \end{aligned} \quad (6)$$

where y'_t is the differenced value of the variable at time t, c is a constant, $\phi_1, \phi_2, ..., \phi_p$ are the parameters of the autoregressive (AR) component, $\Phi_1, \Phi_2, ..., \Phi_P$ are the parameters of the seasonal autoregressive (SAR) component, $\beta_1, \beta_2, ..., \beta_k$ are the regression coefficients for the exogenous variables $x_{1,t}, x_{2,t}, ..., x_{k,t}$, $\theta_1, \theta_2, ..., \theta_q$ are the parameters of the moving average (MA) component, $\Theta_1, \Theta_2, ..., \Theta_Q$ are the parameters of the seasonal moving average (SMA) component, ϵ_t is the error term at time t, s is the length of the seasonal cycle, and p, q, P, and Q are the orders of the AR, MA, SAR, and SMA components, respectively.

The orders of the AR, MA, SAR, and SMA components can be determined using the ACF and PACF plots of the differenced and seasonally differenced data. The orders can be selected based on the significant lags in the ACF and PACF plots.

The SARIMAX model can be fitted to the data using the method of maximum likelihood estimation. Once the model is fitted to the data, it can be used to make forecasts for future values of the time series data, accounting for the impact of the exogenous variables.

The SARIMAX model is important because it can handle time series data with both trend, seasonality, and exogenous variables. The inclusion of exogenous variables in the model can lead to more accurate forecasts by accounting for the impact of external factors on the time series data. The SARIMAX model provides a more robust approach to modeling time series data with external factors and can lead to more accurate forecasts.

4.8 Implementing SARIMAX Model

Apart from improving Model performance, there is another reason to introduce exogenous variables. The purpose of implementing the SARIMAX Model concerns with the fund manager's objectives to study the variation of market volatility with various macro-economic variables and other market data.

4.8.1 Order of the Model

When selecting the optimal order of the SARIMAX model with exogenous variables, the same approach can be used as for a regular SARIMA model, by minimizing the AIC or BIC. The order

of the SARIMAX model includes the number of exogenous variables to be included in the model, denoted by the variable x in the order notation (p, d, q, P, D, Q, m, x).

It is important to note that the exogenous variables do not need to be stationary, as the stationary transformation is only applied to the time series being forecasted. However, it is still important to preprocess the exogenous variables to ensure they are in an appropriate format for the SARIMAX model. For example, categorical variables may need to be encoded as numerical variables before being included in the model. Additionally, outliers and missing values in the exogenous variables should be addressed before fitting the SARIMAX model.

4.8.2 Model Performance

Using a train-test-split of 0.9, the model has been trained on the train-data and used to forecast the test-data. The exogenous variable introduced have caused to increase the model performance, however, the effect is minimal.

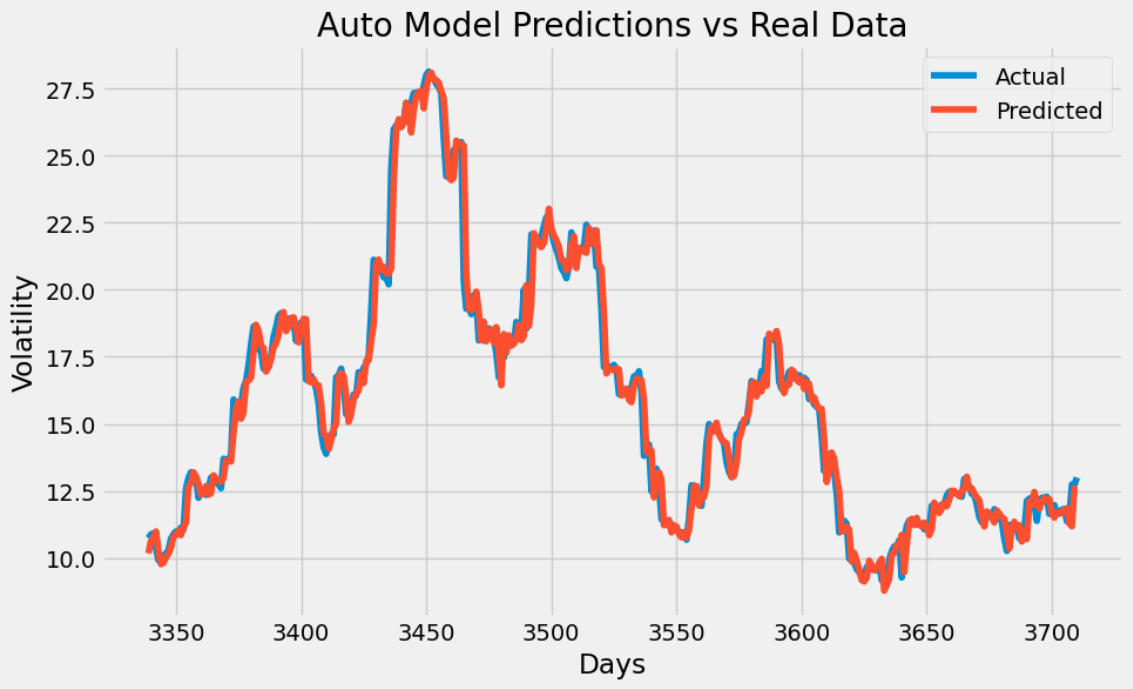


Figure 4: Model performance

4.8.3 Analysing Feature Significance

Coming to the other major reason to implement this model, we analyze the statistical significance of each of these features (exogenous variables) to accurately determine which variables significantly affect Indian market volatility. This has been captured using p-value of the variables and we see that all the variables except Crude Oil futures and Gold have p-values less than 0.05, i.e., significant at 5% confidence. Even if these variables are removed no effect is seen on the model performance, the feature importance, or the significance of other exogenous variables.

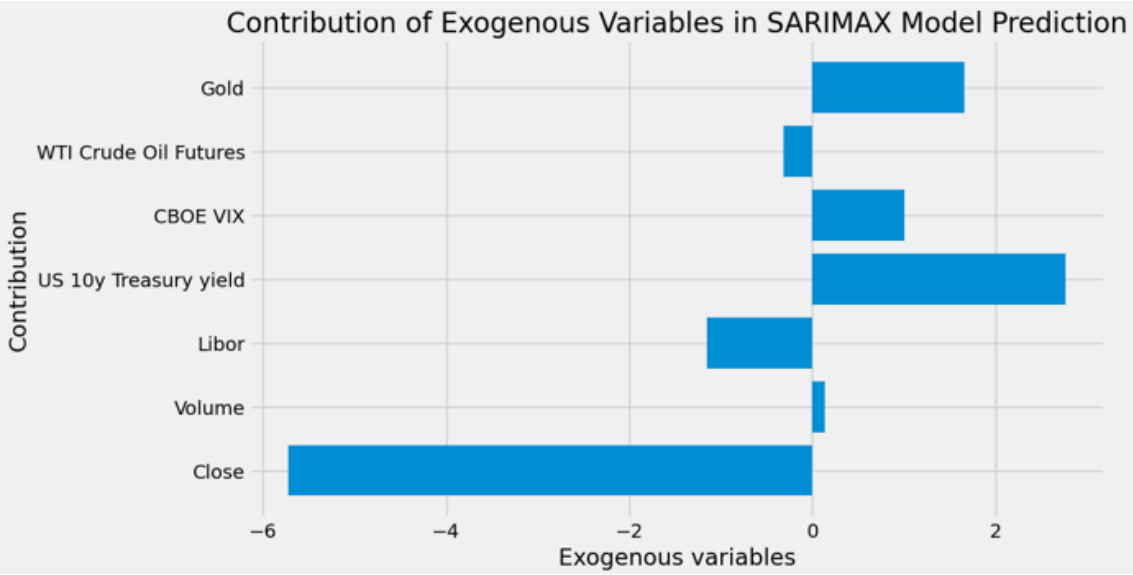


Figure 5: Feature Importance of Exogenous variables

SARIMAX Results

Dep. Variable:	Volatility	No. Observations:	2500
Model:	SARIMAX(0, 1, 0)x(2, 0, 0, 20)	Log Likelihood	-3749.383
Date:	Sat, 15 Apr 2023	AIC	7518.766
Time:	17:05:12	BIC	7577.003
Sample:	0 - 2500	HQIC	7539.910
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
Close	0.0001	0.000	0.605	0.545	-0.000	0.000
Volume	1.143e-09	3.18e-10	3.599	0.000	5.21e-10	1.77e-09
Libor	-1.6465	4.53e-06	-3.63e+05	0.000	-1.646	-1.646
US 10y Treasury yield	1.7692	4.51e-06	3.92e+05	0.000	1.769	1.769
CBOE VIX	0.0010	0.000	7.196	0.000	0.001	0.001
WTI Crude Oil Futures	7.553e-05	0.000	0.307	0.759	-0.000	0.001
Gold	0.0002	0.001	0.153	0.878	-0.003	0.003
ar.S.L20	0.0415	0.000	96.599	0.000	0.041	0.042
ar.S.L40	-0.0375	0.000	-230.306	0.000	-0.038	-0.037
sigma2	1.1568	0.004	292.695	0.000	1.149	1.165

Ljung-Box (L1) (Q):	0.03	Jarque-Bera (JB):	3691355.09
Prob(Q):	0.87	Prob(JB):	0.00
Heteroskedasticity (H):	0.16	Skew:	-2.76
Prob(H) (two-sided):	0.00	Kurtosis:	191.20

	0	4
0		P>|z|
1	Close	0.000
2	Volume	0.000
3	Libor	0.000
4	US 10y Treasury yield	0.000
5	CBOE VIX	0.000
6	ar.S.L20	0.000
7	ar.S.L40	0.000
8	sigma2	0.000

Figure 6: (Left: With all variables), (Right: Removing 2 insignificant variables)

5 Why Machine Learning?

Machine learning has become increasingly popular due to its flexibility and adaptability to complex patterns. It does not require explicit assumptions about the data distribution and can handle large datasets with a high number of variables. Moreover, machine learning models can capture temporal dependencies and long-term patterns, utilizing recurrent neural networks (RNNs) and convolutional neural networks (CNNs). However, there are some limitations to machine learning, including interpretability issues, large data requirements, and sensitivity to noise and outliers due to the absence of assumptions such as stationarity, linearity, and normality.

6 Data Processing

Time series dataset is transformed into supervised learning using a sliding-window representation as shown in the figure. We have to take care that model must be trained on the past and predict the future. So, the training set is the part and test set is in the feature. There are three parameters

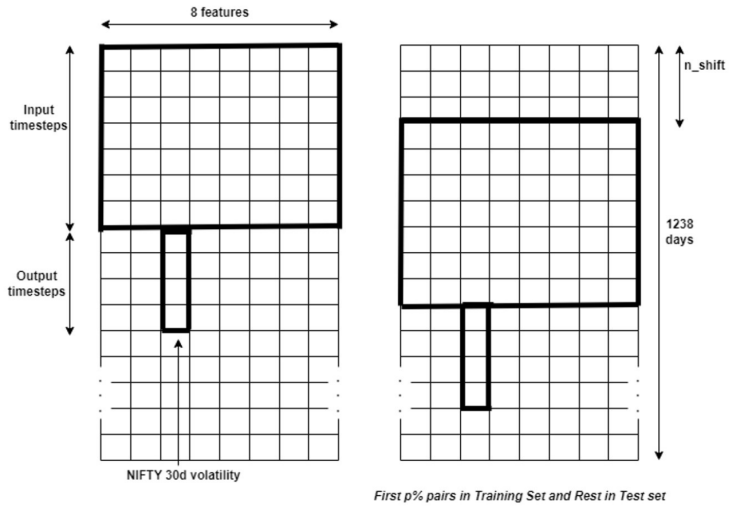


Figure 7: Time series dataset

used to make dataser which are Input Timesteps (number of days we are willing to go in past and use that information for prediction), Output Timesteps (number of days in the future we want to predict volatility for), and N-Shift (the shift of days between two consecutive input-output pairs). Later in the project, these parameters are also varied and the trend are noted.

7 Random Forest

Random Forest is a machine learning algorithm that is commonly used for financial time series prediction. In a financial time series, the goal is to predict the future value of a financial variable, such as stock prices, based on historical data. Random Forest works by creating a large number of decision trees and then combining the predictions of each tree to obtain a final prediction. Each decision tree is trained on a random subset of the available data, and at each split in the tree, only a subset of the available features are considered. This helps to reduce overfitting and improve the generalization performance of the model. Random Forest can be used for both regression and classification tasks, depending on the nature of the problem. For financial time series prediction, it is typically used for regression tasks to predict the continuous values of the financial variable.

7.1 Parameters and Results

Input timesteps = 125, Output timesteps = 1, Nshift = 1, Total number of features = 125 8 = 1000, Train:Test = 85:15

We evaluated each model using two metrics: mean squared error (MSE) and R2 score. MSE is a measure of the average squared difference between the predicted and actual values, while R2 score

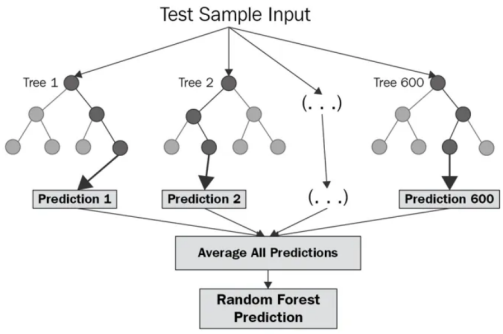


Figure 8: Source: Random Forrest Prediction [3]

is a measure of how well the model fits the data compared to a simple baseline model. Mean square error of the output is found to be 31.3e-5, and R2 square is 0.9256.

7.2 Impurity-based Feature Importance

The feature importance in Random Forest is calculated using the average impurity method[2], which is based on the concept of Gini impurity. Gini impurity is a measure of the impurity or randomness of a node in a decision tree. The feature importance is calculated as the decrease in Gini impurity that results from splitting a node on a particular feature. When training a Random Forest model, the algorithm creates a large number of decision trees, each of which is trained on a random subset of the available data and a random subset of the available features.

To calculate the feature importance, the algorithm examines each decision tree and calculates the average decrease in Gini impurity for each feature over all the decision trees.The feature importance values range from 0 to 1, with higher values indicating greater importance. The feature with the highest importance is considered the most important feature for predicting the target variable.

8 Neural Networks

Artificial neural networks (ANNs) are a type of machine learning algorithm that can be used for time series prediction. ANNs are particularly well-suited for this task because they can learn complex non-linear relationships between input features and the target variable, making them useful for modeling time-dependent processes.

Some authors suggest that ANN can adequately adjust both seasonality and the linear trend of a time series, based on the fact that ANN are capable of modelling any arbitrary function *(Franses Draisma, 1995)*. Other authors claim that despite being universal function approximators, ANN can benefit from the previous elimination of systematic components, thereby focusing on learning the most complex aspects of the series *(Nelson, Hill, Remus, O'Connor, 1999)*

8.1 Parameters and Results

- Train:Test = 85:15

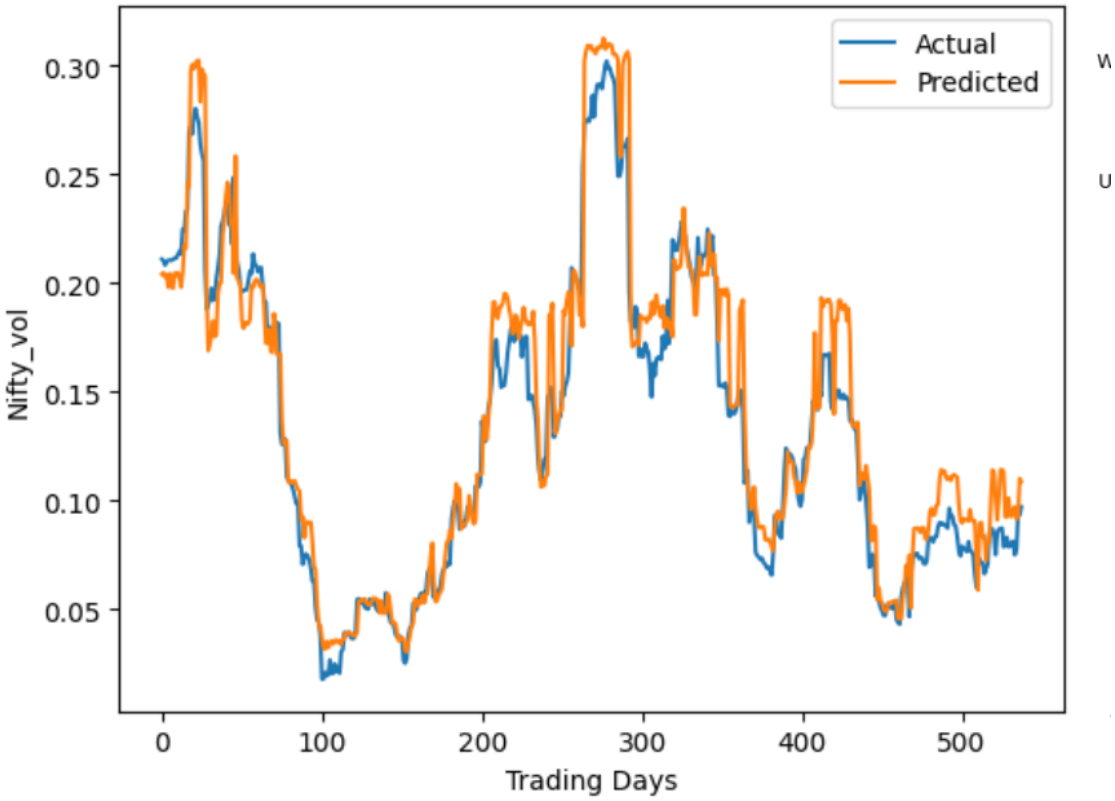


Figure 9: Prediction using Random Forrest

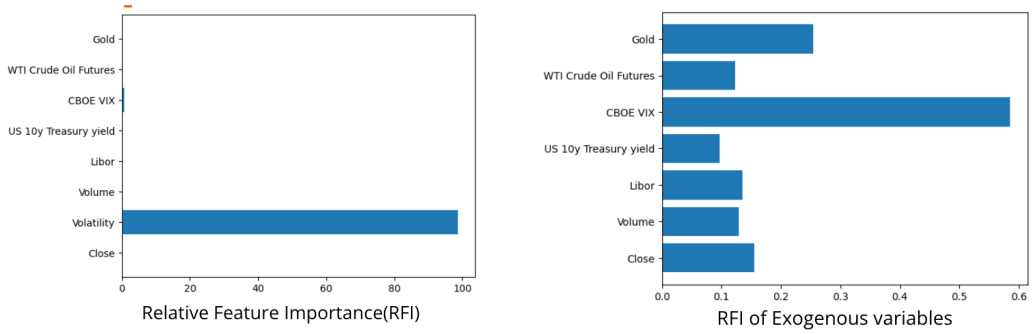
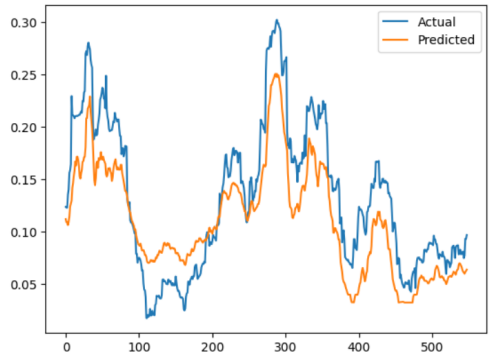


Figure 10: Results of Average-Impurity-based Feature Importance

- 128 input days
- Output Timesteps = 1
- N Shift = 1
- MSE = 151e-5



Layer (type)	Output Shape	Param #
dense (Dense)	(None, 32)	15904
dense_1 (Dense)	(None, 16)	528
dense_2 (Dense)	(None, 8)	136
dense_3 (Dense)	(None, 4)	36
dense_4 (Dense)	(None, 1)	5

Total params: 16,609
Trainable params: 16,609
Non-trainable params: 0

Figure 11: Left: Prediciton results using ANNs (x-axis are days, and y-axis is volatility) Right: Model Specifications

- R2 score: 0.6896

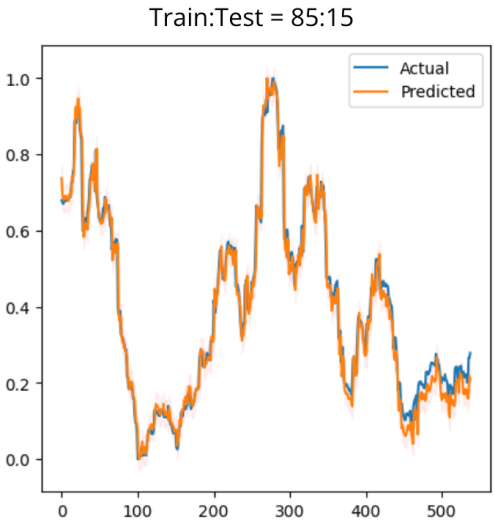
9 RNN-LSTMs

Recurrent neural networks (RNNs) are a type of artificial neural network that are commonly used for processing sequential data such as time-series. The key feature of RNNs is that they have a feedback loop that allows information to be passed from one time-step to the next, making them well-suited for modeling time-dependent processes. For the time-series forecasting task, we can think of your RNN model as a type of "memory machine" that is able to remember the previous values in the time-series and use them to make predictions about future values. This is a powerful capability that allows the RNN model to capture complex patterns and relationships in the data that may be difficult to detect using traditional statistical methods.

LSTMs, these are a type of RNN that are designed to address some of the limitations of traditional RNNs, such as the "vanishing gradient" problem that can occur when training deep networks. LSTMs use a more sophisticated architecture that includes a "memory cell" that can selectively remember or forget previous values in the time-series, making them well-suited for long-term dependencies.

LSTM+Attention model takes this one step further by adding an attention mechanism that allows the model to focus its attention on specific parts of the time-series that are most relevant for making predictions. The attention weights on rows select those variables that are helpful for forecasting. Since the context vector vt is now the weighted sum of the row vectors containing the information across multiple time steps, it captures temporal information *(Shun-Yao Shih et al.)* This is a useful capability when dealing with complex time-series data where different parts of the time-series may have different levels of importance.

BiLSTM+Attention model is a bidirectional LSTM that is able to process the time-series both forwards and backwards. This allows the model to capture information from both past and future time-steps, making it well-suited for tasks such as predicting trend changes or sudden spikes in volatility.



128 days, Output_Timesteps = 1, N_Shift = 1

Model	MSE	R2 Score
RNN	.00226	.9599
LSTMs	.00141	.9779
LSTM + Attention	.00432	.9333
BiLSTM + Attention	.00126	.9794

Figure 12: (Left: Prediction using LSTMs), (Right: Individual Performances of the four models)

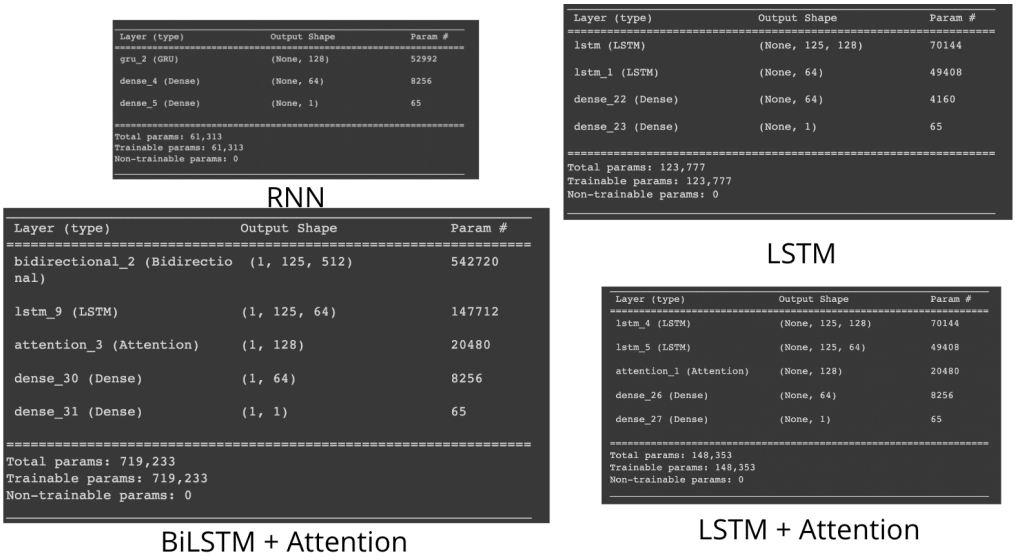


Figure 13: (Left: Prediction using LSTMs), (Right: Individual Performances of the four models)

9.1 Other Parameter Trends

9.2 Permutation Feature Importance

Permutation feature importance[4] is a popular method for determining the importance of features in a machine learning model. This method involves measuring the impact of permuting or shuffling the values of a feature on the model's performance. The idea is that if a feature is important, shuffling its values should significantly reduce the model's performance, while shuffling an unimportant feature should have little effect.

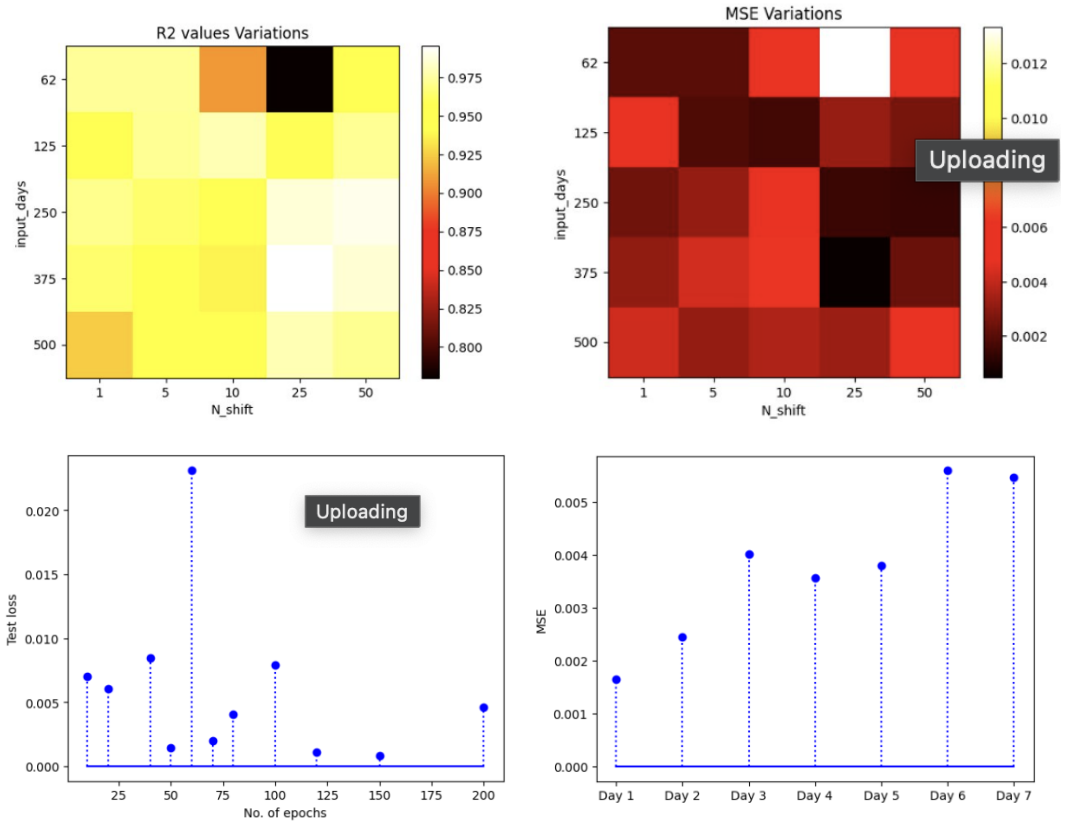


Figure 14: (Above: R2 value and MSE values on varying input-days and N-shift scales), (Below-Left: Test loss on varying number of training epochs), and (Below-Right: Individual MSEs from Day1 to Day7 when output-timesteps is 7)

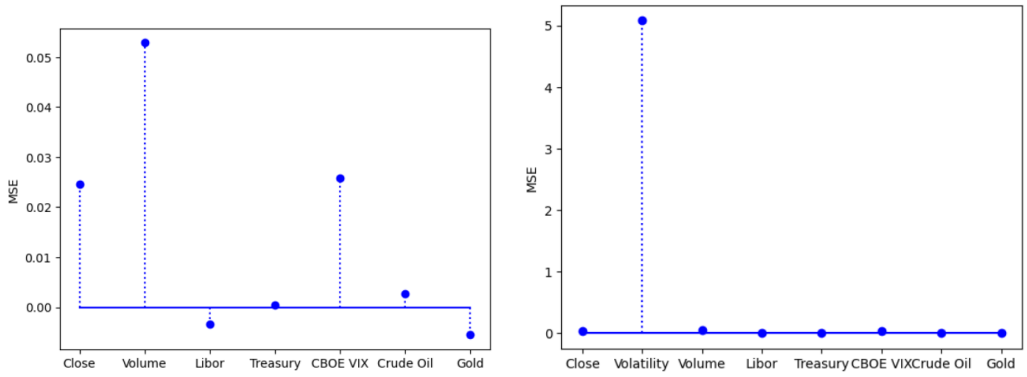


Figure 15: (Left: Relative Feature Importance of Exogenous variables), (Right: Feature Importance of all variables)

To calculate permutation feature importance, the values of a single feature in the test set are randomly permuted and then the model is used to predict the target variable using the permuted

data. The resulting reduction in the model's performance is then used as an indication of the feature's importance. This process is repeated for all features in the model to determine their relative importance.

Permutation feature importance is a powerful technique because it provides a model-agnostic way to evaluate feature importance. This means that it can be applied to any machine learning model regardless of the underlying algorithm. It also has the advantage of being computationally efficient and easy to interpret.

9.3 Exclusion-based Feature Importance

Finding Feature Importance through Exclusion [5] is a method for estimating the importance of a feature in a machine learning model. This method involves training the model multiple times with different features excluded, and then measuring the impact of each exclusion on the model's performance.

To use this method, one can start by training the model with all features included. Then, the model is trained again with one feature excluded, and the resulting loss is compared to the loss from the original model. The larger the increase in loss when a feature is excluded, the more important that feature is considered to be.

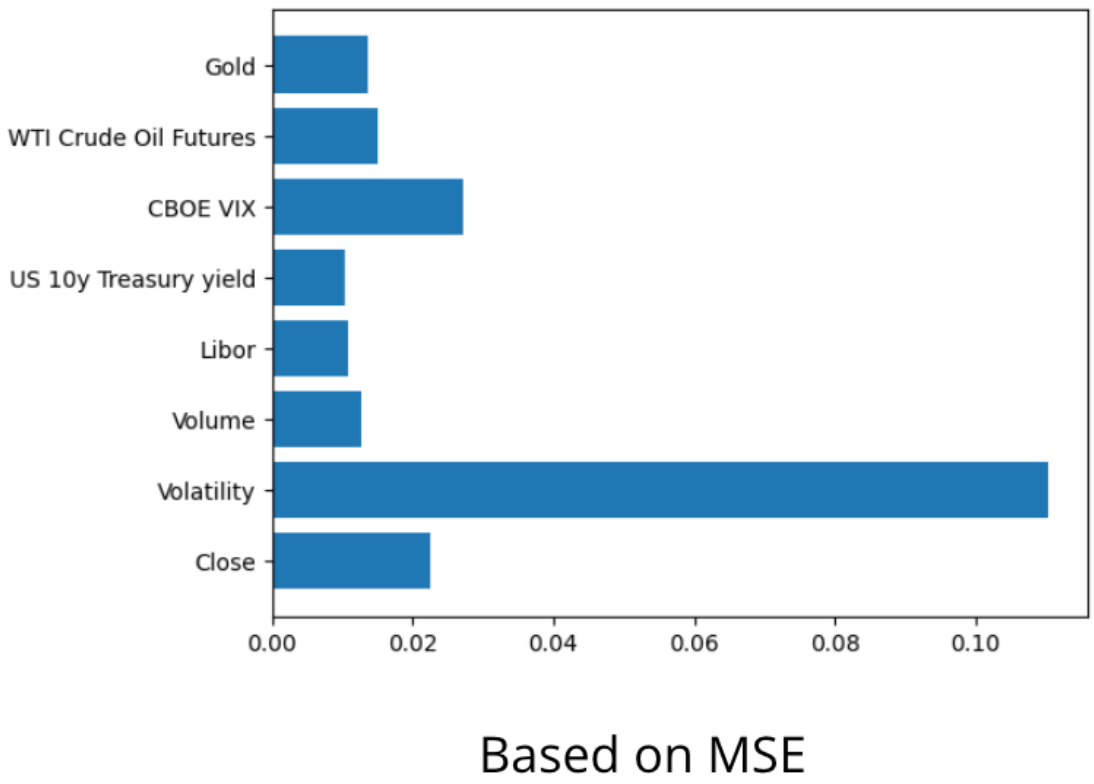


Figure 16: Feature Importance through Exclusion

This process is repeated for each feature in the model, allowing the researcher to rank the features

in order of importance based on the magnitude of the increase in loss when each feature is excluded.

The advantage of this method is that it is model-agnostic and can be applied to any type of machine learning model. It also provides a direct measure of feature importance, allowing for easy interpretation of the results.

10 Conclusion

In this paper, we have examined the effectiveness of various advanced machine learning models in forecasting the volatility of the Indian stock market, using the Nifty30d volatility index as a forward-looking measure. We compared the performance of Random Forest, Artificial Neural Network, Recurrent Neural Network, and Long Short-Term Memory models with the benchmark ARIMA model. Our results show that the advanced machine learning models outperform the ARIMA model in terms of forecasting accuracy, with LSTM performing the best.

We also included several exogenous variables with potential economic significance, including Nifty close prices, Nifty volume, CBOE VIX, US 10-year Treasury yield, LIBOR, WTI crude oil futures, and gold. The inclusion of these variables in the SARIMAX model helped us gain insight into how different macroeconomic factors might affect volatility in emerging markets.

Our findings suggest that advanced machine learning models can be powerful tools for volatility forecasting in the Indian stock market. The inclusion of exogenous variables in the modeling process can further enhance the accuracy of the forecasts and give us better control of the model.

Overall, our study provides useful insights for traders and investors who are interested in volatility trading strategies in the Indian stock market. Our results demonstrate the potential for machine learning models to improve volatility forecasting and provide more accurate predictions of market conditions, which can inform investment decisions and ultimately lead to better financial outcomes.

11 References

References

[1] Hosker, James; Djurdjevic, Slobodan; Nguyen, Hieu; and Slater, Robert (2018) "Improving VIX Futures Forecasts using Machine Learning Methods," SMU Data Science Review: Vol. 1: No. 4, Article 6.

[2] L. Breiman, "Random Forests", Machine Learning, 45(1), 5-32, 2001.

[3] https://corporatefinanceinstitute.com/resources/data-science/random-forest/

[4] Tabatabaei Yazdi, S. A., Naseri, N., Rezaei, M. (2020). Predicting Financial Distress and Corporate Failure: A Review from the State-of-the-Art Machine Learning Algorithms. Journal of Risk and Financial Management, 13(5), 104.

[5] Ghosh, S., Sarker, S., Choudhury, T. (2021). Identifying Drivers of Bitcoin Volatility: An Empirical Study Using Machine Learning. Journal of Financial Data Science, 3(1), 31-45.

[6] Hosker, James; Djurdjevic, Slobodan; Nguyen, Hieu; and Slater, Robert (2018) "Improving VIX Futures Forecasts using Machine Learning Methods," SMU Data Science Review: Vol. 1: No. 4, Article 6.

[7] Ahoniemi, Katja. (2008). Modeling and Forecasting Implied Volatility - an Econometric Analysis of the VIX Index. 10.2139/ssrn.1033812

[8] StockViz - Invest Without Emotions - ARMA + GARCH to Predict VIX - by Shyam

[9] Permutation Feature Importance for ML Interpretability by Seth Billiau

[10] Stock Price Volatility Prediction with LSTM Neural Networks, Jason C. Sullivan