FORECASTING FORTUNES OF STOCK MARKET PREDICTIONS USING MACHINE LEARNING TECHNIQUES

**A PROJECT REPORT**

***Submitted by***

**DURLAB DAS**

Roll Number: CNI0542307000068

***In partial fulfillment for the award of the degree of***

**POST GRADUATE PROGRAM**

IN

**Data Engineering**

**NATIONAL INSTITUTE OF ELECTRONICS AND INFORMANTION TECHNOLOGY: CHENNAI**

**Ministry of Electronic and Information Technology, Government of India**

JULY 2023

APPENDIX 2

**Table of Contents**

[Abstract](#_bookmark0) i

1. [Introduction 1](#_bookmark0)

2. [Literature review](#_bookmark1) 5

* 1. [Progress of stock price prediction](#_bookmark2) 7
  2. [Proposed System 1](#_bookmark3)6

3. [Methodology 1](#_bookmark5)6

* 1. [Data source 17](#_bookmark6)
  2. [Methodology 17](#_bookmark7)
  3. (1) [SVM 1](#_bookmark7)8
  4. [1.1 Polynomial Kernel 1](#_bookmark7)8
  5. [1.2 Radial Basis Function Kernel (RBF)](#_bookmark7) 20
  6. (2) [LSTM](#_bookmark7) 20
  7. (3) Random Forest [21](#_bookmark7)
  8. 3.1 [Random Forest Algorithm For Online Data](#_bookmark7) 22
  9. Mean Absolute Error[(MAE)](#_bookmark7) 23

Mean Square Error(MSE)23

Root Mean Square Error(RMSE)23

4. [Results and discussion 24](#_bookmark8)

* 1. [(1) SVM process 24](#_bookmark9)
  2. [(2) LSTM process 25](#_bookmark10)
  3. [(3) Random Forest process 26](#_bookmark11)

[Table Accuracy Comparison For Models 1](#_bookmark7)8

5. [Conclusion and Implication 30](#_bookmark13)

* 1. [Conclusion 30](#_bookmark14)
  2. [Implication 31](#_bookmark15)

[Reference 32](#_bookmark16)

[Appendix 34](#_bookmark17)

**Abstract**

Accurate Stock market prediction using machine learning has garnered significant attention in recent years due to its potential to enhance investment decision-making and improve financial outcomes. The objective is to develop predictive models that capture the inherent complexity and nonlinearity of financial markets, enabling accurate predictions of future stock prices, trends, and volatility. The research begins with data collection, gathering historical stock market data, financial indicators, news sentiment, and macroeconomic factors. Pre-processing techniques are applied to clean the data, handle missing values, and normalize numerical features. Various machine learning algorithms, including LSTM (long-short term memory), SVM (support vector machines), RF (random forests), are evaluated for their ability to built model. The trained models are rigorously evaluated using testing datasets, employing metrics such as accuracy, mean squared error, and financial indicators specific to stock market prediction. The finally results are compared to get best fitted model for predictions on unseen data, enabling users to gain insights into future market trends, identify investment opportunities, and manage risks effectively.

# Introduction

As a high-risk and high-return market, the stock market has always been closely watched by investors (Daubechies, I. 1992), and stock forecasting has always been a research topic of great concern to researchers. In addition, the stock market is an important part of my country's financial market, it reflects the operation of the national economy, and the operation of the stock market has an important impact on the operation of the national economy. Although the issue of predictability of stocks has always been controversial, the study of stock forecasts still helps us understand the laws of some market changes and development (Fama, E. F., & Blume, M. F. (1966)). With the advancement of science and technology, a large amount of financial data has been retained(Fama, E. F., & French, K. R. (1988)), providing a solid data foundation for the analysis of the stock market; at the same time, the continuous development and updating of algorithms has provided a powerful tool for people to analyze the stock market(Faria, G., & Verona, F. (2018), Ferreira, M. I., & Santa- Clara, P. (2011), Gençay, R., Selçuk, F., & Whitcher, B. (2002)).

As an important part of a country's economy, the stock market provides a

financing and investment environment for the country's companies and investors. Predicting the future performance of the stock market can not only provide investors with investment advice, but also help companies formulate financing plans, thereby promoting the healthy development of the economy (Campbell, J. Y., & Thompson, S.

B. (2008)). At the same time, establishing a stable investment portfolio based on the forecast results combined with the portfolio theory can help investors to further improve their investment returns. Therefore, it is a very meaningful problem for stock market forecasting and investment portfolio method research.

Investors pay more and more attention to the allocation of financial assets. In addition to savings and debt, relatively traditional investment and financial management methods such as securities, stocks, as a new method of asset allocation,

have gradually become a new key for investors. The first stock in human history, the stock started in 1611 and was created in Amsterdam, the Netherlands. The subject of the transaction is the East India Company in the Netherlands, which was established in 1602 (Campbell, J. Y., & Vuolteenaho, T. (2004)).

Investors usually adjust the allocation of investment assets to reduce their own decision-making risks, this makes it very important for investors to predict the price of stocks or other financial assets. It is a challenging problem to accurately predict when and how to allocate asset budgets at that time, because there are many factors that can affect stock prices, such as the company’s asset allocation or operating conditions, the impact of economic and political policies in related industries, and the occurrence of emergencies and currency exchange rates, etc.(Jiang, F., Lee, J. A., Martin, X., & Zhou, G. (2019)). Therefore, many investors have used technology and quantitative methods to try to predict the fluctuation of asset prices. These methods include finding a relatively suitable model from historical market data and at the same time finding the best time for accurate investment decisions. The issue of whether the stock market trend is predictable has been controversial for decades.

With the development of computer science and artificial intelligence, more and more researchers choose machine learning models for prediction, such as support vector machines, lstm models, etc. Because they can handle nonlinear data, especially support vector machines. This model has a non-linear kernel function, so it has been used by the majority of people in the industry for a period of time.

The stock market not only reflects the development of the country's economy,

but also provides a basis for the country to formulate the next economic policy. The research on the stock market has a long history. At present, the representative stock investment theories include: random walk theory, modern portfolio theory, efficient market hypothesis, behavioral finance and evolutionary securities and so on. Among them, many empirical studies show that the efficient market hypothesis does not hold in emerging markets such as China. Therefore, many researchers began to use statistical models. However, because the statistical model itself has many assumptions, and many of these assumptions are not satisfied in practical applications, it has been difficult for statistical models to achieve good results. Subsequently, the wide application of the classic machine learning model has led many scholars to apply the model to stock price forecasting, and at the same time compare the traditional statistical model. The classical machine learning model avoids many assumptions of the statistical model and has efficient nonlinear learning ability, which makes the performance of the model much better than the statistical model in stock price prediction. Subsequently, people began to utilize classical machine learning models to further improve the out-of-sample performance of stock price prediction.

With the development of deep learning neural networks, people have gradually

realized that neural networks can be used as a new predictive method: First, neural networks have low data requirements and do not require strict assumptions; at the same time, it can also choose non-linear activation. The function converts the linear mapping into a nonlinear mapping, and then through the processing of the hidden layer, it further enhances its ability to process nonlinear data. However, the general neural network does not make much use of the time sequence. Each network layer is performing calculations at the same time, ignoring the time sequence of the data. Therefore, the Recurrent Neural Network (RNN) was born. The connection of the same layer completes the task of extracting data sequence. Of course, RNN also has its own shortcomings. For example, if there are too many hidden layers, RNN will not have too much memory for information from a long time ago. Therefore, in 1997, Hochreiter and Schmidhuber proposed a long and short-term memory neural network model and introduced the concept of "gate", which solved this problem well.

With the rapid development of computer information technology, electronic trading is becoming more and more mature, making it possible to process massive high-frequency trading data. Due to the rapid increase in the amount of data that can be processed and the substantial enhancement of computer hardware, deep learning technology stands out from many machine learning models and performs significantly better than classical machine learning models. In recent years, deep learning technology has been successfully applied to fields such as natural language processing, speech recognition and image processing(Addison, P. S. (2002), Avramov, D. (2002), DeMiguel, V., Garlappi, L., Nogales, F. J., & Uppal, R. (2009)). As the core model of deep learning technology, the deep neural network model has attracted the attention of scholars and has become a research hotspot in recent years. Moreover, the model is completely data-driven and does not require preconditions, and at the same time has efficient nonlinear learning capabilities, which makes the model much better than the classical machine learning model. Therefore, some scholars have turned to various neural network models to predict the rise and fall of stock prices, and have achieved good results. At the same time, the model is also widely used in other financial fields, such as quantitative stock selection, algorithmic trading, high-frequency trading, etc. Then, continuing to study how to apply this model to stock market forecasting more efficiently is of great significance to both investors and researchers.

Thus, in this paper, we will introduce a mixed approach, static dataset and live data

to predict stock prices. Stock market forecasting is the act of trying to determine the future value of shares of companies listed on exchanges or other investment targets. The stock market has multi-scale properties. The so-called multi-scale refers to the existence of multiple data at different time intervals in the stock market. For example, taking stock price data as an example, there are not only short-term real-time price data per second, every minute, and hour, but also daily, weekly and even real-time price data. Monthly mid-term average stock price data, these prices may have different patterns of change at different scales. Stock market forecasting research is usually based on stock market data under a certain scale. After analysis, some patterns that recur in the data are extracted under a certain scale, so as to predict the movement trend of the stock market under this scale. Although data at different scales may have different changing laws, there is also a close interaction between them. If the data at different scales can be considered comprehensively, the state of the stock market can be described more accurately, and thus better forecasting the stock market. Traditionally, stock market forecasts need to be given by stock market researchers with deep knowledge and extensive analytical experience. They make predictions on the future development direction of the stock market and the degree of fluctuations based on multi-source heterogeneous data such as foreign exchange, policies, events, and stock prices in the global economic data.

The factors that affect the stock price of stock data come from many aspects, which makes the stock data itself not stable and linear. Although the difference method can be used to make the sequence stationary, the difference operation also causes data loss. With the development of computer science and artificial intelligence, more and more researchers choose machine learning models for prediction, such as support vector machines, perceptron models, etc., because they can deal with nonlinear data, especially support vector machines , the model has a nonlinear kernel function, so it has been used by the majority of people in the industry for a period of time.

# Literature review

Research in finance has explored how stock markets are affected by their multi- source and heterogeneous data on some scales. Multi-source heterogeneous datain the stock market means that the data of the stock market includes data from different sources such as the stock market, the foreign exchange market and even the weather system, as well as the structure of stock prices, trading volumes, and stock news, announcements and social networks. and other unstructured data. In particular, the efficient market hypothesis believes that information from various sources in the stock market will have an impact on the stock market, while behavioral finance believes that financial markets are explained, studied and predicted from the individual behaviors of traders and the motivations that produce such behaviors the trend and extent of price fluctuations. These studies point out that the internal mechanism of the stock market is very complex, similar to Brownian motion. Combining the multi-source heterogeneous data in the stock market can more accurately classify and predict the stock market state. With the vigorous development of the stock market, it continues to generate a large number of multi-source heterogeneous data of various scales. The traditional idea of relying solely on experts to analyze and predict has been difficult to meet the needs of industry development (Guo, H. 2006 ,Haven, E., Liu, X., & Shen, L. 2012). In order to quickly analyze massive stock market data and assist or even completely replace investors in making stock market investment decisions, a large number of researches on stock market forecasting based on information technology have emerged. These studies have also contributed to the rapid development of quantitative funds that rely on automated computer analysis to execute and even make investment decisions entirely on their own(Chen, J., Jiang, F., & Tong, G. 2017).

Obtaining accurate stock price forecasts can more effectively avoid future risks for decision makers; for regulators, it can strengthen the control of the stock market, regulate and guide the stock market in a timely manner, and contribute to the sustainable development of the economy. Development provides firm confidence and strong guarantees.

The so-called stock price forecast is to use various scientific methods to predict the development prospects of the stock market through the regularity of the development of the stock market and its history and status, relying on a large amount of stock market information and accurate statistical survey data (Dangl, T., & Halling, M. 2012). For decades, scholars have explored various forecasting methods. Therefore, reading about relevant research and summarizing and classifying these forecasting methods has certain positive significance for further research.

However, the factors that affect the stock price of stock data come from many aspects, which makes the stock data itself not stable and linear. Although the difference method can be used to make the sequence stationary, the difference operation also causes data loss, which makes the traditional time series model have great limitations in forecasting. With the development of computer science and artificial intelligence, more and more researchers choose machine learning models for prediction, such as support vector machines, perceptron models, etc., because they can deal with nonlinear data, especially support vector machines, the model has a nonlinear kernel function, so it has been used by the majority of people in the industry for a period of time.

Predicting stock prices or other financial asset prices is very important to investors because investors usually reduce their decision-making risk by adjusting the allocation of investment assets. It is a very challenging problem to accurately predict when and how to allocate the asset budget at that time, because there are many factors that can affect the stock price, such as the company's asset allocation or operating conditions, and the impact of economic and political policies in related industries. , the occurrence of emergencies and the exchange rate of currencies, etc. Therefore, many investors have used technical and quantitative methods to try to predict the volatility of asset prices. These methods include finding relatively suitable patterns from historical market data, as well as pinpointing the best time to make investment decisions. The question of whether the stock market is predictable has been debated

for decades, and there is still no conclusion.

## Progress of stock price prediction

The research on stock behavior was first conducted by Bachelier in 1900. He used random walks to express stock price trends. Fama tested that stock price changes are characterized by random walks. Malkiel and Fama studied valid market assumptions in 1970 and found that all new information will be reflected in asset prices immediately without delay. Therefore, changes in future asset prices have nothing to do with past and present information. From their perspective, predicting future asset prices is considered impossible. On the other hand, many studies try to prove effective market hypotheses experimentally, and empirical evidence shows that the stock market can be predictable in some ways. Virtanen and Yliolli used six explanatory variables to estimate the Finnish stock market index, including the lagging index and macroeconomic factors in an econometric model based on ARIMA. Work (Clark, T. E., & West, K. D. 2007) proposed a stock price prediction system based on ARIMA in 2014, which has been tested in the listed stocks originated from the Stock Exchange in New York and the Stock Exchange running from the country Nigeria. Then the ARIMA model is regarded as a high potential model for forecasting short-term series.

Although econometric models mentioned above can easily describe and evaluate the relationship between large number of variables through inference in the view of statistical, however these methods still have owned limitations for time series analysis in domain of finance. Firstly, they assume that the model structure is linear, and they cannot capture the non-linear nature of stock prices. In addition, these models all assume that the data as a constant value, although the actual time series for finance are full of noise and have time-varying oscillation. Because of its ability in nonlinear mapping and induction, it has been widely used. Many experts try to model financial time nonlinear models, such as multi-layer neural networks and support vector machines (SVM) with nonlinear kernel functions. They are differences from traditional economic models. Neural networks lack of a strict model structure and a series of apparent assumptions. As long as there is enough data, it can be modeled. Work from proposed two mixed models to predict, combining ANN with exponential generalized ARIMA, and later predicted the volatility for S&P500 index return for the year 2012. Their calculation results show that the mixed model has lower test errors and its performance is better than the non-mixed single model. Kristjanpoller et al. merged the generalized autoregressive conditional heteroscedasticity model (GARCH) and ANN in 2014, and proposed a prediction model for the volatility in the Latin American market, and showed that this model is superior to the GARCH model (its MSE is smaller). Work(Cochrane, J. H. 2007) from proposed a hybrid model of neural networks, random forest and support vector regression (SVR) in 2015 to predict the Indian stock market. Agarwal and Sastry combined the RNN neural network into two kinds of linearization models with ARMA and exponential smoothing function in 2015, and predicted stock returns. The experimental results show that the predictability has been greatly improved, and the improvement is mainly contributed by the RNN neural network.

For recent studies, LSTM neural networks that are properly built to learn temporal module have been widely used in various tasks of time series analysis. The reason why LSTM is advanced than traditional RNN is that it solves the problem that RNN neural network fails to solve, that is, the problem of gradient explosion and gradient disappearance, and it can learn effectively through storage units and "gates", and is useful for information for long-term memory. Therefore, many experts have used LSTM to conduct a lot of research on financial time series modeling. In experiments, LSTM is superior to support vector machines due to the addition of emotional features, so that the accuracy of predicting the opening price of the next day has been significantly improved (from 78.57% to 87.86%). The work from Dai, Z. F., Dong, X. D., Kang, J., & Hong, L. (2020b) used the textual data from the newspaper at Nikkei as the input of the LSTM neural network, and combined with the time series data in stock market to predict the opening price of 10 selected companies. A trading strategy based on the predicted results is simulated. The experimental results show that the model has a higher profit value than the trained model only with stock data.

When using deep neural networks (DNN) for financial time series analysis,

researchers are more concerned about the problem of overfitting. Within a year, we can collect only about 252 data points per day. DNN has a good representation ability, because they learn the highly complex nonlinear relationship between variables, so the model has a high accuracy on the training set, but this makes the model prone to overfitting. In order to improve the generalization ability of the model, many researchers have conducted research on regularization methods such as L1 and L2 regularization, Dropout, early stopping, and reducing learning rate. These methods can avoid the problem of overfitting. For artificial neural networks, reducing the size of the neural network can also prevent overfitting, but since larger and deeper networks can solve more complex problems, there must be enough data to use deeper and larger networks. Therefore, the data enhancement method becomes a two-pronged method that can reduce the degree of over-fitting and improve the accuracy of generalization at the same time. However, this method is widely used in image processing problems. Unlike image data, data enhancement of financial time series is not a simple matter. Image data enhancement can be achieved through a variety of transformation techniques. For example, transformation-based data expansion methods will distort the original data to generate new composites. Therefore, although data enhancement of financial time series is of great significance in improving the performance and robustness of deep learning models, it has limited academic attention.

If the dimensionality of the input data is reduced and used as input in a neural

network, information loss may occur. An important advantage of deep learning is that it can learn features from the input data itself. However, for RNN neural networks, as the number of network layers continues to increase, the earlier the input data, its influence on the output results will be more and more weakened due to the increase in sequence length, which leads to the RNN neural network Long-term memory of information is weak. Hochreiter and Schmidhuber in their 1997 paper "Long Short- Term Memory", they proposed the LSTM neural network for the RNN neural network's inability to solve the long-order dependence of the time series. They introduced the "gate" in the LSTM neural network. concept. Felix et al. optimized the LSTM neural network in 2000. Considering that the memory storage unit of the neural network will increase with the increase of the sequence length, which may cause the network to collapse, they proposed to add a forgetting gate inside the LSTM neuron. At this point, the prototype of the LSTM neural network has been completed. The LSTM neural network can control the transmission of input data through input

gates, forget gates and output gates, and maintain the independence of the output of the memory storage unit and the result output, so that the sequence can retain important information during transmission and maintain it for a longer period of time memory. Therefore, the application of LSTM neural network in the prediction of financial time series has become more and more extensive.



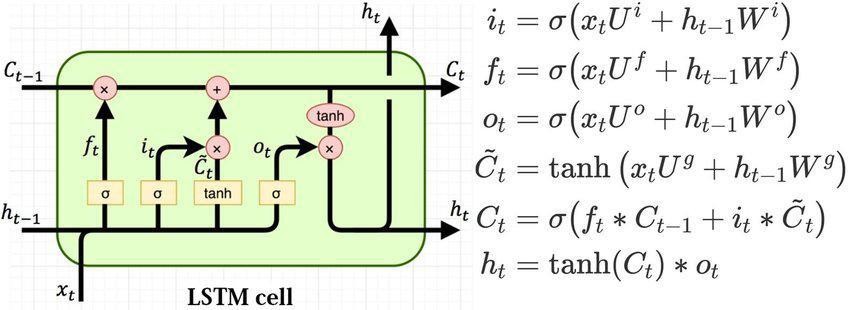


Figure 4 Structure of LSTM

Support Vector Machines (SVMs) have been extensively researched in machine

learning community for the last decade.Yang et al. (2002) used SVM to find the varying

in the margins of SVM regression and change in volatility of financial data. The study

also analyzed the effect of asymmetrical margins to allow for the reduction of the

downside risk. The study found that the former approach produced the lowest error

when predicting the daily closing price of Hong Kong's Hang Seng Index. Kim (2003)

predicted the direction of change in the daily Korean composite Stock Price Index 200

(KOSPI 200) by using SVM and ANN model. The experiment showed that the SVM

outperformed the ANNs model in predicting the future direction of a stock market. The

study reported that the best prediction performance obtained with SVM is 57.8 %

significantly above the 50% threshold. Huang et al. (2004) used SVM for credit rating

analysis and Back Propagation Neural Network (BNN) as a benchmark. The study

obtained prediction accuracy around 80 % for the United States and Taiwan markets by

using both BNN and SVM models. However, SVM model shows only a slight

improvement than BNN model. Shin et al. (2005) demonstrated that SVM performs

better than Back-Propagation Neural Networks when applied to corporate bankruptcy

prediction. The accuracy and performance of SVM model are better than BPN as the

46 training set size is getting smaller. Shah (2007) conducted a study on stock prediction

using various machine learning models and found that the best results achieve with

SVM are 60%. The study confirmed with Kim’s conclusion. Soni and Srivastava (2010)

used Machine-learning algorithms like Classification and Regression Tree (CART),

Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) for

taking investment decisions in the stock market. The results and comparison of all the

models reveal that classification and regression misclassification rate, is only 56.11%

whereas LDA and QDA show 74.26% and 76.57% respectively. The study concludes

that CART algorithm performs better in comparison to LDA and QDA algorithms in

Indian stock market. Yakup et a. (2011) predicted the direction of daily Istanbul Stock

Exchange (ISE) National 100 Index using Artificial Neural Networks (ANN) and

Support Vector Machine (SVM), model. The study indicated that the performance of

ANN model is significantly better than SVM model. Yanshan Wang (2014) predicted

the direction of the Korean stock exchange and Hangseng index by using an integrated

model of Principal component analysis and Support Vector Machine (PCA-SVM). The

results of the study indicated that the proposed model provides distinctly high hit ratios

for predicting the movement of the directions of Korean and Hong Kong stock market.

The Machine Learning Approach and their Applications in finance:

Machine learning methods allow a machine to decide with explicit programming (Avrim and Langley, 1997). Machine learning techniques broadly divided into two

categories – supervised learning and unsupervised learning. A set of training data

supplied to the machine in supervised learning to learn them whereas, in unsupervised

learning, no training data provided. The unsupervised learning algorithm tries to find a

similarity between the data without any given labels (Avrim and Langley, 1997) Vapnik

and co-workers developed support Vector Machine in 1992. It is a supervised machine-

learning algorithm based on statistical learning theory. SVM is a useful method for data

classification and regression analysis. It is also an effective method for pattern

recognition and regression. SVM mostly used in classification problems such as

classification of linear and nonlinear data. SVM creates a boundary and the data points

on either side of the boundary labeled differently (Keerthi et al., 2005). The boundary

in the multidimensional case called a hyperplane. The most critical elements in this

technique are the data points closest to the hyperplane. The optimal hyperplane

maximizes the distance of the hyperplane from the extreme points on either side of the

hyperplane. The optimal hyperplane referred to as the maximum margin hyperplane

(MMH). The hyperplane selection only based on these extreme points. These extreme

data points called the support vectors, and the maximum margin hyperplane known as

the Support Vector Classifier (SVC). Figure-1 depicts the idea of the optimal

47 hyperplane in SVM that separate two classes. In figure-1, support vectors shown as a

rectangle. Support vector regression is different from conventional regression

techniques. In ordinary least squares regression, the optimal line is the one for which

the sum of error is minimized. In SVM, the SVC is the hyperplane for which the sum

of the distance between the hyperplane and the support vector is the maximum. SVM

provides not only linear boundaries but also models nonlinear hyperplane. SVM does

not reduce the empirical risk of making a few mistakes but pretends to build reliable

models with future data. This principle called Structural Risk Minimization (SRM) in

statistical learning theory. SVM uses the Structural Risk Minimization (SRM) but not

the Empirical Risk Minimization (ERM) orientation principle. SVM seeks to minimize

an upper bound of generalization error and not the training error. Hence, it expected to

perform better than conventional techniques. SVM is usually resistant to the over-fitting

problem.

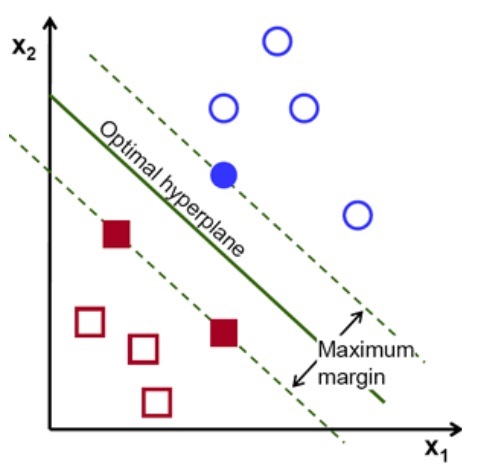


Figure-1 Optimal hyper plane in support vector machine

Random Forest is an Ensemble Supervised Machine Learning technique that has emerged recently. Machine learning techniques have applications in the area of Data mining. Data mining is broadly classified as Descriptive and Predictive. Descriptive data mining concentrates more on describing the data, grouping them into categories, and summarizing the data. Predictive data mining analyzes past data and generates trends or conclusions for future prediction. Predictive data mining has its roots in the classical model building process of statistics. Predictive model building works on the basis of feature analysis of predictor variables. One or more features are considered as predictors. Output is some function of the predictors, which is called hypothesis. The generated hypotheses are tested for their acceptance or rejection. Accuracy of this model is decided by following various error estimation techniques. Usually, descriptive data mining is implemented using unsupervised machine learning techniques, while predictive data mining is carried out using supervised machine learning techniques. Supervised machine learning uses labeled data samples; labels are used to classify samples into different categories. Predictive model learns using training dataset. Test dataset is used to estimate accuracy of the model. Decision tree is commonly used technique for supervised machine learning. Random Forest uses decision tree as base classifier. Random Forest generates multiple decision trees; the randomization is present in two ways:

(1) random sampling of data for bootstrap samples as it is done in bagging

(2) random selection of input features for generating individual base decision trees.

Strength of individual decision tree classifier and correlation among base trees are key issues which decide generalization error of a Random Forest classifier. Accuracy of Random Forest classifier has been found to be at par with existing ensemble techniques like bagging and boosting. As per Breiman [11], Random Forest runs efficiently on large databases, can handle thousands of input variables without variable deletion, gives estimates of important variables, generates an internal unbiased estimate of generalization error as forest growing progresses, has effective method for estimating missing data and maintains accuracy when a large proportion of data are missing, and has methods for balancing class error in class population unbalanced data sets. The inherent parallel nature of Random Forest has led to its parallel implementations using multithreading, multi-core, and parallel architectures. Random Forest is used in many recent classification and prediction applications due to its above mentioned features. In this paper, we have concentrated on the empirical research related to Random forest classifier rather than exploring and analyzing its theoretical background in detail.

**Proposed system**

• Three types of prediction model will be implemented; one using Support Vector Machines, LSTM, and Random Forest.

• The experimental objective will be to compare the forecasting ability of machine learning

algorithms.

• We will test and evaluate both the systems with same test data to find their prediction accuracy.

# Methodology

Of course, stock market prediction technology has great economic value for stock market investors and investment institutions, helping investors and investment institutions to make profits and avoid investment risks. But the value of stock market forecasting technology is far more than that. From a social perspective, stock market forecasting technology can prevent systemic risks in the financial market, help rationally allocate social funds, and contribute to the harmonious and stable development of the economy. Stock data has its own characteristics, and the existing forecasting technology methods are not fully used, so its research brings new challenges to the technology. In particular, the multi-scale and multi-source heterogeneous prediction technology can not only be used for stock market prediction, but also has broad application prospects in many fields such as personal health state prediction, energy demand prediction and website traffic prediction. This research not only has important socio-economic value, but also has important academic research value.

The multi-scale property of stock market data refers to the existence of data at

different time intervals, and the data at different scales will reflect the stock movement state of different time periods. Large-scale stock market data can reflect the long-term movement state of the stock market, and small-scale stock market data can reflect the short-term movement state of the stock market. Data of different scales have associated information and their own unique information. In order to describe the current market state more accurately, it is necessary to comprehensively consider

stock market data of multiple scales. However, most of the existing researches only focus on the single-scale data of the stock market. This can lead to less-than-expected forecast performance due to an inaccurate description of the state of the stock market. How to effectively use multi-scale data is the key to accurately describe and predict

the market state.

* 1. **Data source**

This article selects NIFTY index through NSE (National Stock Exchange), and the transaction data for each trading day from January 2013 to May 2023. The data includes 2600 observations. The selected data are divided into two parts. First part occupied 70% of the selected data to train the model , and the remaining

observations are considered to test and validation.

* 1. **Methodology**

The volatility of stock prices is controlled by the trend of the stock, but is also sensitive to many other factors. Due to the relative stability and predictability of the intrinsic value of stocks, the factors that have impacts on the stock market price mainly include the following aspects: 1. Macro factors; 2. Industrial and regional factors; 3. Company factors; 4. market factors. This article predicts the closing index of the S&P500 rather than specific company stock price forecasts, so aside from the more microscopic industry and company factors, it mainly focuses on the influence of macroeconomic factors and market factors. Macroeconomic factors refer to the impact of macroeconomic environment and its changes on stock prices, including regular factors such as cyclical fluctuations in macroeconomic operations and policy factors such as monetary policy implemented by the government. This article predicts the daily data of the NIFTY closing index, mainly focusing on the impact of monetary policy and other policy factors on stock prices.

There are two types of stock price forecasting methods: qualitative analysis and

quantitative analysis. The qualitative analysis method is the fundamental analysis method, which is a subjective analysis method relying on the experience of financial practitioners. This thesis is a numerical prediction of the daily closing index of the

NIFTY rather than a trend judgment of price fluctuations, so this thesis mainly focuses on the literature review of quantitative analysis methods.

Numerical data-based stock market forecasting research uses numerical data on a certain time scale in the stock market, such as sky-level index prices and stock price volume data, to predict specific stocks or other investments in the stock market on the same scale. Predict the future price of the underlying. According to the focus of the research, these studies can be divided into research on the characteristics of numerical data stock market forecasting and research on the numerical data stock market forecasting model.

In order to build our model, this article will SVM model, LSTM model and Random Forest model. The model in this article uses 70% of the data for training, and the remaining 30% of the data is used for testing. For training, we use Root Mean Square Error.

1. **SVM**

SVM is generally considered to implement a classification approach but I can be used for regression problems too. A hyperplane is generated in multidimensional space to differentiate the classes. To minimize the error, the optimal hyperplane is constructed in an iterative manner. The main feature of SVM is to neatly segregate the corresponding data and the hyperplane with the maximum margin is opted between the support vectors. Basically, the algorithm is generated using a kernel. The kernel trick is a technique where the low dimensional input space is taken and it is converted into a higher dimensional space. The data was not linear separable, so RBF (Radial Basis Function) Kernel was used which to give better results for non-linear kernel. So, firstly SVM is applied to the train dataset and the values are predicted for the test dataset.

## Polynomial kernel:

In general, the polynomial kernel is defined as ;

https://miro.medium.com/v2/resize:fit:321/1*Tt5V_m9iIwXc1xYoDLYmCA.png

b = degree of kernel & a = constant term.

in the polynomial kernel, we simply calculate the dot product by increasing the power of the kernel.

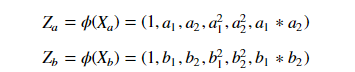
Example:

Let’s say originally X space is 2-dimensional such that

Xa = (a1 ,a2)

Xb = (b1 ,b2)

now if we want to map our data into higher dimension let’s say in Z space which is six-dimensional it may seem like



In order to solve the solve this dual SVM we would require the dot product of (transpose) Za ^t and Zb.

Method 1:

traditionally we would solve this by :

https://miro.medium.com/v2/resize:fit:363/1*otd3-mR4GUY84GrkRg8ZRg.png

which will a lot of time as we would have to performs dot product on each datapoint and then to compute the dot product we may need to do multiplications Imagine doing this for thousand datapoints….

Or else we could simply use

Method 2:

using kernel trick:

https://miro.medium.com/v2/resize:fit:473/1*Ro7cu4K4vuAYGbPv2guivA.png

In this method, we can simply calculate the dot product by increasing the value of power.

## Radial basis function kernel (RBF)/ Gaussian Kernel:

Gaussian RBF(Radial Basis Function) is another popular Kernel method used in SVM models for more. RBF kernel is a function whose value depends on the distance from the origin or from some point. Gaussian Kernel is of the following format;

https://miro.medium.com/v2/resize:fit:336/1*jTU-kuAWMnMMYwBWj8mTVw.png

||X1 — X2 || = Euclidean distance between X1 & X2

Using the distance in the original space we calculate the dot product (similarity) of X1 & X2.

1. **LSTM**

This model chooses the NIFTY return as the only input feature. First, it is necessary to test the stationarity of the closing price series: generally choose to draw a time series diagram first, and check whether the image has an obvious trend; then, we also need to draw a correlation diagram, and through observing whether the acf image is quickly reduced to 0 to judge the stationarity; then perform the ADF test; finally, if the data does not have stationarity, then the difference method is needed to smooth the data.

After smoothing the data, you can construct a single-feature LSTM neural network. This article chooses a three-layer LSTM network, that is, there is only one hidden layer, and the input layer has 20 neurons, so that it can process 20 days of stock prices. Because we are calculating the closing of the next day, so the output layer has only 1 neuron, which is used to output the stock price of the twenty-first day. 20 is chosen because after n-fold cross-checking, 20 is found to be the optimal parameter.

1. **Random Forest**

Random Forest is a classifier consisting of a collection of tree-structured classifiers {h(x, Θk) k=1,2, ….}, where the {Θk } are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input x. Random Forest generates an ensemble of decision trees. To achieve diversity among base decision trees, Breiman selected the randomization approach which works well with bagging or random subspace methods. To generate each single tree in Random Forest Breiman followed following steps: If the number of records in the training set is N, then N records are sampled at random but with replacement, from the original data, this is bootstrap sample. This sample will be the training set for growing the tree. If there are M input variables, a number m << M is selected such that at each node, m variables are selected at random out of M and the best split on these m attributes is used to split the node. The value of m is held constant during forest growing. Each tree is grown to the largest extent possible. There is no pruning. In this way, multiple trees are induced in the forest; the number of trees is pre-decided by the parameter Ntree.

The number of variables (m) selected at each node is also referred to as mtry or k in the literature. The depth of the tree can be controlled by a parameter nodesize (i.e.number of instances in the leaf node) which is usually set to one.

Once the forest is trained or built as explained above, to classify a new instance, it is run across all the trees grown in the forest. Each tree gives classification for the new instance which is recorded as a vote. The votes from all trees are combined and the class for which maximum votes are counted (majority voting) is declared as classification of the new instance. This process is referred to as Forest RI in the literature. Here onwards, Random Forest means the forest of decision trees generated using Forest RI process. In the forest building process, when bootstrap sample set is drawn by sampling with replacement for each tree, about 1/3rd of original instances are left out. This set of instances is called OOB (Out-of-bag) data. Each tree has its own OOB data set which is used for error estimation of individual tree in the forest, called as OOB error estimation. Random Forest algorithm also has in-built facility to compute variable importance and proximities. The proximities are used in replacing missing values and outliers.

Illustrating Accuracy of Random Forest:

The Generalization error (PE\*) of Random Forest is given as,

PE \* = P x,y (mg(X,Y)) < 0

Where mg(X,Y)is Margin function. The Margin function measures the extent to which the average number of votes at (X, Y) for the right class exceeds the average vote for any other class. Here X is the predictor vector and Y is the classification. The Margin function is given as,

mg (X,Y) = avk I(hk (X) = Y) – max j≠Y avk I(hk (X) = j)

Here I(.) is Indicator function. Margin is directly proportional to confidence in the classification.

Strength of Random Forest is given in terms of the expected value of Margin function as,

S = E X, Y (mg (X, Y))

The generalization error of ensemble classifier is bounded above by a function of mean correlation between base classifiers and their average strength (s). If ρ is mean value of correlation, an upper bound for generalization error is given by,

PE\* ≤ ρ (1 – s2) / s2.

**3.1 Random Forest Algorithm for Online Data**

Standard Random Forest algorithm works on off-line data. Many recent applications deal with data streams. Streams are conceptually end-less sequence of data records, real-time, and often arriving at high flow rates. The challenge with streaming data is that there cannot be multiple passes through the data for analysis. Streaming Random Forest is a classification algorithm that combines techniques used to build streaming decision trees with attribute selection techniques of Random Forest. The streaming version of Random Forest achieves classification accuracy comparable to the standard version on artificial and real data sets using only single pass through the data. The limitation is that the algorithm handles only numerical or nominal attributes for which minimum and maximum values of each attribute are known. It also handles multi-class classification problem. Online Random forest algorithm generates on-line decision trees based on concepts from on-line bagging and extremely randomized trees. It also uses Temporal Weighting scheme to discard non performing trees based on their out-of-bag error performance. Incremental Extremely Random forest algorithm is specially designed for small data streams. The algorithm works on the basis of expanding the leaf nodes without reconstructing the whole trees. This approach avoids use of Hoeffding bounds which need large number of samples.

**Mean Absolute Error (MAE)**

Mean Absolute Error (MAE) is the most basic evaluation method, and its expression is as follows:

*MAE* 

**Mean Square Error (MSE)**

1 *N*

*N i*1

*observed*

*i*  *predictedi*

The mean squared error (Mean Squared Error) expression is as follows:

*MSE* 

1 *N*   2

(4)

*N i*1

*observedi predictedi*

The value of MSE is inversely proportional to the accuracy of the model. The larger the MSE, the worse the prediction effect of the model.

**Root Mean Square Error (RMSE)**

Root Mean Square Error (Root Mean Square Error) can be used to calculate the deviation between the observed value and the true value. Because the average index is non-robust, this makes the average error very sensitive to outliers. The expression is as follows:

*RMSE* 

1 *N* 

*N*

*observed*  *predicted*

*i*

*i*



2

*i*1

# Results and discussion

# SVM

# The value response variable is predicted using explanatory or independent factors. The variables that are utilized for prediction are stored in the X dataset. Variables like "Open-Close" and "High-Low" are part of the X. These can be viewed as markers that the algorithm will use to forecast the trend for the upcoming day. Feel free to include more metrics and assess the results. The target dataset y contains the appropriate trade signal, which the machine learning algorithm will try to predict.

## C:\Users\SGG\Desktop\Project Viva\a12.jpg

## (2) LSTM process

In traditional neural networks, neurons in the same hidden layer are not connected to each other, and this structural defect directly leads to their poor performance in dealing with certain problems. This shortcoming becomes especially acute when dealing with time series and speech recognition problems where information is contextualized. The emergence of the recurrent neural network solves this problem very well. The neurons in the same hidden layer are connected to each other, which can effectively obtain the contextual information of the data. The output of the recurrent neural network is determined according to the input and the previous related information, so it can play its short-term memory when dealing with time series problems.

Although the effect of recurrent neural network in dealing with time series problems is very good, there are still some problems. The more serious one is that gradient disappears or explodes easily in the processing of long-term span problems. Causes the phenomenon of small memory value. After the cyclic neural network is expanded, it can be regarded as a multi-layer feedforward neural network with each layer sharing the same weight parameters. Although it keeps trying to learn the long- term dependencies of sequences, actual research finds this to be a difficult task indeed. Long-term reliance on signals tends to become very weak and highly susceptible to short-term signal fluctuations. There is a multiplier of the derivative of the activation function in time-based backpropagation, and the continuous accumulation will cause uncontrollable problems. Although it can be solved theoretically by adjusting the parameters, it is found that this problem is difficult to solve in practice, so it still needs to be optimized from the structure. This leads to its improved structure - the LSTM neural network.

Next, we can start the construction of the LSTM neural network. The first is the determination of several parameters. After n-fold cross-validation, we choose the hidden layer to have 10 neurons. The number of iterations is selected epochs=100 times, and formed into a batch for training, that is, batchsize = 64, Adam algorithm is used as the optimizer of the model, the learning rate is 0.001, and the training set data is randomly scrambled. Use the MSE indicator as the loss function of the model for training. Figure 8 is the training diagram of the neural network. It can be seen from this diagram that after iteration, the loss function of the model decreases quickly and tends to converge. It can be seen that the prediction model is more reasonable.

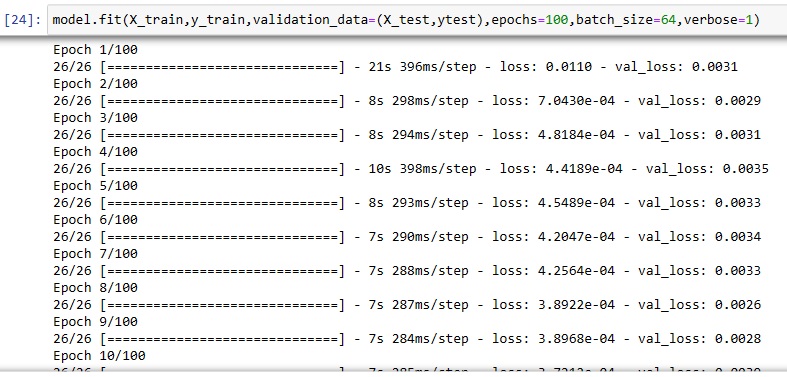


Figure 8 Training for LSTM Model

## 

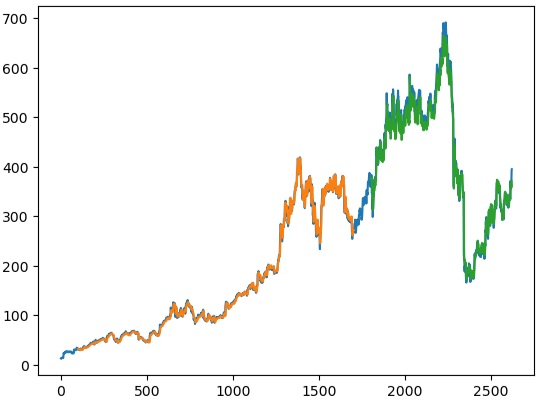
****

Figure 9 The results for Prediction

The code of LSTM can be seen in Apppendix.

This chapter chooses the SVM model, LSTM neural network model, and Random Forest to construct the stock price model and make predictions. First SVM model is used for closing price sequence is stabilized, the model is determined, and the model is checked. Finally, the stock price forecast was made. Next, using the same data, the LSTM neural network with single-feature input and multi-feature input was constructed, the number of layers was selected and the neurons were determined, and when the model parameters were trained to meet the standards, predictions of closing prices were given.

We are going to leverage the model of LSTM into traditional financial time series forecasting, and a stock forecasting model based on long short-term memory neural network (LSTM) is established. The absolute error and the coefficient of determination were evaluated, and a better prediction effect was obtained. It proves the feasibility of deep learning in financial time series forecasting, which can guide the investment behavior of institutions and individuals to a certain extent, and provides new ideas for stock forecasting research.

**(3) Random Forest Process**

**Table1 Accuracy Comparison For Models**

|  |  |
| --- | --- |
| Model | Model Accuracy |
| SVM Polynomial kernel | 63.68% |
| SVM RBF kernel | 89.20% |
| LSTM | 97.44% |
| Random Forest | 99.94% |

Base on Table1, the Model name and Model Accuracy of each model was calculated separately for comparison between the models. It can be seen that the Model Accuracy of the Random Forest is the highest, so it can be clearly seen that the Random Forest is better than the among all models.

# Conclusion and Implication

* 1. **Conclusion**

The importance of the stock market to a country's economy will make the types of stock price forecasting methods continue to develop and grow, and will continue to be derived from the development of other disciplines. In the development process of the follow-up forecasting method, it is necessary to continuously explore and deeply study the characteristics of the stock market, so as to make the model closer to reality, expand the applicability of the method, and obtain better forecasting accuracy.

Because stock data is affected by economic factors, political factors or environmental factors, the law of its change is elusive, and the cycle of the law of change is difficult to determine. Therefore, the model still needs a lot of historical data and selection of appropriate variables for analysis to obtain the desired results. In the SVM model, when analyzing complex stock markets, its prediction results are not particularly ideal, and there are still certain errors in price prediction. As a technology in the field of deep learning, neural network can solve non-linear problems well. LSTM neural network is optimized on traditional neural network and introduces the concept of "gate", which enhances the long-term memory ability of the model, which enhances its generalization ability.

Previously, it was difficult to predict stock prices with high accuracy or inevitability, making investors turn to technical analysts to see what stock prices might be in the future. At present, AI has fulfilled the dream of forecasting stock prices at the annual stock price level. The results of this study revealed that a new technology in the field of indirectly forecasting stocks as knowledge of the most profitable industry areas is very important when buying stocks. Through machine learning algorithms, we were able to predict the world’s top three most profitable industries that were predicted through the use of two types of databases. The paper also showed that the random forest algorithm has the best accuracy because it received a 99.94% accuracy ratio in this work.

**Implication**

The prediction model studied in this article is based on the LSTM neural network, and preliminary results have been obtained. However, due to objective factors such as research time and data sources, there is still a lot of room for research in this article. There is still much to be done for the model constructed in this article. Update and improvement, the follow-up work mainly includes the following aspects:

1. Handling of abnormal values

Because stock market has a certain degree of speculation and is also susceptible to policy influences, there are often skyrocketing and plummeting situations. This leads to outliers in the stock data we obtain. There are many reasons for the occurrence of outlier points in stock data, which cannot be obtained by quantitative analysis. This makes the problem unable to simply use the LSTM neural network constructed in this article. Therefore, some methods to deal with outliers can be used to perform data processing. Noise reduction, such as wavelet transform, Fourier transform, etc.

1. About feature selection

The number of features of the data set obtained in this article is not very large, and for real stock data, in addition to the features in the stock market, can you use other features, such as corporate financial reports; also, because the stock market is affected by policy Larger, using the application of LSTM neural network in text learning and text sentiment analysis, can we obtain some features from news and financial reports, so as to enable the model to make corresponding judgments on stocks in an economic sense, thereby improving the accuracy of our predictions rate. (3)About model optimization

When constructing the neural network model, whether the number of hidden layers is small, and whether more hidden layers will have better prediction results, this is also the lack of research in this article.

**Reference**

Addison, P. S. (2002). The illustrated wavelet transform handbook. Napier University.

Avramov, D. (2002). Stock returns predictability and model uncertainty. Journal of Financial Economics, 64, 423–458.

Brock, W., Lakonishok, J., & LeBaron, B. (1992). Simple technical trading rules and the stochastic properties of stock returns. The Journal of Finance, 47, 1731–1764.

Campbell, J. Y., & Thompson, S. B. (2008). Predicting excess stock returns out of sample: Can anything beat the historical average? Review of Financial Studies, 21,1509–1531.

Campbell, J. Y., & Vuolteenaho, T. (2004). Bad beta, good beta. The American Economic Review, 94, 1249–1275.

Chen, J., Jiang, F., & Tong, G. (2017). Economic policy uncertainty in China and stock market expected returns. Accounting and Finance, 57, 1265–1286.

Clark, T. E., & West, K. D. (2007). Approximately normal tests for equal predictive accuracy in nested models. Journal of Econometrics, 138, 291–311.

Cochrane, J. H. (2007). The dog that did not bark: A defense of returns predictability. Review of Financial Studies, 21, 1533–1575.

Conrad, J., & Kaul, G. (1998). An anatomy of trading strategies. Review of Financial Studies, 11, 489–515.

Cowles, A., 3rd (1933). Can stock market forecasters forecast? Econometrica. Journal of the Econometric Society, 309–324.

Dai, Z., Zhou, H., Wen, F., & He, S. (2020a). Efficient predictability of stock return volatility: The role of stock market implied volatility. The North American Journal of Economics and Finance, 52, 101174.

Dai, Z., & Zhu, H. (2020). Stock returns predictability from a mixed model perspective. Pacific-Basin Finance Journal, 60, 101267.

Dai, Z. F., Dong, X. D., Kang, J., & Hong, L. (2020b). Forecasting stock market returns: New Technical indicators and two-step economic constraint method. The North American Journal of Economics and Finance, 53, 101216.

Dangl, T., & Halling, M. (2012). Predictive regressions with time-varying coefficients. Journal of Financial Economics, 106, 157–181.

Daubechies, I. (1992). Ten lectures on wavelets. Philadelphia, PA: SIAM (Society for Industrial and Applied Mathematics).

DeMiguel, V., Garlappi, L., Nogales, F. J., & Uppal, R. (2009). A generalized approach to portfolio optimization: Improving performance by constraining portfolio norms. Management Science, 55, 798–812.

Fama, E. F., & Blume, M. F. (1966). Filter rules and stock market trading. Journal of Business, 39, 226–241.

Fama, E. F., & French, K. R. (1988). Dividend yields and expected stock returns. Journal of Financial Economics, 22, 3–25.

Faria, G., & Verona, F. (2018). Forecasting stock market returns by summing the frequency-decomposed parts. Journal of Empirical Finance, 45, 228–242.

Ferreira, M. I., & Santa-Clara, P. (2011). Forecasting stock market returns: The sum of the parts is more than the whole. Journal of Financial Economics, 100, 514–537.

Gençay, R., Selçuk, F., & Whitcher, B. (2002). An introduction to wavelets and other filtering methods in finance and economics. Academic Press.

Goyal, A., & Welch, I. (2003). Predicting the equity premium with dividend ratios. Management Science, 49, 639–654.

Goyal, A., & Welch, I. (2008). A comprehensive look at the empirical performance of equity premium prediction. Review of Financial Studies, 21, 1455–1508.

Guo, H. (2006). Time-varying risk premia and the cross section of stock returns. Journal of Banking & Finance, 30, 2087–2107.

Haven, E., Liu, X., & Shen, L. (2012). De-noising option prices with the wavelet method. European Journal of Operational Research, 222(1), 104–112.

Inoue, A., & Kilian, L. (2004). In-sample or out-of-sample tests of predictability: Which one should we use? Economic Review, 23, 371–402.

Jaffard, S., Meyer, Y., & Ryan, R. D. (2001). Wavelets from a historical perspective. Philadelphia, PA: SIAM (Society for Industrial and Applied Mathematics).

Jiang, F., Lee, J. A., Martin, X., & Zhou, G. (2019). Manager sentiment and stock returns. Journal of Financial Economics, 132, 126–149.

**Appendix**

from sklearn.model\_selection import train\_test\_split

from sklearn import preprocessing, svm

from sklearn.linear\_model import LinearRegression

import pandas as pd

import math

import numpy as np

import matplotlib.pyplot as plt

from matplotlib import style

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, LSTM

from sklearn.preprocessing import MinMaxScaler

from sklearn import metrics

import matplotlib.pyplot as plt

from matplotlib.pyplot import figure

from sklearn.metrics import accuracy\_score

df1 = pd.read\_csv("D:/Project/Used dataset/NFLX(2013\_2023)new.csv",parse\_dates = True, index\_col=0)

print(df1.index)

df1.describe()

df1.isna().any()

correlation = df1.corr()

print(correlation["Close"].sort\_values(ascending=False))

df=df1

df = df.dropna()

df

#SVM CODING

df['High\_Low\_per'] = (df['High'] - df['Close']) / df['Close']\*100

df['Per\_change'] = (df['Open'] - df['Open']) / df['Close']\*100

df = df[['Prev\_Close','High\_Low\_per','Per\_change','Volume']]

label\_col = 'Prev\_Close'

forecast\_ceil = int(math.ceil(0.001\*len(df)))

df['label'] = df[label\_col].shift(-forecast\_ceil)

#feaures X, labels Y

X = np.array(df.drop(['label'],1))

X = preprocessing.scale(X)

X = X[:-forecast\_ceil:]

X\_lately = X[-forecast\_ceil:] #no y value

X\_lately

df.dropna(inplace=True)

y = np.array(df['label'])

len(X),len(y)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,y,test\_size=0.2)

clf = svm.SVR(kernel='poly') #svm.SVR()

clf.fit(X\_train, y\_train) #train

accuracy = clf.score(X\_test, y\_test)\*100 #test Accuracy squared error for linreg

print(accuracy) #directionally accurate

clf1 = svm.SVR(kernel='rbf')

clf1.fit(X\_train, y\_train)

accuracy = clf1.score(X\_test, y\_test)

print(accuracy)

forecast\_set = clf.predict(X\_lately) #pass a single value or array

print(forecast\_set, accuracy)#forecast\_out) # stockprices next 30 days

df['Forecast'] = np.nan

figure(num=None, figsize=(40, 20), dpi=160, facecolor='w', edgecolor='k')

df['Prev\_Close'].plot()

#df['Forecast'].plot()

plt.legend(loc=4)

plt.xlabel('Date')

plt.ylabel('Price')

plt.show()

#LSTM CODING

# LSTM are sensitive to scale of the data so we apply MinMax scaler

scaler=MinMaxScaler(feature\_range=(0,1))

df=scaler.fit\_transform(np.array(data1).reshape(-1,1))

df

# Splitting dataset into train & test split

training\_size=int(len(data1)\*0.65)

test\_size=len(data1)-training\_size

train\_data,test\_data=data1[0:training\_size,:],data1[training\_size:len(data1),:1]

# Convert array of values into datset matrix

import numpy

def create\_dataset(dataset,time\_step=1):

dataX,dataY=[],[]

for i in range(len(dataset)-time\_step-1):

a=dataset[i:(i+time\_step),0] ### i=0, 0,1,2,3\_ \_ \_ \_ \_ 99 100

dataX.append(a)

dataY.append(dataset[i+time\_step,0])

return numpy.array(dataX),numpy.array(dataY)

# Reshape into X=t,t+1,t+2,t+3 & Y=t+4

time\_step=100

X\_train,y\_train=create\_dataset(train\_data,time\_step)

X\_test,ytest=create\_dataset(test\_data,time\_step)

print(X\_train.shape),print(y\_train.shape)

print(X\_test.shape),print(ytest.shape)

# Reshape input to be [Sample, time steps, features] which is required for LSTM

X\_train=X\_train.reshape(X\_train.shape[0],X\_train.shape[1],1)

X\_test=X\_test.reshape(X\_test.shape[0],X\_test.shape[1],1)

# Create Stack LSTM model

model=Sequential()

model.add(LSTM(50,return\_sequences=True,input\_shape=(100,1)))

model.add(LSTM(50,return\_sequences=True))

model.add(LSTM(50))

model.add(Dense(1))

model.compile(loss='mean\_squared\_error',optimizer='adam')

model.summary()

X\_train[:100]

model.fit(X\_train,y\_train,validation\_data=(X\_test,ytest),epochs=100,batch\_size=64,verbose=1)

# Lets do the prediction & check performance metrics

train\_predict=model.predict(X\_train)

test\_predict=model.predict(X\_test)

# Transformback to original form

train\_predict=scaler.inverse\_transform(train\_predict)

test\_predict=scaler.inverse\_transform(test\_predict)

# Calculate RMSE performance metrics

import math

from sklearn.metrics import mean\_squared\_error

math.sqrt(mean\_squared\_error(y\_train,train\_predict))/100

# Test Data RSME

math.sqrt(mean\_squared\_error(ytest,test\_predict))/100

# Plotting

# Shift train prediction for plotting

look\_back=100

trainPredictPlot=numpy.empty\_like(data1)

trainPredictPlot[:,:]=np.nan

trainPredictPlot[look\_back:len(train\_predict)+look\_back,:]=train\_predict

# Shift test predictions for plotting

testPredictPlot=numpy.empty\_like(data1)

testPredictPlot[:,:]=np.nan

testPredictPlot[len(train\_predict)+(look\_back\*2)+1:len(data1)-1,:]=test\_predict

# Plot baseline & predictions

plt.plot(scaler.inverse\_transform(data1)) # Complete Dataset=Blue Color

plt.plot(trainPredictPlot) # Train data=Orange

plt.plot(testPredictPlot) # Test data=Green Color

plt.show()

# Calculate accuracy

#print('Model accuracy (%)')

#Y\_p=model.predict(X\_train)

#Y\_t=Y\_train.reshape(Y\_train.shape[0],1)

#print((1-(metrics.mean\_absolute\_error(Y\_t, Y\_p)/Y\_t.mean()))\*100)

print('Model accuracy (%)')

Y\_p=model.predict(X\_train)

Y\_t=y\_train.reshape(y\_train.shape[0],1)

print((1-(metrics.mean\_absolute\_error(Y\_t, Y\_p)/Y\_t.mean()))\*100)

#RANDOM FOREST CODING

X = df[['Open', 'High', 'Low', 'Close', 'Prev\_Close', 'Volume']]

y = df['Close']

# Step 2: Split the data into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Step 3: Build the Random Forest model

rf\_model = RandomForestRegressor(n\_estimators=100, random\_state=42)

# Step 4: Train the model

rf\_model.fit(X\_train, y\_train)

# Step 5: Model evaluation

y\_pred = rf\_model.predict(X\_test)

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print(y\_pred)

print(y\_pred.shape)

error=abs(y\_pred-y\_test)

mape=100\*(error/y\_test)

accuracy=100 -np.mean(mape)

print('Accuracy:', round(accuracy,2), '%.')

print(f"Mean Squared Error (MSE): {mse}")

print(f"R-squared (R^2): {r2}")

pip install nsepy

import datetime as dt

import nsepy as nse

stock\_price =nse.get\_history(symbol="SBIN",index=True,start=dt.date(2023,1,1), end=dt.date(2023,5,23))

stock\_price.dropna()

pip install yfinance

import yfinance as yf

nse50=yf.Ticker("^NSEI")

NSE=nse50.history(period="max")

NSE

NSE.index

NSE.plot.line(y="Close",use\_index=True)

del NSE["Dividends"]

del NSE["Stock Splits"]

NSE["Tomorrow"] =NSE["Close"].shift(-1)

NSE

#creating target based on tomorrow

NSE["Target"]= (NSE["Tomorrow"]>NSE["Close"].astype(int))

NSE

#MACHINE learning model impliment

#from sklearn.ensemble import RandomForestClasifier

model=RandomForestClassifier(n\_estimators=100, min\_samples\_split =100, random\_state=42)

train=NSE.iloc[:-100]

test=NSE.iloc[-100:]

predictors=["Close","Volume","Open","High","Low"]

model.fit(train[predictors], train["Target"])

#market accurace prediction either market go up

from sklearn.metrics import precision\_score

preds =model.predict(test[predictors])

preds=pd.Series(preds, index =test.index)

preds

#preds is predective targets

precision\_score(test["Target"],preds)

combined=pd.concat([test["Target"], preds],axis=1)

combined

def predict(train, test, predictors, model):

model.fit(train[predictors],train["Target"])

preds=model.predict(test[predictors])

preds=pd.Series(preds, index=test.index, name="predictions")

combined=pd.concat([test["Target"],preds], axis=1)

return combined

def backtest(data, model, predictors,start=2500,step=250):

all\_predictions=[]

for i in range(start, data.shape[0],step):

train =data.iloc[0:i].copy()

test=data.iloc[i:(i+step)].copy()

predictions =predict(train, test, predictors, model)

all\_predictions.append(predictions)

return pd.concat(all\_predictions)

predictions = backtest(NSE, model, predictors)

predictions["predictions"].value\_counts()

precision\_score(predictions["Target"], predictions["predictions"])

#adding prediction

horizons =[2,5,60,250,1000]

new\_predictors= []

for horizon in horizons:

rolling\_averages =NSE.rolling(horizon).mean()

ratio\_column = f"Close\_Ratio\_{horizon}"

NSE[ratio\_column]=NSE["Close"]/rolling\_averages["Close"]

trend\_column = f"Trend\_{horizon}"

NSE[trend\_column]=NSE.shift(1).rolling(horizon).sum()["Target"]

new\_predictors += [ratio\_column, trend\_column]

NSE

NSE.dropna()

model = RandomForestClassifier(n\_estimators =200, min\_samples\_split=50,random\_state =1)

def predict(train, test,predictors, model):

model.fit(train[predictors],train["Target"])

preds=model.predict\_proba(test[predictors])[:,1]

preds[preds >= .6] = 1

preds[preds < .6] = 0

preds=pd.Series(preds, index=test.index, name="predictions")

combined=pd.concat([test["Target"],preds], axis=1)

return combined