Technical Analysis-Driven Stock Market Prediction Using a Novel History Bits Machine Learning Approach

Nitin Sakhare[1,*] , Divya Midhun[2], Dharmesh Dhabliya[3]

[1]Department of Computer Engineering, Vishwakarma Institute of Information Technology, Pune, India
[1]Lincoln University College, Malaysia.

[2]Lincoln University College, Malaysia.
[3]Department of Information Technology, Vishwakarma Institute of Information Technology, Pune, India.

nitinsakhare4@gmail.com, divya@lincoln.edu.my, dharmeshdhabliya@gmail.com

Abstract-

Stock market movement has been of immense interest to financial experts for the long time. Investors and analysts frequently analyze historical market data—such as stock prices (including open, high, low, and close values) and trading volumes—to detect patterns and trends.. Although technical analysis seems to be a commonly used method to evaluate market behavior, the signals can sometimes be contradictory. A good example would be one indicator saying that we have a buying opportunity and another one signaling a sell at the same time thereby making us losses because of poor trading timing. In the stock markets, Machine learning (ML) models have emerged as very powerful for enhancing the reliability of predictions of the NIFTY 50 index on the Indian stock exchange. This study introduces History Bits, a new algorithm designed to predict stock trends based on historical trading data, and describes how it is used to give meaningful insights into stock trends. The idea behind the algorithm is to help investors make buying or selling decisions better informed. The History Bits algorithm uses a total of 75 of technical indicators based on core of trading metrics like men such as opening and closing prices, highs and lows, and trade volume. Thus, these indicators are ranked and filtered by an ensemble based feature selection method to select attributes as informative as possible. The model was trained and tested on a pre processed NIFTY 50 dataset across two decades for evaluation. Cross comparison of the proposed model with the existing well known machine learning algorithms such as Decision Tree, Naïve Bayes, Random Forest, Support Vector Machine (SVM), and Multilayer Perceptron Artificial Neural Network (MLP-ANN) was made. The results prove that History Bits algorithm prediction is a reliable and simple method that manages to obtain high prediction accuracy compared to its counterparts.

Keywords— Stock market; prediction; technical analysis; machine learning; Decision Tree; Naïve Bayes; MLP; Random Forest, SVM; History Bits

1. Introduction

Like any other investor, they have the aim to create wealth or capital appreciation through their funds. There are many avenues other than the stock market in which such investments can take place; one particular prominent avenue is the stock market where several companies are listed thus becoming available for trades. In this environment, sellers set their asking price, buyers place bid at certain prices. When these two prices participate they complete [2]. However, in that case, when more than one investor bids on the same stock, the first bid gets considered first while attracting the shares. An example of forecasting stock prices includes the analysis of past data to estimate future value of a company's shares traded in on an exchange. Often, the high volatility in the stock market forces investors to take decisions at the wrong moment, missing the chance to take profit [3]. This

unpredictability adds question and complexity into forming well informed trading choices. We would have the potential to earn quite a lot of returns if future stock prices could be accurately predicted. Technical analysis plays a crucial role in assessing various tradable instruments—be it stocks, indices, commodities, or futures—whose values are influenced by supply and demand dynamics. Such analysis may span from an intraday interval (say, 1 minute, 5 minutes, or hourly) to daily, weekly, even monthly, datasets making its time frame several hours or a number of years. The purpose of technical analysis is to assist investors in making more informed and secure investment decisions, as well as in identifying promising trading opportunities. [4] Often its effectiveness is evaluated in terms of the profits it helps to generate. Technical analysis is based on the relationship between stock demand and supply. There are different technical indicators, which provide unique outlook to market trends and conditions [5]. As it was possible with these indicators, investors could create strategies in which entries, exits, and settling trades would follow certain specific rules. This study does not develop individual trading strategies, but presents a clear way of using technical indicators with machine learning techniques to predict market movements and establish buying and selling tips [6–7–8]. According to experts, it is not recommended to use just one indicator in technical analysis [9]. One more effective way is to use a number of indicators at the same time. Technical analysis using machine learning enables faster and a more accurate prediction on stock market dataset, instead of relying on an extensive analysis. It enables traders to find profitable trades and avoid risks as well as alter their strategies in the dynamic way by responding to real time market behavior [10].

The development of stock price movement forecasting has incorporated various ML techniques during different periods. The researchers from Tsai and Quan [11] established a stock prediction system that utilized technical indicators to identify candlestick chart patterns for market analysis. Their study did not address the individual importance of the utilized indicators. The incorporation of multiple diverse indicators tends to improve predictive precision levels. Feng et al. [12] established a high-yield stock detection model through Temporal Graph Convolutional Networks for maximizing expected returns. The study demonstrates neural network capabilities in stock prediction but additional comparison tests between different ML and deep learning approaches would identify optimal methods. The stock forecasting process becomes more effective by using a hybrid platform which merges Extreme Learning Machine (ELM) with Support Vector Machine–Back Propagation Neural Network (SVM-BPNN) as proposed by Xiaodong Li. This approach incorporated both historical pricing data and market-related news. To further extend the study, one must add fundamental analysis of the market, which is vital for long term investment making decisions. Furthermore, it would be useful to look into a broader spectrum of machine learning models in order to improve prediction accuracy. As described in [13], Calzon et al. developed a group-based decision-making framework for indicator selection, which was optimized through expert input. This approach emphasizes collaborative evaluation, allowing domain experts to contribute their insights in selecting the most relevant indicators, thereby enhancing the reliability and effectiveness of the predictive model. The objective in their methodology was to avoid poor trading decisions. Working with the above, it could be possible to combine expert inputs with technical indicator signal predictions to obtain more precise predictions. The Indian stock market volatility was analyzed by Sheikh, Alam, and Agarwal [14] by using Auto-Regressive (AR), Moving Average (MA), and ARMA models of time series. The addition of technical indicators based on volatility, momentum oscillators and moving averages would also improve these models. Pehlivanli and Guzhan [15] have contributed to indicator optimization by information gain for feature selection in order to predict the one day in advance of stock prices. In general, they were able to sift out and discard the useless and redundant indicators. The alternative attribute evaluation techniques, i.e., PCA [16], Pearson correlation, gain ratio, and mutual information, aid to further enhance this process. Jean-Marc Le Cailee [17] attempted to investigate how different combinations of technical indicators would affect the performance of different ML algorithms. In his work, he stresses importance of having a variety of indicators rather than using the same kind of indicators, such as moving averages, which can give you conflicting or misleading signals. There is also a stock trading model inspired by classical feedback control systems by Ross Barmish [18]. We enrich this framework with statistical technical indicators and presence a backtracking analysis to perform the result review. Based on frequent pattern mining to reveal sequence from the noisy financial time series, Min Wen [19] proposed a new methodology for analysing such noisy financial time series. Using machine learning models, the identification of frequent trading patterns could be expanded using this technique. In his work [20], Eunsuk Chong explored the use of three unsupervised feature extraction techniques: PCA, autoencoders, and restricted Boltzmann machines to the stock prediction task. Forecasting accuracy improved considerably when their integration with autoregressive models

is performed. Feature extraction techniques applied on a wider set of technical indicators can lead to the generation of optimum indicator subsets. At last, the hybrid predictive model comprised of Support Vector Machine and K-Nearest Neighbors (KNN) has been developed by Rudra Kalyan Nayak [21] for the Indian stock market technical analysis. The prediction performance was studied using different kernel functions. Additional ML algorithms are extended to provide further insights into effectiveness and profitability of model.

Its volatility and dynamism makes it a very challenging task for anyone to predict the movement of the stock market. Still, machine learning (ML) has shown to be helpful in revealing the hidden patterns and bringing the useful information into that will be helpful in making a decision. ML models can predict future market trends by analysing many things that affect the stock prices and this prediction can help the investors make a more strategic decision and take more profits from their investments. Machine learning has been very widely applied in different domains, and it also showed very promising results in financial market forecasting. Technical analysis is used for short term trading by the analysis of several types of technical indicators, such as trend, momentum, volatility and volume indicators and the options from them to find market possibility [20]. While under no circumstances can perfect accuracy in guessing the fluctuations of the stock market market be achieved merely by the use of market data from the past, historical market data for building predictive models can reduce the risk substantially and help minimize losses in cases of hasty trades. Investors need to be able to interpret stock related data accurately and they need accurate forecasts to make profits. There is plenty of time-series data surrounding the behavior of a market, and analysis is required to be effective.

In this research, we introduce a novel machine learning algorithm, termed **History Bits**, aimed at improving the accuracy of stock market prediction. To evaluate its effectiveness, we compare its performance against several well-established algorithms, including Multilayer Perceptron Artificial Neural Network (MLP-ANN), Decision Tree, Support Vector Machine (SVM), Random Forest, and Naïve Bayes. Both proposed model as well as all other models are trained and tested on a trend specific transformed version of the NIFTY dataset for the past 20 years.

In the remainder of the paper, we structure the research dataset, the set of technical indicators utilized and an ensemble based ranking approach used for feature selection. The proposed History Bits algorithm is explained in detail in section 3, and subsections of it are discussed here. In section 4, History Bits is benchmarked against various machine learning approaches for stock market prediction to assess its relative performance. Finally, the study concludes by highlighting the key findings and insights derived from the comparative analysis.

2. Research Data

For the experimental analysis, we utilized the NIFTY 50 index dataset from the National Stock Exchange (NSE) of India, incorporating 75 technical indicators to enhance the depth and robustness of the predictive modeling. The methodology is flexible and can be used on dataset of any company listed on either National Stock Exchange (NSE) or Bombay Stock Exchange (BSE) if the dataset comprises fields such as date, open, high, low, close prices and trading volume [21]. These datasets can be fetched from official sources like www.nseindia.com and are of level 1 dataset i.e. raw market data. Transforming this data set based is an important step of our approach which involves computing a wide variety of technical indicators. In the final step, the FINTA (Financial and Technical Analysis) library that can be found on GitHub at https://github.com/leomrocha/finta for this purpose. The level 2 dataset is the output of this step that includes computed statistical values for the 75 technical indicators. The level 2 dataset is further processed to the level 3 dataset which involves interpretation of the indicator signals to produce actionable insights. Unlike many previous studies based on binary classification, that is, buy or sell, our work brings in a much more complex classfication. Specifically, we define five such predictive classes: strong_buy, buy, hold, sell, and strong_sell. It is meant to provide investors with an enhanced granularity that can help in making more informed and precise trading decisions. In order to evaluate History Bits, we have conducted extensive experimentation using the NIFTY data set over 20 years of time, from June 1999 to November 2019. Introducing such an extended spanning time period ensures data sample diversity and representation as the model evolves to be more robust and generalizable in the face of possible overfitting to some market condition or period in time.

Table 1: Technical Indicators

Moving Averages & Trends	Oscillators & Momentum Indicators	Volume-Based Indicators	Other Technical Tools and Patterns
Simple Moving Average (SMA)	Stochastic Oscillator (%D and %K)	Volume Weighted Average Price (VWAP)	Ichimoku Kinko Hyo (Ichimoku Cloud)
Exponential Moving Average (EMA)	Stochastic RSI	On-Balance Volume (OBV)	Bollinger Bands
Weighted Moving Average (WMA)	Williams %R	Volume Flow Indicator (VFI)	Pivot Point Analysis (Standard and Fibonacci)
Triangular Moving Average (TMA)	Relative Strength Index (RSI)	Volume Zone Oscillator (VZO)	Commodity Channel Index (CCI)
Hull Moving Average (HMA)	Ultimate Oscillator	Volume Price Trend (VPT)	Chande Momentum Oscillator (CMO)
Zero-Lag Exponential Moving Average (ZLEMA)	Awesome Oscillator	Accumulation/Distribution Line	Directional Movement Index (DMI)
Smoothed Simple and Exponential Averages	Market Momentum (MOM)	Chaikin Oscillator	Average True Range (ATR)
Double and Triple Exponential Moving Averages	Rate of Change (ROC)	Ease of Movement (EMV)	Mass Index
Adaptive and Efficiency-Based Averages (KAMA, KEI)	Know Sure Thing (KST)	Money Flow Index (MFI)	Elder's Force Index
Elastic Volume-Based Averages (EVMA, EVMACD)	True Strength Index (TSI)	Volume Adjusted Moving Average (VAMA)	Coppock Curve
Typical Price, Median, and Moving Standard Dev.	Inverse Fisher Transform	Weighted OBV, Price Zone & Buy-Sell Pressure (BASP)	Chandelier Exit
Squeeze Momentum Indicator	Vortex Indicator	Cumulative Force Index	Adaptive Price Zone
Vector Size Indicator	Finite Volume Element	Twiggs Money Flow	Qstick Indicator
Stop and Reverse (SAR)	Fisher Transform	Bull/Bear Power	Convergence-Divergence MACD

3. Experimental Work

In this work, for improving the accuracy of stock market trends prediction, we propose a History Bits based algorithm. An experimental validation of the proposed method is made on a dataset coming from the Indian stock market, the NIFTY 50 index, during a period of 20 years having a total of 4,977 data instances. Table 2 presents an excerpt of raw NIFTY 50 dataset available online for reference. This dataset is the basis of the calculation of a lot of technical indicators of great importance for technical analysis. Summary of the calculated indicators that comprise the enriched dataset for training and testing of the model is shown in Table 3.

Table 2: Layer 1 dataset (sample)

Date	Low	Open	Close	High
12-Nov-18	10587.3	10610.5	10644.2	10675.9
16-Nov-18	10661.3	10673.9	10709.5	10726.8
19-Nov-18	10719.5	10762.4	10794.8	10806.1
20-Nov-18	10670.3	10771.3	10685.9	10772.1

Table 3: Layer 2 dataset (sample)

Date	PIVOT	VWAP	EMA	TP
15-Nov-18	--	10606.9	10616.7	10606.9
16-Nov-18	10606.9	10640.45	10653.09	10669.5
19-Nov-18	10669.5	10670.85	10698.3	10742.3
20-Nov-18	10742.3	10672.91	10684.04	10679.3

The stockanical indicators are technical indicators which incorporate the stock's movement trends into interpreted numerical values. These trends indicated are interpreted and transformed into actionable trading signals, which would be buy or sell. Table 4 illustrates the mapping of numerical indicator outputs to trading signal.

Table 4: Layer 3 dataset (sample)

Date	TP	VWAP	Pivot	EMA
11-Nov-18	sell	buy	--	hold
13-Nov-18	hold	hold	buy	buy
19-Nov-18	hold	buy	sell	buy
20-Nov-18	buy	sell	buy	sell

We turn this problem into a five class classification problem by adopting a wisdom of the crowd [[rule 1]]. The prediction classes are : strong_buy, buy, hold, sell and strong_sell. Providering this multi class categorize gives you much more granular, actionable trading signals than the traditional binary classification.
Furthermore, to evaluate the effectiveness of the proposed History Bits algorithm, a comparative analysis was conducted against several well-known machine learning models, including Support Vector Machine (SVM), Random Forest, Multilayer Perceptron (MLP), Naïve Bayes, Decision Tree, and Artificial Neural Network (ANN), we have performed comparative analysis for all algorithms. Despite the wide recognition of their effectiveness, these algorithms tend to apply well in classification and prediction tasks across all domains, most notably in financial forecasting problem areas.

3.1. Decision Tree

Here Decision Tree uses a hierarchical tree structure, each internal node represents an attribute (feature), and each leaf node represents a class used for predicting. Key assumptions of the model are as follows:

- The entire training dataset is initially considered as the **root node**.
- Decision Trees are better suited for **categorical features**; hence, continuous attributes are typically **discretized** before model construction.
- The selection of attributes for internal nodes is guided by **attribute selection metrics** such as **Information Gain** and **Gini Index**, which help determine the optimal order of attribute placement in the tree.

At each node, the algorithm **partitions** the training instances based on attribute values. This partitioning alters the **entropy** of the dataset, and the **reduction in entropy** is quantified as **Information Gain**.

$$Gain(I, A) = Entropy(I) - \sum_{x=0}^{n} V(A) \cdot E(I_x I_x) \qquad (1)$$

Here's a refined version of your statement:
Let **I** represent the set of instances, **A** the set of attributes, and **x** the values that these attributes can assume, where$I_x I_x$ is subset of I denotes the subset of instances having attribute value **x**.**Entropy** quantifies the uncertainty or impurity associated with an attribute — the higher the entropy, the greater the uncertainty. Conversely, a higher **information gain** is achieved when the entropy is reduced, making the attribute more useful for classification or prediction tasks.

3.2. Naïve Bayes

The **Naïve Bayes** algorithm operates under the assumption that each feature is **independent** of the others, meaning that the presence (or absence) of one feature does not influence the presence (or absence) of any other feature.
This assumption allows each feature to contribute equally and independently to the final prediction.

Naïve Bayes utilizes **Bayes' Theorem** to calculate the probability of an event AAA occurring, given the occurrence of another event BBB. The theorem is mathematically expressed as:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \qquad (2)$$

Where **P(B)** remains unchanged.
In this scenario, we aim to determine the chance of occurrence of event X, assuming that event Y has already taken place. Therefore, Y is termed as the observed data or evidence. The likelihood of X occurring independently, expressed as P(X), is called the prior probability, reflecting the estimated chance of event X before factoring in any occurrence of event Y. Conversely, the conditional probability of X given Y, denoted as P(X|Y), is known as the posterior probability, indicating the updated chance of X happening after accounting for the influence of event Y.

For prediction purposes, let Y represent the outcome variable (the class label we wish to predict), and let X be the set of input features (attributes) used for prediction. Using Bayes' Theorem, the posterior probability can be calculated based on the likelihood of X given Y, the prior probability of Y, and the evidence from X.

$x = \{x_1, x_2, x_3, \dots, x_n\}$

According to Bayes theorem,

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \qquad (3)$$

$$p(y|x_1, x_2, \dots, x_n) = \frac{P(x_1(y)p(x_2|y)\dots PCy)}{p(x_1, P(x_2)\cdots P(x_n)} \qquad (4)$$

Since the denominator remains unchanged, equation (4) can be reformulated as:

$$p(y|x_1, x_2, \dots, x_n)\alpha P(Y)\Pi_{i=1}^{n} P(X_i|Y)(Y) \qquad (5)$$

To build a Naïve Bayes prediction model for all possible outcomes, we calculate the probability for each input feature set and select the outcome with the highest probability.

$$Y = argmax_y P(y)\Pi_{i=1}^{n} P(X_i, Y) \qquad (6)$$

The probability of **Y** represents the class probability, while **P(Xi | Y)** denotes the conditional probability of feature **Xi** given class **Y** [10].

3.3. MLP-ANN

The perceptron is a fundamental linear classification algorithm designed to separate data into two distinct classes. This works by dividing the input space in two regions using a straight line (or a hyperplane in higher dimensions) that divides the input space into two regions, one containing examples from each of the classes and the other being the complements to this set, one which contains examples from each of the class, and the other being the complements region.

$$y = w * x + b \qquad (7)$$

In this version of the approach the input features are measured in terms of a vector and this is multiplied by the appropriate set of weights. Then, a bias term ensures that the decision boundary is shifted away. The output of the perceptron is determined by the result of this linear combination. Activation functions are applied to the result of a linear computation, to introduce non linearity and handle more complex relationships between input and output.

These activation functions model non linear pattern so a broader range of input conditions can be handled by the perceptron. As with the basic perceptron, the use of activation functions in a perceptron enables it to take more complex classification tasks on more difficult data; however this is done at the expense of the perceptron's ability to process linearly separable data only.

$$y = \varphi(\sum_{i=1}^{n} w_i x_i + b) \qquad (8)$$

$$y = \varphi(w^T + b) \qquad (9)$$

One of the classification models employed in this study is the Multilayer Perceptron (MLP). An MLP is a type of neural network consisting of an input layer, one or more hidden layers, and an output layer. For example, taken one

layer at a time in the previous layers is a set of neurons all connected to the neurons below. Using also these connections carry weights, that can be trained in the continuation to make better the model performance.

Due to the use of non-linear activation functions in its neurons, the MLP can effectively model complex and non-linear relationships within the data. It learns through a function approximation approach that maps input features to their corresponding target labels. The training process is supervised and employs the backpropagation algorithm, which adjusts the model's weights based on the error between predicted and actual outputs. The weights are initially assigned randomly. When the error is not zero, the weights are updated from the error and from the input values and at the same time they see the value of the learning rate. The learning rate sets how much this will change in the weights with every iteration and how much speed and stability the learning will have.

$$W = W + l * (actual - predicted) * x \quad (10)$$

The forward pass during training is the first step, which is used to utilize inputs and process them through the layers to produce an output. A loss function is used to compute difference between this output and actual label. Better performance is given by a lower loss. In the backward pass the gradients of the loss against each weight are determined and updated so future errors are reduced. MLPs are generally suitable for tasks concerning complex patterns, and their flexible architecture can accommodate different classification problems, as it has been shown in this research, for example, the prediction of stock market trends.

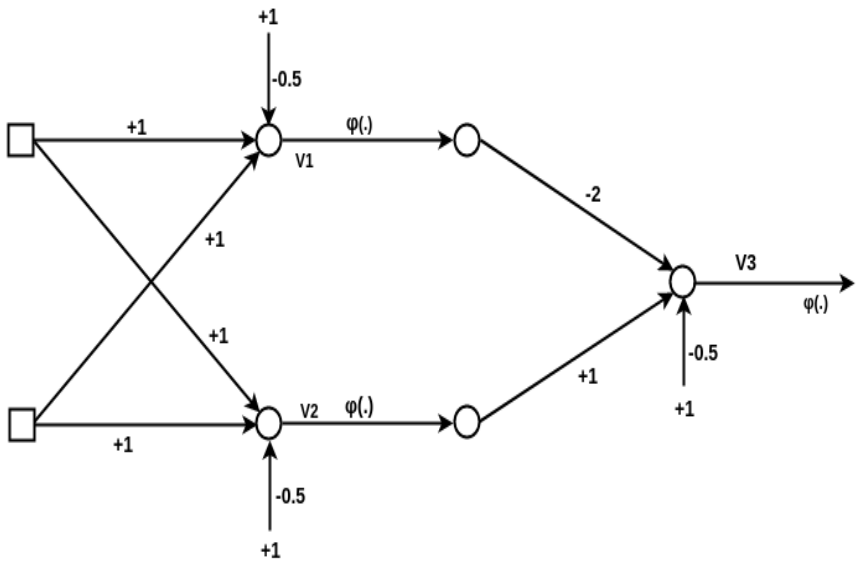

FIGURE 1: Forward pass in MLP

Training an MLP includes assessing the loss during its second operational phase after the forward pass completes. Customary output emerges from the forward pass so it becomes the expected forecast output. The estimated output receives comparison against actual class labels to calculate a loss value which defines forward pass error.

The backward pass constitutes the third vital training phase that leads to weight updates of the network. The loss function gradient calculation with respect to the weights during gradient descent obtains this achievement. Weight adjustments performed during the backward pass move toward decreasing loss values so that the model develops improved prediction abilities. By performing forward and backward passes numerous times the MLP develops its ability to create precise outputs by reducing the total prediction inaccuracy.

3.4. Random Forest

The supervised learning algorithm Random Forest constructs numerous decision trees through the application of bootstrapped subsets taken from the training data. The training data splitting process generates equal-sized settings which use sampling with replacement for each subsetting. Random Forest creates predictions by combining the outputs from its individual decision trees which use majority voting for classifications and average voting for regression problems. The procedure called Bootstrap Aggregation functions under the name Bagging. Random Forest serves as an ensemble learning algorithm by using multiple decision trees to reach better accuracy and more reliable results. It combines predictions of different trees and it is meant to prevent overfitting, which is a common issue of one single decision tree. Individual trees are averaged out to reduce biases and variances, hence the ensemble nature helps in error reduction. In bagging, a dataset is sampled with replacement, so the amounts drawn are NN (of the original dataset) and some instances may be repeated while some others may be missing. It is geared up with random decision tree for random subset of this subsets in the sense that it trains one of these decision tree in the forest on one of this subsets, and through this randomness it improves the model's diversification. Evidently, this process reduces the tree correlation, and leads to more accurate predictive results overall.

3.5. Support Vector Machine(SVM)

SVM serves as a supervised learning method which finds a maximal boundary between separate data classes by identifying optimal hyperplanes containing support vectors. SVM achieves maximum separation between its hyperplane and data points of each class through support vectors which act as the closest data points. The portion between hyperplane and support vectors constitutes the margin region. Higher margin size creates better generalization when classifying new data thus making the optimal hyperplane selection possible. Linear SVM becomes the suitable option when data points are easily distinguishable using straight linear boundaries or flat planes. SVM employs kernel functions during the mapping process of data points to higher-dimensional spaces when linear separation of data proves impossible. The model becomes capable of discovering linear boundaries in this new space to perform more effective class separation.The position of new data points regarding the decision boundary determines their assigned class. A point located on the hyperplane side gets assigned to a particular class but a point on the opposite side goes to the competing class. A reliable classification boundary requires optimal position and adjustment of the margin across all data points to reach the best separation.

if y * f(x) >= 1

c (x, y, f(x)) = 0 (11)

else,

c (x, y, f(x)) = 1 – y * f(x) (12)

When the predicted value matches the actual value, the cost is considered zero. In cases where they differ, the loss is computed through a cost function. To ensure a balanced model, a regularization parameter is applied to manage the trade-off between reducing the loss and maximizing the decision margin.

$$min\,\lambda|w|^2 + \sum_{i=1}^{n}(1 - y_i\langle x_i\mu\rangle) \quad (13)$$

Gradients are obtained by calculating the partial derivatives of the loss function with respect to the model's weights, and these gradients are subsequently applied to adjust the weights during the optimization process.

$$\frac{\delta}{\delta w_k}\lambda|w|^2 = 2\lambda w_k \quad (14)$$

$$\frac{\delta}{\delta w_k}(1 - y_i\langle x_i, w\rangle) = 0 \; if \; y_i\langle x_i, w\rangle \geq 1$$
$$= -y_i x_{ik} \quad (15)$$

When SVM correctly classifies the gradient will be updated using regularization parameter.

$$w = w - \alpha, (2\lambda w) \quad (16)$$

In the event of a misclassification, the gradient is updated based on the calculated loss and the regularization parameter, ensuring both error correction and model generalization.

$$w = + \alpha, (y_i x_i - 2\lambda w) \quad (17)$$

3.6. History Bits Algorithm

This research suggests the introduction of a new algorithm, which we base on the History Bits principle, to perform a highly efficient stock trading. Historical stock market data are used by the algorithm to turn it into trend based buy or sell signals on 75 technical indicators. The signals in Table 4 are generated using a statistical interpretation of each indicator This is different from many existing studies which predict 2 outcome of buy or sell but our model expands that to a 5 outcome classification system: strong buy, buy, hold, sell and strong sell. The broader classification provides a more updated vision of market trends and helps investors make better decisions. It can be argued that the core of the algorithm is crowds wisdom strategy. The voting mechanism used in this method is simple while the final prediction relies on the majority of indicators. The main idea is that we have more accurate results by combining many separated indicators. Using these indicators' diversity reduces the effect of random noise and inconsistency that may affect reliability of prediction. This ensemble based method helps to make the overall predictive performance of this model stronger. The algorithm is more stable and robust in decision making in the sense that it depends on the collective intelligence of multiple indicators. This results in introducing the model as a very strong tool to forecast stock market trends with higher accuracy and confidence..

Algorithm: Preparation of Transformed Dataset
Input: Dataset with Buy/Sell recommendations from 75 indicators.
Output: Transformed dataset with labels:

- strong_buy
- strong_sell
- buy
- sell
- hold

Steps:

1. For each trading day (i = 1 to n):
    - Count percentage of indicators showing Buy signals.
    - Count percentage of indicators showing Sell signals.
2. Assign class label based on these percentages:
    - If ≥ 75% indicators signal Buy → label = strong_buy

- If ≥ 75% indicators signal Sell → label = strong_sell
- If indicators signaling Buy > 55% and < 75% → label = buy
- If indicators signaling Sell > 55% and < 75% → label = sell
- If indicators signaling Buy or Sell are between 45% and 55% inclusive → label = hold

3. Repeat the above steps for all trading days.

End of Algorithm

Using its statistical values, this paper selects 75 technical indicators for determining if buy or sell signals are generated. Some indicators will give accurate calls but others will give wrong signals. One key idea behind the wisdom of crowd approach is the idea of using collectively the power of these indicators to correct themselves of one another lowering of individual errors. As a result, five different classes of predictions are formed in this approach as follows:

1. Strong_Buy / Strong_Sell : If for example, a day 60 or more indicators are bought or sold out, it will be then classified as strong Buy or strong Sell for next day.
2. The next day prediction will be hold if it is balanced outcome like 37 indicators suggest to buy and 38 indicate to sell.
3. Buy or Sell: For that number between 39 and 59 of the indicators generating a buy or sell signal, the prediction will be labeled as buy or sell, respectively.

This way, all indicators, even those that generate false signals, contribute to keep leading the process of decision making in the right direction, regardless of some indicating these false signals. Together working, they offer more reliable and confident grounds for making trading decisions.

The History Bits algorithm employs an attribute ranking strategy that organizes the 75 technical indicators into multiple groups and assigns priority to them. It incorporates various ranking techniques, such as information gain, correlation coefficient, gain ratio, OneR, relief evaluator, and symmetric uncertainty evaluator, to assess the relevance of each attribute for stock market prediction.

However, relying solely on one ranking strategy could introduce bias in determining the ranks of the indicators. To mitigate this bias, we propose an **ensemble-based ranking strategy**, which combines the outcomes from different ranking algorithms, thereby minimizing errors that might arise from any single ranking method.

Once the indicators are ranked, the **History Bits algorithm** categorizes them into groups based on their priorities. In our case, we have divided the 75 indicators into five groups, with each group containing 15 indicators. After grouping, weights are assigned to the indicators according to their respective groupings, which further aids in making the final prediction. This methodology ensures that the technical indicators are appropriately prioritized and weighted, leading to a more accurate prediction model.

Step 1 – Indicators are ranked based on their importance, and those with a higher rank are given more weight. At the beginning, weights are assigned to each indicator using a method similar to how weights are initialized in perceptron learning. This helps in giving more influence to the most relevant indicators during the prediction process.

$$y = w * x + b \quad (18)$$

$$y = \varphi\left(\sum_{i=1}^{n} w_i x_i + b\right) \quad (19)$$

$$y = \varphi(w^T + b) \quad (20)$$

However, the assigned weights are not updated under any circumstances, and the process then advances to step 2.

Step 2: **History Bit Generation for Each Group**

- For each group, the algorithm evaluates whether the count of 'buy' signals surpasses that of 'sell' signals. If the number of 'buy' signals is greater, the history bit for that group is assigned a value of 1; otherwise, it is assigned a value of 0.

However, Step 2 may introduce ambiguity when there is a close contest between the number of 'buy' and 'sell' signals. This scenario arises when the difference between their counts is minimal, potentially causing slight deviations in the prediction and increasing the likelihood of generating inaccurate trading signals.
For example, the **relief evaluator** ranking can lead to a scenario where:

- Indicators in the first group are ranked higher than those in the second group.
- However, some indicators in the second group may generate conflicting signals (a mix of 'buy' and 'sell'), leading to ambiguity in determining whether to assign a **1** or **0** to the bit pattern for that group.

This **boundary-case** scenario—where the number of 'buy' and 'sell' calls are almost equal—could potentially cause a small deviation in prediction. Nonetheless, such cases are rare and have minimal impact on the overall prediction accuracy due to the **average approach** used in the algorithm, which helps smooth out these deviations over a large dataset.

- In the worst-case scenario, less than 20% of indicators in the first three groups might generate signals opposite to the majority signals. Therefore, these rare cases have a negligible effect on the model's overall performance.

Step 3: **Collecting the Bit Pattern**

- After processing all the groups, the history bit patterns generated for each group are collected. These bit patterns are binary values representing the signal for each group, either **1** (for 'buy' signals) or **0** (for 'sell' signals).

Step 4: **Bit Pattern Interpretation**

- The resulting bit pattern will contain 5 bits, corresponding to the 5 groups of indicators. Each bit is derived from the signals generated by the indicators within each group.

Table 5: Interpretation of the bit patterns

Strong Buy	Buy	Hold	Sell	Strong Sell
['1', '1', '1', '1', '0']	['0', '1', '1', '1', '0']	['0', '1', '1', '0', '0']	['0', '1', '0', '1', '0']	['0', '0', '0', '1', '0']
['1', '1', '1', '0', '1']	['1', '0', '1', '1', '0']	['1', '0', '1', '0', '0']	['0', '0', '1', '1', '0']	['0', '1', '0', '0', '0']
['1', '1', '0', '1', '1']	['1', '1', '0', '0', '1']	['0', '1', '0', '1', '1']	['0', '0', '1', '0', '1']	['0', '0', '1', '0', '0']
['1', '0', '1', '1', '1']	['0', '1', '0', '1', '0']	['0', '1', '1', '0', '1']	['1', '0', '0', '1', '0']	['0', '0', '0', '1', '1']
['1', '1', '1', '0', '0']	['1', '1', '0', '1', '0']	['0', '0', '1', '0', '1']	['1', '0', '0', '0', '0']	['0', '0', '0', '0', '1']
['1', '0', '1', '1', '0']	['0', '1', '1', '0', '1']	['0', '0', '0', '1', '0']		
['1', '1', '1', '1', '1']	['1', '0', '1', '0', '1']			
['1', '1', '0', '1', '0']	['0', '1', '1', '1', '1']			
	['1', '0', '0', '1', '0']			
	['0', '1', '0', '0',			

	'1']			
	['1', '0', '1', '0', '0']			
	['0', '0', '1', '1', '0']			

4. Result and Discussion

Thus, in the present work, a new History Bits algorithm is introduced to predict the values of stock market using multiple indicators which are averaged to come up with an improved forecast. Asked as to the reason for employing 75 technical indicators in the case of stock price prediction, our endeavor is to achieve great results with the help of improved prediction results of the collective knowledge of 75 technical indicators.

In order to test the efficiency of History Bits algorithm, the proposed algorithm was compared with some renowned one in machine learning area.

Decision Tree

Naïve Bayes

Multilayer Perceptron Artificial Neural Network (MLP-ANN)

Random Forest

Support Vector Machine (SVM)

In this research, we considered the dataset of the NIFTY 50 index of the Indian stock market consisting of historical trends of the market at different time intervals. To achieve this, the dataset was divided into a training set comprising of 70% and testing set of 30% to check for ability of the model in predicting the performances on brand it has never encountered before.

Evaluations were made by using the confusion matrix of 5x5 the three categories which include strong_buy, buy, strong_sell, sell, and hold. This confuses matrix presents the instances that were classified correctly and those that were classified wrongly hence better performance comparison among all the models.

The values that can be obtained from the confusion matrix are as follows:

Accuracy: Overall proportion of correct predictions.

What kind of metrics can be used for evaluation:

Precision: Percentage of the right labels for each class.

Recall: The actual number of instances of the class that have been identified correctly in relation to the total actual number of instances in the class.

F1-Score: Derives from the ‘precision-values’ and the ‘recall-values,’ then their average value is found offering a fair assessment.

The findings discussed in this section also verify that the History Bits algorithm that has been proposed in the present work is capable of integrating information from a diverse set of technical indicators. This type of integration allows the model to analyze the data on stock market development with greater accuracy of patterns and trends detection. It can be seen in the presented paper that the History Bits approach achieves the same or even better results compared to the other conventional machine learning models in different tasks of prediction.

In this case, the strength of the model is seen from the general of the indicators which are summed up within the context of making of the forecast results. They present various aspects of the information which makes the model more capable to understand the market trends.Therefore, comparing the strategies laid out in both models can afford insight into the differences between them as well as the advantages and disadvantages of each. In this context, it is possible to establish that History Bits achieves a relatively equivalent efficiency for different classes and offers reasonable outcomes in most cases.

When evaluating the performance of each classification model some criteria were put into consideration. A target signal is the situation wherein the model accurately picked out a signal, while an unrelated signal is when the model inaccurately picked up on a negative signal. This measures the precision of the model whereby if a particular class is chosen, then the degree of correctness is measured. F-measure is a better measure than the accuracy rate as it integrates precision and the recall in its assessment, which gives round efficiency of the model. Furthermore, Matthews Coefficient gives a single value of reliability of the model with reference to the actual and predicted values and also the false positive and false negatives. In general, the analysis proves that the History Bits algorithm is applicable for achieving high and stable accuracy in stock market predictions. The result is mainly attributed to the use of a broad range of the technical indicators which enables the identification of collective intelligence.

4.1. Performance evaluation of Decision Tree

Table 6: Summary of the performance

Total number of test instances	1510
Correctly classified instances	1140
Misclassified instances	370
Overall prediction accuracy	75.50%
Kappa statistic	0.6812
Mean Absolute Error (MAE)	0.1089
Root Mean Squared Error (RMSE)	0.2943

Table 7: Decision Tree - Confusion matrix

Actual \ Predicted	Hold (a)	Buy (b)	Strong Buy (c)	Sell (d)	Strong Sell (e)
Hold (a)	69	30	0	26	0
Buy (b)	34	313	64	1	0
Strong Buy (c)	0	98	315	0	0
Sell (d)	17	4	0	284	61
Strong Sell (e)	0	0	0	37	140

Table 8: Detailed Accuracy

Class	TPR (Recall)	FPR	Precision	F1-Score	MCC

Hold	0.556	0.039	0.582	0.568	0.682
Buy	0.755	0.119	0.710	0.732	
Strong Buy	0.768	0.063	0.825	0.793	
Sell	0.779	0.055	0.809	0.794	
Strong Sell	0.785	0.044	0.705	0.743	

4.2. Performance Evaluation of Naïve Bayes

Table 9: Summary of the performance

Total number of test instances	1493
Correctly classified instances	1064
Misclassified instances	429
Overall prediction accuracy	71.26%
Kappa statistic	0.632
Mean Absolute Error (MAE)	0.1162
Root Mean Squared Error (RMSE)	0.3245

Table 10: Naïve Bayes - Confusion matrix

Actual \ Predicted	Hold (a)	Buy (b)	Strong Buy (c)	Sell (d)	Strong Sell (e)
Hold (a)	104	10	0	11	0
Buy (b)	87	251	74	0	0
Strong Buy (c)	0	87	325	1	0
Sell (d)	56	0	0	260	50
Strong Sell (e)	0	0	0	53	124

Table 11: Detailed Accuracy

Class	TPR (Recall)	FPR	Precision	F1-Score	MCC
Hold	0.832	0.105	0.421	0.559	0.633
Buy	0.609	0.090	0.721	0.661	
Strong Buy	0.787	0.069	0.815	0.800	
Sell	0.710	0.058	0.800	0.753	
Strong Sell	0.701	0.038	0.713	0.707	

4.3. Performance evaluation of Multilayer Perceptron

Table 12: Summary of the performance

Total number of test instances	1498
Correctly classified instances	1140
Misclassified instances	358
Overall prediction accuracy	76.11%
Kappa statistic	0.692
Mean Absolute Error (MAE)	0.1015
Root Mean Squared Error (RMSE)	0.288

Table 13: Multilayer Perceptron - Confusion matrix

Actual \ Predicted	Hold (a)	Buy (b)	Strong Buy (c)	Sell (d)	Strong Sell (e)
Hold (a)	81	26	0	17	0
Buy (b)	31	309	73	0	0
Strong Buy (c)	0	72	342	0	0
Sell (d)	21	3	0	291	49
Strong Sell (e)	0	0	0	59	128

Table 14: Detailed Accuracy

Class	TPR (Recall)	FPR	Precision	F1-Score	MCC
Hold	0.638	0.036	0.612	0.624	0.693
Buy	0.751	0.095	0.752	0.751	
Strong Buy	0.822	0.070	0.818	0.820	
Sell	0.793	0.061	0.808	0.800	
Strong Sell	0.719	0.041	0.715	0.717	

4.4. Performance evaluation of Random Forest

Table 15: Summary of the performance

Total number of test instances	1490
Correctly predicted instances	1210
Misclassified instances	280
Prediction accuracy	81.21%
Kappa Statistic	0.751
Mean Absolute Error (MAE)	0.129
Root Mean Squared Error (RMSE)	0.2396

Table 16: Random Forest - Confusion matrix

Actual \ Predicted	Hold (a)	Buy (b)	Strong Buy (c)	Sell (d)	Strong Sell (e)
Hold (a)	65	30	0	31	0
Buy (b)	14	340	60	1	0
Strong Buy (c)	0	52	361	0	0
Sell (d)	13	0	0	307	46
Strong Sell (e)	0	0	0	41	136

Table 17: Detailed Accuracy

Class	TPR	FPR	Precision	F1-Score	MCC
Hold	0.516	0.021	0.691	0.592	0.752
Buy	0.823	0.078	0.802	0.812	
Strong Buy	0.874	0.058	0.861	0.867	
Sell	0.838	0.062	0.814	0.825	
Strong Sell	0.779	0.060	0.749	0.764	

4.5. Performance evaluation of Support Vector Machine

Table 18: Summary of the performance

Total number of test instances	1493
Correctly predicted instances	1170
Misclassified instances	323
Prediction accuracy	78.36%
Kappa Statistic	0.714
Mean Absolute Error (MAE)	0.247
Root Mean Squared Error (RMSE)	0.3271

Table 19: Support Vector Machine - Confusion matrix

Actual \ Predicted	Hold (a)	Buy (b)	Strong Buy (c)	Sell (d)	Strong Sell (e)

Hold (a)	75	22	0	27	0
Buy (b)	21	315	70	1	0
Strong Buy (c)	0	61	353	0	0
Sell (d)	13	2	0	295	56
Strong Sell (e)	0	0	0	45	132

Table 20: Detailed Accuracy

Class	**TPR**	**FPR**	**Precision**	**F1-Score**	MCC
Hold	0.593	0.024	0.689	0.631	0.719
Buy	0.767	0.079	0.780	0.773	
Strong Buy	0.851	0.065	0.829	0.840	
Sell	0.812	0.065	0.803	0.807	
Strong Sell	0.752	0.040	0.718	0.734	

4.6. Performance evaluation of History Bits

For the proposed algorithm, there is no training-testing split of the dataset and we consider all the instances as unknown instances

Table 21: Summary of the performance

Total number of instances	4975
Correctly predicted instances	4350
Misclassified instances	625
Prediction accuracy	87.45%
Kappa Statistic	0.792
Mean Absolute Error (MAE)	0.282
Root Mean Squared Error (RMSE)	0.844

Table 22: History Bits - Confusion matrix

Actual \ Predicted	Buy (a)	Hold (b)	Sell (c)	Strong Buy (d)	Strong Sell (e)
Buy (a)	14	20	0	108	0
Hold (b)	30	330	19	129	14
Sell (c)	2	80	13	0	18
Strong Buy (d)	5	10	0	2596	0
Strong Sell (e)	0	179	16	0	1402

Table 23: Detailed Accuracy

lass	TPR	FPR	Precision	F1-Score	MCC
Buy	0.094	0.009	0.253	0.142	0.793
Hold	0.632	0.062	0.550	0.589	
Sell	0.121	0.008	0.270	0.172	
Strong Buy	0.995	0.096	0.924	0.953	
Strong Sell	0.880	0.012	0.986	0.981	

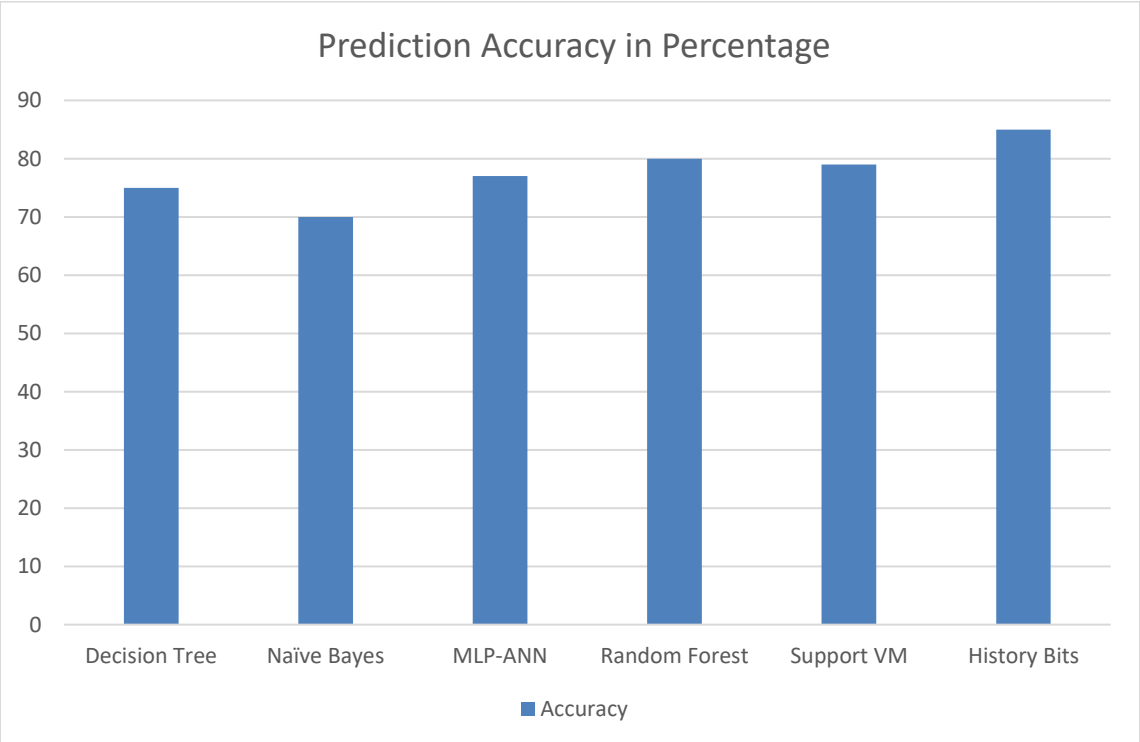

FIGURE. 2: Prediction accuracy comparison of history bits with other algorithms

From the results, it is clear that the proposed history bits algorithm outperforms other machine learning algorithms in the prediction task

5. FUTURE SCOPE & CONCLUSION

Stock forecasting has long been a key focus for investors, aiming to predict future trends and make informed decisions about buying and selling. This paper introduces a novel History Bits algorithm, designed to enhance decision-making by using 75 technical indicators derived from stock trading data (open, high, low, and close prices). The proposed algorithm prioritizes these indicators through an ensemble-based attribute ranking strategy, combining the results from multiple ranking methods, such as Information Gain, Correlation Coefficient, OneR, Relief Evaluator, Gain Ratio, and Symmetric Uncertainty Evaluator, to minimize bias and improve the reliability of predictions. The History Bits algorithm goes beyond traditional two-class predictions by offering five prediction classes: strong_buy, buy, hold, sell, and strong_sell, providing traders with more nuanced decisions. The paper compares the performance of the History Bits algorithm with other well-known machine learning techniques, such as Decision Tree, Naïve Bayes, Multilayer Perceptron, Support Vector Machine, and Random Forest, showing superior prediction accuracy. Notably, the History Bits algorithm is unsupervised, which makes it adaptable and potentially useful as input for other machine learning models. Future work could explore integrating fundamental analysis and testing the algorithm on different markets to assess its broader applicability and generalization.

REFERENCES

[1] Tsai, C and Quan, Z. Stock Prediction By Searching for Similarities in Candlestick Charts, DOI: 10.1145/2591672, ACM Transaction on Management Information System, Vol. 5, Issue 2. 2014.

[2] https://www.investopedia.com/terms

[3] Minh, D., L., Niaraki, A., S., Huy, H., D., Min, K., Moon, H and Deep Learning Approach For Short Term Stock Trends Prediction Based On Two-Stream Gated Recurrent Unit Network, DOI: 10.1109/ACCESS.2018.1868970, INSPEC Accession Number: 18159342, IEEE Access, vol. 6, 2018, pp. 55392-55404.

[4] Barmish, B., R., and Primbs, J., A. On a New Paradigm For Stock Trading Via a Model- Free Feedback Controller, DOI: 10.1109/TAC.2015.2444078, INSPEC Accession Number: 15806153, IEEE Transaction on Automatic Control, Vol. 6, Issue 3, 2016.

[5] Huynh, H., D., Dang, L., M., and Dhong, D. A New Model for Stock Price Movement Prediction Using Deep Neural Network, DOI: 10.1145/3155133.3155202, ACM, *Proceedings of the 8th ISICT*, 2017, pp. 57-62.

[6] Li, X., Xie, H., Wang, R., and Cai, Y.,. Empirical Analysis: Stock Market Prediction Via Extreme Learning Machine , DOI: 10.1007/s00521-014-1550-z, Springer-Neural Computing, and Applications, Vol. 27, Issue 1, 2016, pp. 67-78.

[7] Sakhare, N., and Imambi, S. Performance Analysis of Regression-Based Machine Learning Techniques for Prediction of Stock Market Movement, IJRTE, Vol.7, issue 6S, 2019.

[8] Patel, J., Shah, S., Thakkar, P., and Kotecha, K.. Predicting stock market index using fusion of machine learning techniques, DOI: 10.1016/j.eswa.2014.10.031, ScienceDirect, Expert Systems with Applications, 2014, pp.1-11.

[9] Sakhare, N., and Joshi, S. Classification of Criminal Data Using J48 Algorithm, International Journal of Datawarehousing and Mining, 2014, pp. 167-171.

[10] Ocharoem, P., and Vateekul, P. Deep Learning Using Risk-Reward Function for Stock Market Prediction, DOI: 10.1145/3297156.3297173, ACM, Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence, 2018, pp. 556-561

[11] Yetis, Y., Kaplan, H., and Jamshedi, M. Stock Market Prediction by Using Artificial Neural Network", DOI: 10.1109/WAC.2014.6936118, INSPEC Accession Number: 14700939 IEEE, World Automation Congress, 2014.

[12] Sharma, N., and Juneja, A. Combining of Random Forest Estimates Using LSboost for Stock Market Index Prediction, DOI: 10.1109/I2CT.2017.8226316, INSPEC Accession Number: 17449287, *2017 2nd International Conference for Convergence in Technology,* 2017.

[13] Joshi, S., and Sakhare, N. History Bits based novel algorithm for Classification of structured data. DOI: .1109/IADCC.2015.7154779, INSPEC Accession Number: 15292807, 2015 *IEEE International Advance Computing Conference (IACC)*, Banglore, 2015, pp. 609-612.

[14] Feng, F., He, X., Wang, X., Luo, C., Liu, Y., Chua, and T. Temporal Relational Ranking for Stock Exploration, DOI: 10.1145/3309547, ACM Transaction on Information System, Vol. 37, Issue 2, 2019.

[15] Calzon, C., B., Bulla, H., M., Garzas, J., J., E., Herrera, F., and J. Expert Selection in Prediction Market With Homological Invariants,DOI: 10.1109/ACCESS.2018.2846878, INSPEC Accession Number: 17905708, IEEE ACCESS, Vol. 6., 2018, pp. 32226-32239.

[16] Waqar, M., Dawood, H., Bilal, M., Ali, M., and Guo, P. Prediction of Stock Market by Principle Component Analysis, DOI: 10.1109/CIS.2017.00139, INSPEC Accession Number: 17578286, *2017 13th International Conference on Computational Intelligence and Security (CIS).*

[17] Pehlivanli, A.C., and Guzhan, A. Indicator Selection with Committee Decision of Filter Methods for Stock Market Price Trend in ISE, DOI: 10.1016/j.asoc.2016.09.004, Elsevier, Applied Soft Computing, Volume 49, 2016, Pages 792-800

[18] Cailee, J., L., Itani, A., Gueriot, D., and Rakotondratsimba, Y. Stock Picking by Probability-Possibility Approaches, DOI: 10.1109/TFUZZ.2016.2574921, INSPEC Accession Number: 16776016, IEEE Transaction on Fuzzy Systems, Vol. 25, Issue 2, 2017.

[19] Wen, M., Pi, L., Zhang, L., and Chen, Y. Stock Market Trend Using High Order Information of Time Series DOI:10.1109/ACCESS.2019.2901842, INSPEC Accession Number: 18521046, IEEE Access Vol. 7, 2019, pp.28299-28308.

[20] Chang, E., Han, C., Frank, and C. Deep Learning Networks for Stock Market Analysis and Prediction: Methodology, Data Representation, and Case studies, DOI: 10.1016/j.eswa.2017.04.030, Elsevier, ScienceDirect, Expert Systems with Applications, Vol. 83, 2017, pp. 187-20

[21] Patel, J., Shah, S., Thakkar, P., and Kotecha, K. Predicting stock and stock price index movement using trend deterministic data preparation and machine learning techniques, DOI: 10.1016/j.eswa.2014.07.040, ScienceDirect, Expert Systems with Applications, 2014.

[22] Slanka, C., Skiera, B., and Span, M. Prediction Market Performance and Market Liquidity: A comparison of Automated Market Makers, DOI: 10.1109/TEM.2012.219618, INSPEC Accession Number: 13234687, IEEE Transaction on Engineering Management Vol. 60, Issue. 1, 2013.

[23] Zhang, W., Li, C., Ye, Y., Li, W., and Ngai, E. Dynamic Business Network Analysis for Correlated Stock Price Movement Prediction, DOI: 10.1109/MIS.2015.25, INSPEC Accession Number: 15013000, IEEE Transaction on IEEE Intelligent Systems, 2015.

[24] Wu, M., and Diao, X.,. Technical Analysis of Three Stock Oscillators- Testing MACD, RSI and KDJ Rules in SH and SZ stock markets, DOI: 10.1109/ICCSNT.2015.7490760, INSPEC Accession Number: 16090318, *4th International Conference on Computer Science and Network Technology (ICCSNT),* 2015.