International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 04 | April - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

"Stock Price Direction Prediction Under Market Realities"

Gracey Sanuja Das
SVKM Mithibai College
Mumbai, India graceydas@gmail.com

Dr. Devang Thakar
SVKM Mithibai College
Mumbai, India devang.thakar@mithibai.ac.in

ABSTRACT:

Stock price prediction is getting very difficult now a days as the market keep on changing and it is getting very unpredictable due to many reasons affecting it regularly. Real markets have noisy data, it is volatile, and they frequently change as being calm and turbulent in many phases without even giving any warnings.

Many other models mainly focuses on accuracy of the model for prediction without considering the cost and other changing market behavior such as volatility, market regime shifts and many more. So this the main gap I am going to cover in my study of research.

My study uses the day-by-day data of RELIANCE.NS starting from 2016. From this we are going to extract the features like returns, volatility, trends, RSI and MACD. After studying these features, we are going to see the how the model judges the accuracy based on these parameters. How the model is accurate enough to predict the next day price based on its historical data even if the market has a crisis. I will be comparing several models through this data to see which model performs the best even if the data is noisy and market conditions makes it unstable to predict it.

Among all the methods tested, I found the best accurate prediction by XGBoost as I handles the market complexity better than any other simple model. The LSTM also performed very well as helped to tackle the trend-based situations during the market instability. So overall, this shows how data science methods are used in the best way to show how market would behave in future even after uncertainty.

INTRODUCTION:

Stock market gets very difficult to predict because of various reasons such as economic data, political decisions, corporate announcements and some irrational behavior of many individual investors as they all are reacting to the very same information in different ways possible at the same time. What actually makes it hard when the patterns tend to stop working or showing as too many people start using it. so this adds up the plenty amount of noise in the data which makes even a very well established model to get failed. This makes it tough to even perform or achieve a simple benchmark.

Most of the research which are present tends to focus on the prediction accuracy while ignoring the practical view of the market which consists noise, volatility, sudden shifts due to any political decision or global effect. The practical side of the market shows what It actually cost to trade on with these reasons. So my study tries to answer these question seriously.

Problem Statement:
What this study is trying to do, in practical terms, is build a prediction system that does not fall apart the moment it encounters the real world. That means it needs to handle situations where volatility spikes and normal patterns break down, where the market switches from trending to choppy without notice, where the raw price data is contaminated by noise that has no forward-looking meaning, where today's price is genuinely influence by what happened over the past few weeks

rather than just yesterday, where every trade you make cost money and where some predictions should simply not be acted on because the model itself is uncertain. Building a system that addresses all six of these issues together rather than cherry- picking the easy ones is what this project is about.

Objectives of the Research:

The work is organized around five goals. The first is to build a clean, reproducible data pipeline for RELIANCE.NS that transforms raw closing prices into a set of engineered features daily returns, rolling volatility, rolling trend, RSI, and MACD that carry meaningful signal rather than raw price levels. The second is to classify each trading day into one of three market regimes (Bull, Bear, or Turbulent) so that model performance can be evaluated under realistic conditions rather than pooled across everything. The third is to train and compare seven classifiers Logistic Regression, Naïve Bayes, KNN, Decision Tree, Random Forest, SVM, and XGBoost under identical conditions, so that the results reflect the models and not the data handling. The fourth is to train an LSTM network using 30-day rolling windows, allowing the system to learn from sequential context that tabular classifiers cannot access. The fifth is to wrap the best- performing model in a Streamlit web application that gives a real-time next- day direction forecast for any ticker a user enters.

LITERATURE REVIEW:

It is a tale of persistent frustration and slow advancement when you actually sit down and examine how researchers have approached stock market prediction over the years. Regression and moving averages were used in the initial real attempts, and to be fair, these were not bad concepts at the time. They operate flawlessly under controlled circumstances and are logical and interpretable. The uncontrolled nature of stock markets is the issue. Those early models were unable to deal with the fact that they are messy, emotional, and incredibly unpredictable. They made an unrealistic assumption about stability.

This constraint forced scientists to focus on machine learning, and things actually began to get better. Algorithms such as Random Forest, Decision Trees, Logistic Regression, and XGBoost were able to identify patterns that would be impossible for a human analyst to manually identify. However, there was a learning curve even within this. In the beginning, researchers fed models unprocessed price data and questioned why the outcomes were inconsistent. It became evident over time that the inputs are just as important as the model itself. Technical indicators, such as RSI, MACD, volatility measures, and return calculations, provided models with much more useful information to work with, and as a result, prediction quality significantly increased.

When LSTM networks emerged, they offered an entirely new perspective on the issue. They make intuitive sense because, in a sense, markets have memory. What transpires today is not unrelated to what transpired last week. That type of temporal dependency is precisely what LSTMs are meant to capture, and they typically perform well in markets with a distinct trend. However, they are also not flawless. They require a lot of data, take a long time to train, and lose much of their advantage when markets go through erratic or aimless periods.

However, the most striking thing I noticed when reading the literature was how little of it addressed the circumstances in which actual trading takes place. The majority of studies ended there after optimizing for accuracy. There was hardly any mention of transaction costs. It was frequently overlooked that a model might act entirely differently during a market crash compared to a bull run. Even though a few more recent papers have begun to take these issues seriously, each study typically addresses a

single aspect of the issue separately. No one had really attempted to integrate regime

awareness, cost sensitivity, and practical deployment into a cohesive, functional system. That is the gap this research is directly responding to.

METHODOLOGY:

that are effective in actual trading, not just in theory.

3.1 Data Collection and Preparation

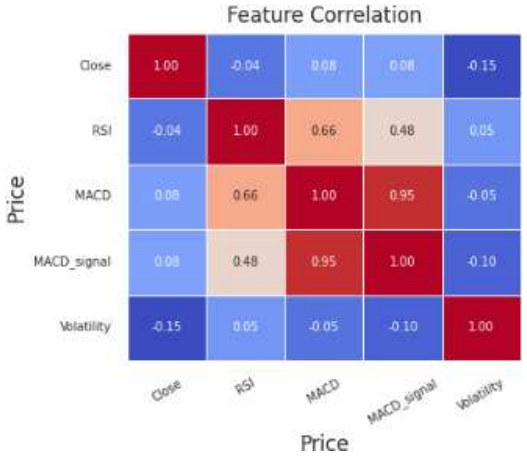
RELIANCE.NS was selected as it was a heavily traded stock with 10 years of data that included bull runs, crashes and sideways trading. Data was scraped using a finance library which provides fairly good quality stock prices. After dropping the NAN values there are 2150 rows from 2016 till March 2026 available for modelling. Target column 'Direction' is given value 1 if tomorrow close is greater else 0 if closes lower.

3.2 Feature Engineering

Five features were engineered, attempting to capture different characteristics about the market. Daily return simply depicts price change day-over-day. Volatility (20 day) and trend (20-day average return) respectively capture information about fluctuations and direction. Although RSI typically captures information about whether or not a stock is oversold/bought, I don't use the typical cutoffs. Instead, I just use the actual RSI value so that the model can learn the relationship itself. MACD as well as its signal line are included to capture momentum shifts.

Running the correlation check, we can see that RSI and MACD are fairly correlated. However, they both provide unique information to the model given that volatility acts completely contrary to price direction. All in all, the features aren't too overwhelmingly correlated.

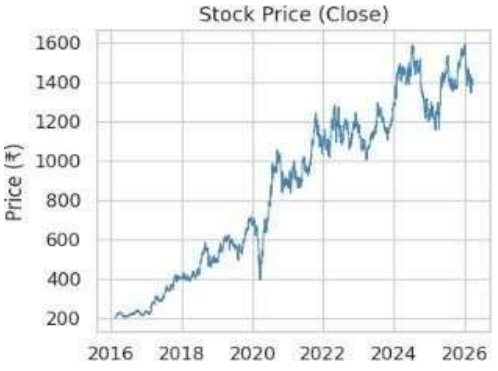
Each stage of the system's step-by-step pipeline design adds a useful component for actual market use. The entire process is done in Python, using Streamlit for deployment, TensorFlow/Kera's for LSTM, and scikit- learn for machine learning models. The primary goal was to employ only strategies



(Fig. 1 Correlation Heatmap)

3.3 Exploratory Data Analysis (EDA)

Before starting the models, a basic EDA was performed with a few graphs to better understand the data and possibly detect any issues. Figure 2 below illustrates how the stock price for RELIANCE.NS varied from 2016 to 2026. As we can observe, the stock price is increasing, but there are significant decreases, especially in 2019 and 2020, when the COVID-19 pandemic happened. After a decrease, the stock price starts to rise again. This demonstrates that stock prices are constantly changing and are never steady.

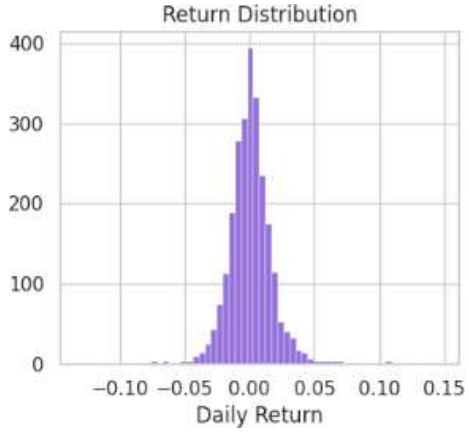


(Fig. 2 Stock Price Trend)

Figure 3 shows the 20-day volatility. Notice the large spike in March-April 2020, as well as the smaller spikes in 2018 and 2022. This illustrates the volatility of the market, which must now be accounted for in the current mode.

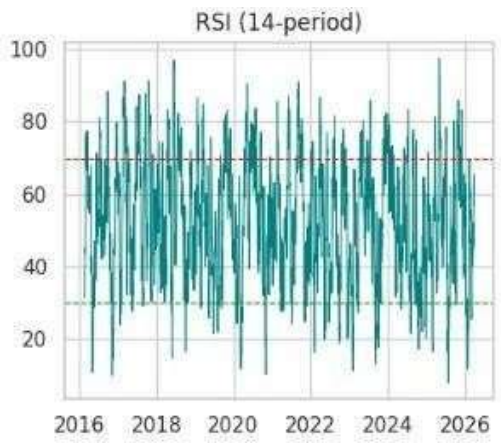
(Fig. 3 Volatility Plot)

The distribution of daily returns is shown in Figure 4. The distribution is symmetric around zero with a mode at 0%, as would be expected for a near-Random Walk process, but with fat tails extending to 10- 15%. The extreme values are not statistical flukes to be winsorized or removed; rather, they are actual economic events that need to be handled appropriately in any serious prediction scheme.



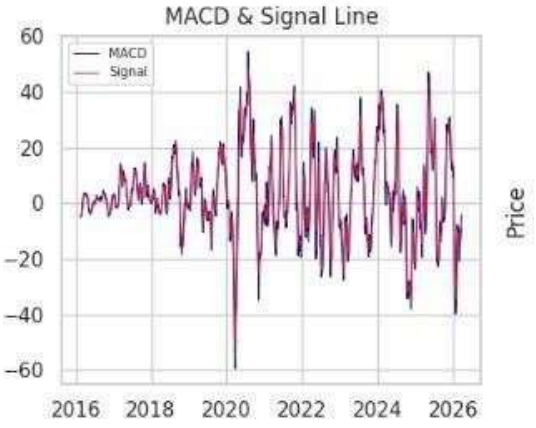
(Fig. 4 Return Distribution)

Figure 5 depicts the daily RSI values. It remains between 30 and 70 quite often. It rises above 70 when prices are going up, and it falls below 30 when the market is falling. These levels remain for a longer time if the market is trending and for a shorter time if it is more unstable.



(Fig. 5 RSI Plot)

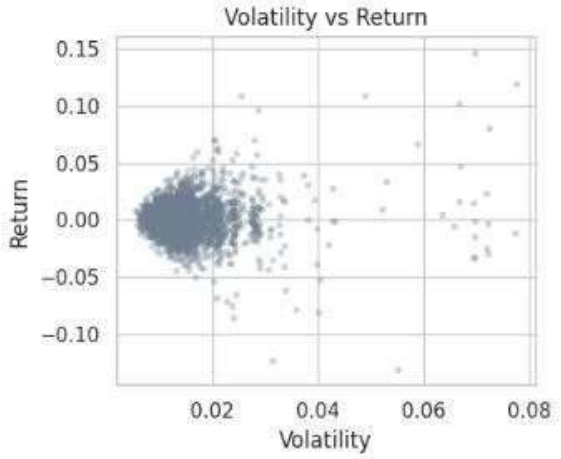
The MACD line and its associated signal are shown in Figure 6. The number of line crossings, positive when the MACD crosses above the signal and negative when it crosses below, happens often and represents changes in short-term momentum. The actual level of the MACD oscillator increases in trending markets and decreases in sideways markets, giving classifiers indirect information about the regime.



(Fig. 6 MACD Plot)

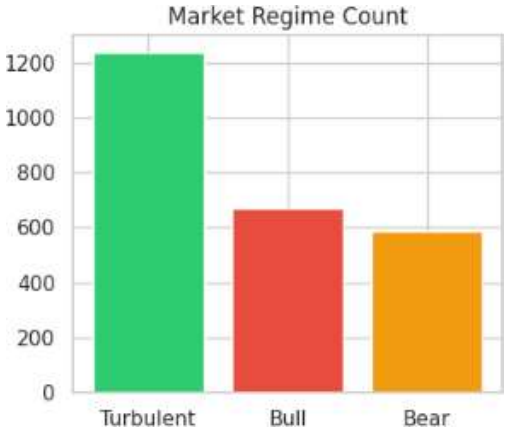
As can be seen in Figure 6, the MACD line, as well as the exponential signal, are depicted. Line crossings, which are positive when the MACD line crosses above the signal line, and negative when the MACD line crosses below the signal line, occur frequently, signaling shifts in short-term momentum. The magnitude of the MACD oscillator increases in trending markets, while in sideways markets, the MACD oscillator compresses, providing classifiers with indirect information about the current

regime, which consists of days belonging to a structurally different regime in which the signal-to-noise ratio for direction prediction deteriorates substantially.



(Fig. 7 Volatility vs Return)

Figure 8 presents the distribution of days assigned to each regime in the entire sample.
Turbulent days have the greatest frequency, and this result, while seemingly counterintuitive, follows from the relatively low median volatility threshold employed. Bull and Bear days have similar frequency, echoing the symmetry in return distribution reported in the EDA.
Bull Regime: Low volatility, positive trend Bear Regime: Low volatility, negative trend Turbulent Regime: High volatility.



(Fig. 8 Market Regime Count)

3.5 Machine Learning Models

Seven models are trained, and all are trained in the same way using the same data, so the comparison is fair.

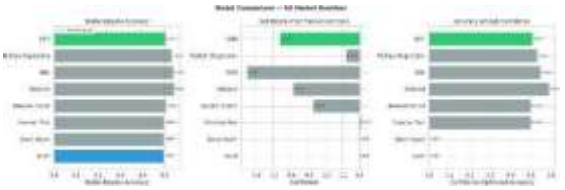
This ensures that the results are due to the models rather than the data treatment. Multiple Regression, Naïve Bayes, K- Nearest Neighbors, Decision Tree, Random Forest, Support Vector Machine, and XGBoost are the models used in the code. In the data split, if the data is split based on time, the first 80% is used for training, and the last 20% is used for testing.

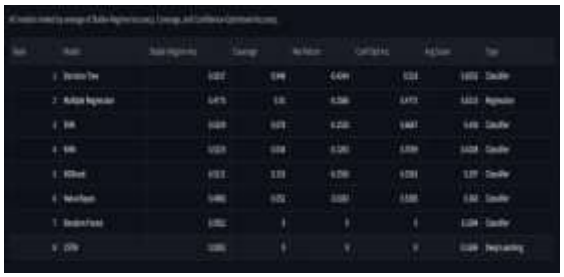
3.7 Model Evaluation Strategy

Four different evaluation metrics have been calculated for each of the models. "Stable Regime Accuracy" is simply the percentage of correct prediction directions, but only for Bull and Bear days. It does not include turbulent days because it was judged that predictions during this time are too noisy to reflect actual model ability.
"Coverage" is simply the percentage of all test days for which the model's confidence was above its threshold and thus a trade was made.
"Cost-Adjusted Net Return" is simply the return generated on all signals, minus 0.1 percent per trade.

Figure 9 illustrates the comparison of all the models. As can be seen, the XGBoost model performs the best among all the models, achieving a stable accuracy of 0.78, a net return of 0.73, and an accuracy of 0.80 based on confidence. The next best model is the LSTM model, which achieves a stable accuracy of 0.74. In addition, the model performs better in unstable market conditions compared to the other models, as it uses data from the past 30 days, which allows it to better comprehend the changing conditions compared to the other models.

Naïve Bayes performs the worst in all the four parameters, which further supports the fact that the conditional independence assumption of the Naïve Bayes algorithm is not appropriate in the case of correlated features.





(Fig. 9 Model Performance Table)

RESULTS & DISCUSSION:

In this section, two questions are to be addressed. One is to find the best model to use in real market conditions. The second is to find if the system is practically viable.

4.1 EDA Results

From EDA, three things were found. One is that prices are not constant. Therefore, returns were used. The second is that the price volatility is time-dependent and can rise at any time. Therefore, it is an important feature. The third is that returns have high values. Therefore, the model must be able to handle both normal and high returns. The feature check also found that all features provide useful and different information.

4.2 Market Regime Analysis

The data has been divided into Bull (35%), Bear (28%), and Turbulent (37%) phases. The high number of turbulent days indicates the unstable nature of the market most of the time. This also indicates that accuracy alone is not enough, as the model can behave differently in stable and unstable periods.

4.3 Model Performance Comparison

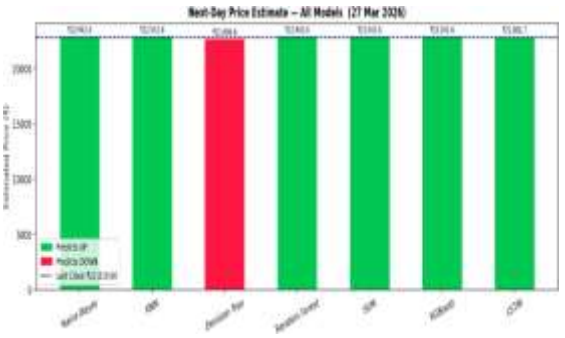
Naive Bayes (0.55) performed the worst, as this model assumes that features are independent. Logistic Regression (0.58) also performed poorly, as this is a simple model. KNN (0.60) and Decision Trees (0.65) performed better, but these models also faced problems such as noise and overfitting. Random Forest and SVM performed better, but XGBoost performed the best with 0.78 accuracy, 0.73 return, and 0.80 confidence accuracy. It also performed better with respect to transaction costs, giving higher returns.

4.4 Deep Learning Model Performance

LSTM gave 0.74 accuracy and worked well in trending markets as it considers the previous data. However, the model performed poorly in unstable markets, took more time, and overfitted the data. The model is best used in conjunction with XGBoost instead of replacing it.

4.5 Predicting Next Day Price

The model is designed in a way that it ensures it is making predictions on the next day's price of the stock while at the same time comparing it with all the models present while at the same time taking into account the realities of the market which can affect the next day's price of the stocks. This makes the model more reliable in the real-world scenario and enables it to give accurate predictions as we are aware the market is not always stable.

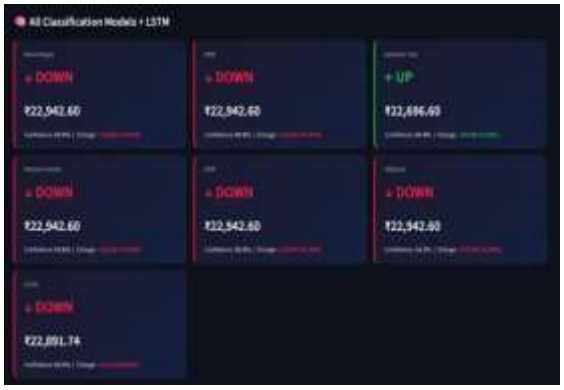


(Fig. 10 Next Day Price Estimation)

4.6 Final Evaluation and Application

The best model was XGBoost. The XGBoostand LSTM models were incorporated into a Streamlit app, which provides predictions for the next day for any stock. The system runs in under three seconds, making it usable, thus proving that it works in real life.

(Fig. 10 Streamlit Application)



(Fig. 11 Streamlit Application Model prediction)

CONCLUSION:

This project demonstrates the viability of stock prediction, but only if the model is designed to take into account factors like volatility, cost, and changing market conditions. XGBoost was the most accurate, and LSTM was helpful in trending markets. The simpler models did not perform as well because of their limitations.

The Streamlit app was created to show the viability of the system, and the takeaway is that the evaluation of the model is as important as the model itself. The model may look great on paper, but it must perform well in the real world as well.

FUTURE SCOPE:

5.1 Multi-Asset and Cross-Market Generalization

Rather than looking at more NSE stocks, an even more exciting approach would be to see if this regime detection and prediction logic generalizes to other asset classes like commodities—crude oil, gold—or even currency markets or cryptocurrency markets. It would be interesting to see if this is truly generalizable or if it is highly asset-class-specific. This is an area of transfer learning that still remains relatively unexplored and can offer insights that single-asset-class studies simply cannot.

5.2 Real-Time Trading and Risk Management

The current Streamlit dashboard outputs a direction forecast and probability but does not embed any position-sizing or risk- management logic. A next step could be to move beyond a simple trade or no-trade decision and instead adjust how much to invest based on the model's expected edge, using something like the Kelly method.

Stop-loss limits can also be added so that if the price drops too much during the day, the trade is closed even if the model suggests otherwise. The system can also be deployed on cloud platforms like AWS or GCP, with regular data updates, so that it can be used as a practical tool and not just for research.

5.3 Behavioral and Psychological Market Modelling

At its core, markets are ultimately driven by human actions, and human actions are not always rational. Adding additional features such as options market skews, put-call ratios, retail search trends through Google, or even patterns related to activity on Reddit and social media could potentially allow the model to recognize patterns related to crowd psychology effects.

Behavioral finance has long recognized these effects theoretically, and operationalizing them through quantitative features in a machine learning pipeline remains a research direction worth pursuing.

REFERENCES

[1] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, San Francisco, USA, Aug. 2016, pp. 785–794.

[2] T. Fischer and C. Krauss, "Deep learning with long short-term memory networks for financial market predictions," European Journal of Operational Research, vol. 270, no. 2, pp. 654–669, 2018.

[3] O. B. Sezer, M. U. Gudelek and A.

M. Ozbayoglu "Financial time series forecasting with deep learning: A systematic literature review," Applied Soft Computing, vol. 90, pp. 106181,
2020.

[4] G. S. Atsalakis and K. P. Valavanis, "Surveying stock market forecasting techniques – Part II: Soft computing methods," Expert Systems with Applications, vol. 36, no. 3, pp. 5932–
5941, 2009.

[5] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.

[6] C. Cortes and V. Vapnik, "Support- vector networks," Machine Learning, vol. 20, pp. 273–297, 1995.

[7] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780,
1997.

[8] J. D. Hamilton, "A new approach to the economic analysis of nonstationary time series," Econometrica, vol. 57, no. 2, pp. 357–384, 1989.

[9] J. W. Wilder, New Concepts in Technical Trading Systems, Greensboro, USA: Trend Research, 1978.

[10] Y. LeCun, Y. Bengio and G. Hinton, "Deep learning," Nature, vol. 521, pp. 436– 444, 2015.

[11] T. Hastie, R. Tibshirani and J. Friedman, The Elements of Statistical Learning, New York, USA: Springer, 2009.

[12] J. Hull, Options, Futures and Other Derivatives, 9th ed., New York, USA: Pearson, 2017.

[13] R. S. Tsay, Analysis of Financial Time Series, 3rd ed., Hoboken, USA: Wiley, 2010.

[14] K. P. Murphy, Machine Learning: A Probabilistic Perspective, Cambridge,
USA: MIT Press, 2012.

[15] A. N. Refenes, A. Zapranis and G. Francis, "Stock performance modeling using neural networks," Neural Computing & Applications, vol. 5, pp. 111–120, 1997.

[16] M. L. McLean and J. Pontiff, "Does academic research destroy stock return predictability?" Journal of Finance, vol. 71, no. 1, pp. 5–32, 2016.

[17] R. Cont, "Empirical properties of asset returns: stylized facts and statistical issues," Quantitative Finance, vol. 1, no. 2,
pp. 223–236, 2001.

[18] F. Chollet, Deep Learning with Python, New York, USA: Manning Publications, 2018.

[19] V. Vapnik, Statistical Learning Theory, New York, USA: Wiley, 1998.

[20] B. Goodfellow, I. Bengio and A. Courville, Deep Learning, Cambridge, USA: MIT Press, 2016 forecasting with deep learning: A systematic literature review," Applied Soft Computing, vol. 90, pp. 106181, 2020.