Risk Analysis for Long-Term Stock Market Trend Prediction

Rounak Bose(✉), Amit Das, Jayanta Poray, and Supratim Bhattacharya

Techno India University,
West Bengal, EM 4/1 Salt Lake, Sector V, Kolkata 700091, India
rb1311997@gmail.com, operon.amit@gmail.com, jayanta.poray@gmail.com,
bhattacharya.supratim@gmail.com

Abstract. Stock market trend analysis is very crucial for the understanding of the way stock market attributes can fluctuate with time. Also it helps investors to analyse when to buy and/or sell financial instruments. Even though the predictions can be made with a certain degree of accuracy, the ultimate aim is to minimise the risk associated with the predictions. In this work, Linear regression, Ridge regression, Bayesian Ridge regression, Lasso regression and FBProphet forecasting models are used and compared to predict stock market prices for a particular dataset with a benchmark accuracy. Also, on the basis of the used forecasting models, we have devised a new risk function for long-term stock market predictions. This risk function is derived from the risk functions proposed by NIST and MEHARI.

Keywords: Stock market trend analysis · Time-series data analysis · Predictive modelling · Regression · Risk functions

1 Introduction

One of the most volatile and fluctuating dynamics of the world is the stock market. A stock market is essentially a huge and complex system in which stocks - shares of companies that indulge in public trading - are issued, then bought and consequently sold, and this process goes on in this cyclic fashion. As much as it is a place where the experienced people pit their expertise against others, the stock market not only takes into account people's sentiment and the public trends in general, but also the technicalities and the economic factors that play a very major role in this domain. A very noteworthy view in case of stock markets is that the stock market is a highly adversarial system of trading [14].

Importance of Stock Market Prediction: Stock market predictions are essentially what is the ruling factor of all the dynamics of every society. Stock market prices actually influence the social political and, needless to say, economic dynamics of the world we live in. Hence predicting stock market values will essentially predict the future. But then, we cannot make predictions with accuracy that is actually useful. Suppose for instance, we get predictions of

M. Singh et al. (Eds.): ICACDS 2019, CCIS 1045, pp. 381–391, 2019.
https://doi.org/10.1007/978-981-13-9939-8_34

accuracy 0.8 and 0.9 respectively. Then what we actually take into account are the mis-predictions of values 0.2 and 0.1 respectively. Hence, since we cannot increase the accuracy, what we can actually do with various regression analysis techniques, is minimise the risk associated with these predictions. So the ultimate aim for stock market predictions is to minimise the risk and consequently increase the accuracy of the predictions.

It is important to note however that technical analysis does not always lead to correct or even desirable results. However, it is not correct to discredit technical analysis; since it does help the investor to take informed decisions and make knowledgable predictions with respect to certain securities/stocks [15].

Analysis, documented and properly calculated, technically, will not only help us to take appropriate decisions but also help us make near-accurate predictions about the stock market trends and values in the future. The three most significant benefits of proper technical analysis on stock market data for stock market predictions, irrespective of whether it is long-term or short-term, are:

- It will help to easily identify the support and resistance levels in that particular security.
- It will help to take informed decisions about good (or bad) entry points.
- Most significantly, it will help to easily spot trends and patterns in existing historical data and make informed and knowledgable predictions about the future with respect to the particular stock/security that is being technically analysed.

The objective of our work is to analyse the factors influencing our predictions and henceforth measure the risk for the predictions. Also, here several regression techniques are used and compared to deduce the risk that is involved in this prediction game.

2 Review of Existing Literature

Supervised learning encompasses different regression techniques. Regression analysis was first coined based on a biological phenomenon - that the descendants of ancestors who were originally very tall, had heights that were gradually decreasing towards a normal average or the mean; in other words, the values were regressing down towards the mean value [1]. This concept has, since then, been put to use in largely statistical analogies, and is now a proven way to explain the relationships between the variables that are of explanatory nature and the cumulative distribution of all the recorded responses [2].

Non-parametric forms of regression, Bayesian and Naive-Bayesian regression methodologies and multi-predictor-valued regression methodologies are some of the rather important advancements on the statistical front [3].

In their work, Pedregosa et al. developed scikit-learn, the highly beneficial machine learning library to be used complementary to the Python programming language. Some of the rather very significant objectives that have been met by scikit-learn include fast and convenient implementations of a wide range of regression analysis techniques and methodologies in code for machine learning purposes [4].

2.1 Linear Regression

The linear regression model is the most basic of all the models that are a part of the module under scikit-learn called the sklearn.linear_model. The main aim of these modules is to implement comparatively statistically generalised models based on linear regression analysis techniques. The *LinearRegression* module is essentially the ordinary regression analysis taking into account the least squares, the class definition for which is as follows:

*class sklearn.linear_model.**LinearRegression** ( )*

As far as the implementation of this statistical technique is concerned, it is very convenient, since this method takes into account only the Ordinary Least Squares segment. To add to the convenience, the segment has been wrapped into an object of type predictor such that it allows for easy usage and provides efficient and reliable solutions.

The scope of this *LinearRegression()* class is to help us make a pattern for the prices of a stock with respect to dates, and use it to make predictions for the unknown dates, which serves as the test data.

2.2 Ridge Regression

The implementation of Ridge Regression enters the domain of statistical analysis with the help of the concept of Regularisation. The *Ridge* module works under the linear_model of scikit-learn using the principles of l2 regularisation. So the main principle behind ridge regression is to combine the least squares results obtained in linear regression, with l2 form of regularisation; and the aim of this module is, like that of *LinearRegression*, to minimise the cost function $J(\theta)$, where:

$$J(\theta) = \frac{1}{2m}[\sum_{i=1}^{m}(hypothesis - predicted\_values)^2 + \lambda \sum_{j=1}^{m} \theta_j^2]$$

The l2-norm of regularisation is evident from the latter half of the equation shown above, in the segment: $\lambda \sum_{j=1}^{m} \theta_j^2$,

where the values of the weights of the parameters are squared to give the regularised term, after multiplication with the regularisation factor (denoted by λ). This form is also called Tikhonov regularisation [5]. The class definition for the *Ridge* module is almost an evolution over that for *LinearRegression*, that has been previously defined, which is as follows:

*class sklearn.linear_model.**Ridge** ( )*

2.3 Bayesian Ridge Regression

The Bayesian Ridge Regression is very much similar to the normal Ridge Regression. In fact, it is nothing more than a significant improvement over the ordinary least squares methodologies followed till now; despite the fact that as of today, there are many disagreements over the most optimised version of ridge regression [6]. The class definition for the *BayesianRidge* module, shown below, adds to that of the *Ridge* module with the implementation of fitting a particular Bayesian Ridge Regression statistical analysis model, followed by the optimisation of the significant-role-playing parameters, namely the regularisation parameter (denoted by λ) and the noise precision (denoted by α).

*class sklearn.linear_model.**BayesianRidge** ( )*

Ridge regression is by far quite capable of preventing any forms of over-fitting to given data leading to poor results on new data. The Bayesian Ridge Regression technique is essentially an almost equivalent methodology that takes merit in being a very flexible approach with respect to construction of the models for a given problem [7]. The bayesian technique is actually very intuitive when we take into the account the uncertainty about the estimations of the weights of the parameters involved, to help in the process of regularisation. The advantages of the *BayesianRidge* computational module are quite pronounced. Not only does it have provisions for the proper studying and estimation of the posterior probabilities with regards to the weight coefficients of the parameters, but also aids in easy interpretation. Adding to it, is the fact that there is an enhanced experience with respect to flexibility in the design of the models for any given problem scenario. However, the only noticeable and significantly observable drawback of this approach employing the Bayesian ridge regression is that the result predictions take significantly more time to be solved and computed.

2.4 Lasso Regression

The Least Absolute Shrinkage and Selection Operator (Lasso) is in itself a statistical method for regression analysis. The model that is analysed by the Lasso regression technique is not only more accurate with respect to the prediction scores, but is also significantly interpretable. This is because lasso not only fits the parameters, but also penalises the coefficients (or the weights) such that we can effectively assign importance levels to the coefficients, understand their significance in the analysis and the score-predictions by the model, and use this knowledge to essentially totally disregard some of the coefficients (depending on their "importance levels" of course), and make use of only the most significant ones in the prediction process [8]. The *Lasso* module strives to implement the same concept. However, the equation that implements this form of regression analysis, the lasso is as follows:

$$J(\theta) = \frac{1}{2m}[\sum_{i=1}^{m}(hypothesis - predicted\_values)^2 + \lambda \sum_{j=1}^{m} ||\theta_j|||][5]$$

The class definition for the *Lasso* module is given below:

*class sklearn.linear_model.**Lasso** ( )*

All the 4 modules discussed here, undertake some common set of functions, that are denoted by the pre-built methods for the same in the scikit-learn library for Python. The method-descriptions are as follows:

1. **Fitting the Model:** For fitting the linear regression model for any particular regression analysis problem, it is important to call the **fit** method, the syntax for which is: *fit (X, y, sample_weight = None)*
   The returned value is *self* - which is essentially an instance of the model itself.
2. **Getting Estimator Parameters:** To get the parameters for any given estimator, the implementation is: *get_params (deep = True)*
   This returns a mapping (*params*), from that of names of the parameters concerned to their values.
3. **Setting Estimator Parameters:** To set the parameters for any given estimator, the implementation is: *set_params (**params)*; and this works not only with single estimators, but also with pipelines and other nested objects of that nature. It is convenient in the sense that it is possible to easily update all the components of a nested object. The returned value is *self* - which is essentially an instance of the model itself.
4. **Making Predictions Using the Model:** To predict by making use of the linear regression model (the *LinearRegression()*, module to be specific), the implementation is simply: *predict (X)*, where X is an array-like shape or a sparse data-representation. The value that is returned by this function is also an array - the predicted output measurements.
5. **Checking the Performance Score:** By using *score(X, y, sample_weight = None)*, we can find the performance score of the model with respect to a particular regression analysis problem. The score of a prediction is essentially its coefficient of determination (R^2). The coefficient R^2 can be defined as $R^2 = 1 - (u/v)$, where u is essentially the residual sum after the squaring, and v is the actual cumulative sum of the squares, such that:

$$u = \sum (y\_true - y\_prediction)^2, and$$

$$v = \sum (y\_true - y\_true.mean())^2.$$

The value that is eventually returned from this method is the float-type value of the coefficient of determination (R^2), which is the prediction score for the model.

2.5 FBprophet

The FBProphet forecasting model is a relatively new development in the field of time-series forecasting. This statistical practice applies not only to linear forms of regression but also non-linear regression analysis techniques. According to Taylor et al. [9], FBProphet is providing a very practical approach to what is known as forecasting-at-scale. Prophet makes use of a decomposable model for the time-series forecasting, and has three main components, namely, trend, holidays and seasonality. It is noteworthy that the model is similar to a GAM or generalised additive model in the fact that apparently time is the only regressor, whereas many other linear (as well as non-linear) derivatives of time are the components of the model. The trend function serves to model non-linear and non-periodic changes in time-series values; the seasonality component takes care of the periodic (for instance, yearly) changes in the time-series data; and last but not the least, the holidays function makes sure that even the effect of sudden and potentially irregular changes in schedules affect the time-series data values, prominently seen over one or more days. The Prophet forecasting time series model can thus be expressed as:

$$y(t) = trend(t) + seasonality(t) + holidays(t) + \epsilon_t,$$

where, the term ϵ_t refers to idiosyncratic changes [9] in the model that are not accommodated by the same, and *y(t)* refers to the trend of the overall model.

2.6 Risk Analysis

The primary concept worth noting with respect to the value of the bid-ask spread is to make sure that the investors would be at a disadvantage irrespective of the predicted value of the bid-ask spread [10]. A very crucial step in the domain of risk analysis and management, thereafter, is the actual calculation with respect to the risks that are actually present in the current scenario [16]. For this very reason, risk functions have been devised to streamline the process of risk calculation. Two very important and significant risk functions, among others, that have been around for quite some time now are the NIST risk function [12] devised by the NIST (the National Institute of Standards and Technologies) and the MEHARI (Method for Harmonised Analysis of Risk) risk function [11]. It is noteworthy that although other risk functions are in use, which are somewhat more efficient than the NIST or the MEHARI risk functions, these two risk functions help to properly depict the causes and the consequences when it comes to risk in long-term stock market predictions.

The NIST Risk Function: The NIST risk function is a quantitative as well as qualitative tool for the analysis and calculation of risk associated with a particular scenario [12]. This risk function takes as input, the impact of a given threat or vulnerability (or, the many factors that determine stock market prices and their fluctuations), and the likelihood of that impact factor, which is essentially the probability associated with that risk factor ever occurring. The output of the risk function is of course the "Risk" value, which is quantitatively defined on a number of scale of suitable range. Both inputs, the impact and the likelihood can have their values in qualitative as well as quantitative terms.

The MEHARI Risk Function: The MEHARI risk function is almost exactly similar to the NIST risk function, both structurally and functionally. The main difference lies in the fact that the MEHARI risk function takes into account everything in a qualitative way. Consequently, the inputs to the risk function, i.e. the impacts and the likelihoods, are both defined only qualitatively [11]. Also, the output, which is the risk associated, is defined qualitatively, and hence cannot be defined numerically.

3 Methodology

The dataset that has been used in the regression techniques have been recorded for each working day for the period of approximately 8 years. The conventional name of the financial security, of which this dataset is a part, is called the SP500. To be precise the daily SP500 values have recorded in the dataset from December 8, 2008 up until August 30, 2017. According to this dataset, there are approximately 252 entries for one full year of SP500 data.

The regression analysis resulted in:

a score of 0.936747975549684 (approximating to 0.93 over 5 rounds of testing) when *model.LinearRegresssion()* was used on the dataset,

a score of 0.9474807086816863 (approximating to 0.945 over 5 rounds of testing) when *model.Ridge()* was used on the dataset,

a score of 0.9545439315539958 (approximating to 0.954 over 5 rounds of testing) when *model.BayseianRidge()* was used on the dataset, and

a score of 0.9568207062764479 (approximating to 0.96 over 5 rounds of testing) when *model.Lasso()* was used on the dataset.

The results can be depicted on the plot as given in Fig. 1.

The prediction by the FBProphet forecasting model can be used to actually visualise the model training parallel to the actual data points. This can be visualised in Fig. 2. Likewise, the final outcome from the FBProphet forecasting model has been depicted in Fig. 3. The greyed out areas are essentially the confidence-bounds as well as the predictions for the values and the predicted confidence-bounds for the respective values, over the course of the next two years which have been used to serve the purpose of a test-dataset. The results, as can be seen, are quite consistent initially, but then diverge out (the confidence-bounds) for later periods of time. So, FBProphet might not apparently be able to perform as efficiently for the long-term predictions as compared to predictions for a shorter time.

3.1 Proposed Risk Function

To make sure that the risk associated with stock market predictions can be analysed and predicted to a certain extent, a new risk function needs to be devised that might turn out to be an improvement over the likes of the NIST and the MEHARI risk functions. The risk function that has been devised to particularly fulfill the needs for risk analysis in the domain of long-term stock

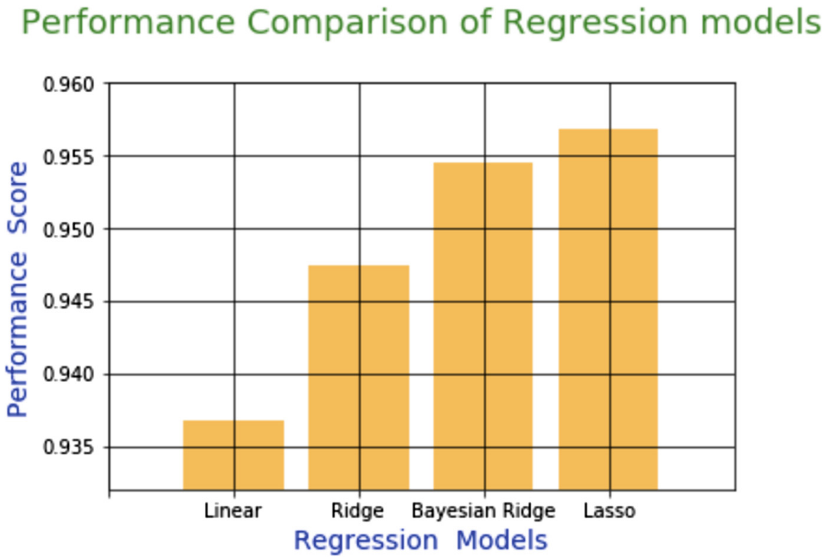

Fig. 1. Comparative study of the regression analysis techniques

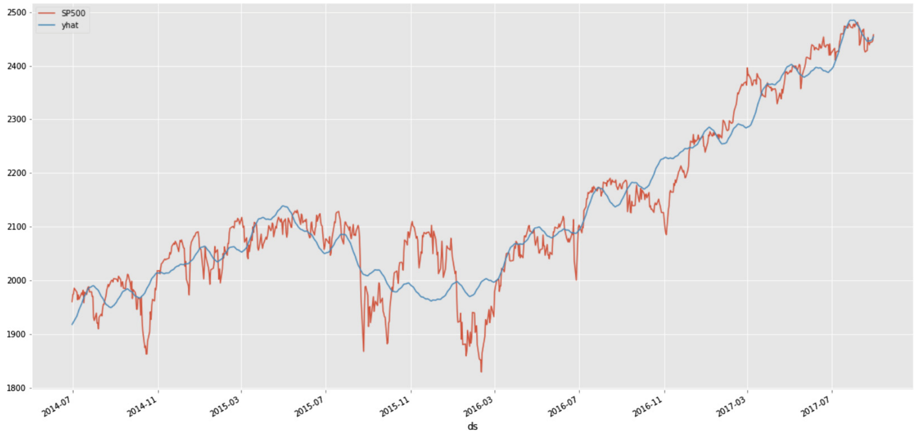

Fig. 2. The FBProphet forecast result for 8 years

market predictions is based loosely on both the NIST and the MEHARI risk functions. Like the NIST and the MEHARI risk functions, it has qualitative valuation for the inputs, "Impacts" and "Likelihoods". Complementary to that, the risk function also includes the quantitative valuation of the input parameters, allowing for scaling and suitable-range-selection for the same. However, unlike the lenience that is allowed by the NIST risk function with respect to the value-levels, the new risk function has a fixed set of 4 levels for each of the input parameters, not unlike the MEHARI risk function. The output for the newly

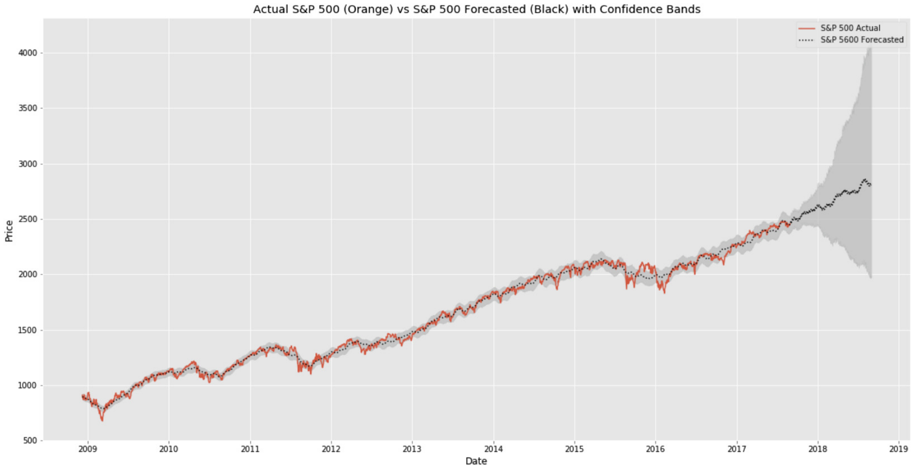

Fig. 3. The FBProphet forecast result with final predictions

devised risk function for long-term stock market predictions and risk analysis, is of course the risk associated with an investment taking into account a given impact factor for a particular likelihood. Like the NIST risk function, this output value is quantitative in nature, and akin to the MEHARI risk function, there are strictly 4 levels (or, ranges) of risk that are attributed to the quantified output value. The new risk function can be seen in Fig. 4.

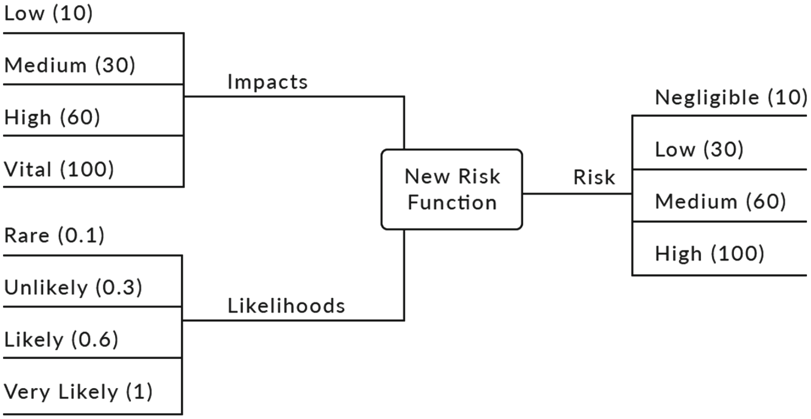

Fig. 4. The new risk function

The impact areas include, but are not limited to factors that affect the stock market namely, momentum, mean reversion, martingales, volatility, stock valuations, interest rates, economic outlooks, inflation (and/or deflation), economic and political shocks, changes in economic policies, exchange rate valuations, supply-and-demand factors, market (or sometimes, even individual [17]) sentiments, and so on and so forth.

4 Future Prospects of Study

The long-term predictions for the stock market with regression analysis techniques and the FBProphet forecasting model, and the risk analysis, have kept huge scope for improvement and future prospects for in-depth study.

1. The first scope of future prospects that this has given us is to study not only the effect of prices for corresponding dates, but to also include other factors in the correlation, such as the volume of the security, the asking price, the bidding price, the bid-ask spread [13], and so on and so forth, either individually or in conjunction with each other.
2. The second scope for improvement and future studies lies in the fact that we can now incorporate even short-term prediction studies to make sure that we can understand the economic dynamics even at a smaller time-scale, which might include other important factors such as election-effects, political fluctuations, social dynamics and day-to-day sentiments, terrorist attacks, natural calamities et cetera.

And the most significant outcome from these 2 areas of study is that we will not only be able to improve on the newly devised risk function for risk analysis in stock-market predictions, but also be able to evaluate and consequently, map and classify new forms of risk associated with this domain, such as *market-value risk, headline risk, rating risk, obsolescence risk, detection risk, legislative risk, inflationary* (and/or *deflationary) risk, interest-rate risk, business-model risk, marketability risk, convenience risk, safety risk, purchasing-power risk, political/societal risk*, and so on and so forth.

5 Conclusion

The ultimate aim of stock market prediction is to make sure that the risk is minimised. Increasing the accuracy might seem synonymous to the previous statement, but it does not hold true for all scenarios. Even when we make predictions with the maximum accuracy (in the ideal case), a certain percentage of risk persists. However, the more the risk can be analysed and ascertained with efficiency, the more knowledgable will be the decisions that are taken on the part of the investor, or any other individual entity linked to the stock market. The aim of this paper is to state that regression analysis studies and other forecasting models can predict future trends in the long-term stock market dynamics with certain accuracy, but only when it is combined with a proper risk function, to properly analyse and mitigate the risk, is it actually fruitful in terms of benefits provided to the real world. The results obtained from this study can safely suggest that, not only are we on the right track to properly work out risks associated with stock market predictions and consolidate the importances of regression analysis in stock market predictions, but also pave the way for future studies and improvements for more refined analysis strategies and efficient results.

References

1. Galton, F.: Kinship and Correlation (reprinted 1989). Statistical Science, Institute of Mathematical Statistics (1989)
2. Fisher, R.A.: The goodness of fit of regression formulae, and the distribution of regression coefficients. J. Roy. Stat. Soc. **12**, 773 (1922)
3. Aldrich, J.: Fisher and regression. Stat. Sci. **20**, 401–417 (2005)
4. Pedregosa, F., et al.: Scikit-learn: machine Learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
5. Ng, A.Y.: Feature selection, L1 vs. L2 regularization, and rotational invariance. In: Twenty-first International Conference on Machine Learning (2004)
6. Vinod, H.D.: A survey of ridge regression and related techniques for improvements over ordinary least squares. Rev. Econ. Stat. **60**, 121–131 (1978)
7. Bishop, C.M., Tipping, M.E.: Bayesian Regression and Classification. Microsoft Research (2003)
8. Friedman, J., Hastie, T., Tibshirani, R.: Regularization Paths for Generalized Linear Models via Coordinate Descent (2010)
9. Taylor, S.J., Letham, B.: Forecasting at Scale (2017)
10. Copeland, T.E., Galai, D.: Information Effects on the Bid-Ask Spread (1983)
11. CLUSIF, MEHARI* V3: Risk Analysis Guide (2004)
12. Stoneburner, A.G.G., Feringa, A.: NIST: RIsk management guide for information technology system. Special Publication 800–30. National Institute of Standards and Technology (2002)
13. Nesbitt, K., Barrass, S.: Finding trading patterns in stock market data. IEEE Comput. Graph. Appl. **24**, 45–55 (2004)
14. Boyacioglu, M.A., Avci, D.: An adaptive network-based fuzzy interference system (ANFIS) for the prediction of stock market return: the case of the Istanbul stock exchange. Expert Syst. Appl. **37**, 7908–7912 (2010)
15. Kimoto, T., Asakawa, K., Yoda, M., Takeoka, M.: Stock Market Prediction System with Modular Neural Networks (1990)
16. Yang, H., Chan, L., King, I.: Support vector machine regression for volatile stock market prediction. In: Yin, H., Allinson, N., Freeman, R., Keane, J., Hubbard, S. (eds.) IDEAL 2002. LNCS, vol. 2412, pp. 391–396. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45675-9_58
17. Bhattacharya, S., Goswami, S., Poray, J., Bose, R., Das, A.: Study on sentiment and opinion for “Review Text” corpus. In: ICCIIoT (2018)