Automatic Creation of Stock Trading Rules on the Basis of Decision Trees

Dmitry Iskrich, Dmitry Grigoriev
St. Petersburg State University
St. Petersburg, Russia
st034749@student.spbu.ru, d.a.grigoriev@spbu.ru

***Abstract*—The main task of any trader is the selection of profitable strategies for trading a financial instrument. One of the simplest ways to represent trading rules is binary decision trees based on the comparison of current values of technical indicators with some absolute values. This approach simplifies the creation of trading systems but their validity is limited to short-term intervals of trade. This paper represents an approach for generating more universal trade rules in the form of binary decision trees based on the comparison of the current values of technical indicators with relative levels. The ranges of these levels are recalculated on a daily basis which allows the trading rule to remain relevant within a long term. The proposed system analyzes the historical price data for a long period and using the genetic algorithm causes trading strategies optimized relatively to the Sharpe ratio.**

I. Introduction

A strategy of a trader concluded in the formalization of rules for opening and closing a position. Quite often these rules are based on methods of the technical analysis and processing of such data as the current price of the instrument, the volume of trading, the maximum / minimum price for a certain period, the price at the time of opening / closing of the trading session. Based on the series of these data, so-called technical indicators are generated. In accordance with postulates of the technical analysis technical indicators allow to predict the probable trend direction in a given security. Thus, the combination of the indicator and its value can serve as a basis for the trading strategy [1], [2], [3].

The discovery of profitable patterns is a non-trivial task. Typically a trader manually programmes his trading system and then optimizes its key parameters by exhaustive search or by heuristic algorithms [4]. In this paper we describe a method for automatic building the rules of a trading system without human intervention. Obviously, its use will allow us to discover a greater number of effective patterns of price behavior.

One of the simplest representations of trade strategy is binary decision trees [5], [6], [7]. In this case nodes contain various assumptions about the state of a technical indicator or another characteristic of the current market situation, and its truth or falsity is determined by both branches. Methods, which use this data structure have a wide area of application [8], [9].

In our work, a similar approach is used, which was first presented in [10]. Its specificity is the use of the genetic algorithm for the generation and selection of optimal decision trees. This algorithm relies on the daily historical data of indicators and generates the most "trained" trading decision tree. Unlike in [7] and [11], the values of technical indicators in the tree are compared not with the real value, but with one of the levels relative to the dynamics in the past. This increases the time of the rule's relevance, as the ranges of levels are recalculated on a daily basis and thus they reflect the latest state of the market.

It should be noted that similar systems use BUY/SELL signals allowing opening short positions and working with them symmetrically to the long ones. In practice, however, there are limitations: the adequate estimation of the effectiveness of short positions is difficult because of the necessity to pay for the borrowed securities; the period of their retention should be as short as possible; there are temporary or permanent prohibitions on their opening.

There are systems based on the markup of historical data with instructions to take a trade action. This stage is necessary for training some classifiers the essence of which is to find the relationship between a particular class and the values of technical indicators. However, at the moment there is no optimal way of such a markup.

So, for example, in paper [8] for the arrangement of BUY, SELL and HOLD signals for a certain day the trading indicators of the next two days were used. After that using algorithm ID3 with these data a decision tree was created. At the same time, according to allegations [12] for the determination of the trade action it may be insufficient to have several subsequent values. The general trend of price changes plays an important role, and its duration time is not always obvious. In system [13] the arrangement of BUY and SELL signals is shown on the basis of a fall or rise in prices for 2 years by 10 or 15 percent respectively. Data with the assigned marks are used by algorithm C4.5 to create the optimal decision tree. The same algorithm is used in paper [14], however, the markup of the data for training is based on the real actions of the most successful trader on a specific day.

Thus, the abundance of all possible variants of the markups of historical data per trading actions led to the decision to use the genetic algorithm as a classifier. This approach requires only the existence of the objective function from the data for the decision tree, thus eliminating the need to subjectively mark out historical data.

The contribution of this paper is as follows:

1) In trade rules only long positions are allowed.

2) A new optimization algorithm for the decision tree is presented which is obtained after the application of crossover and mutation operators.

3) The principle of calculating the levels of indicators is described.

A more detailed description of the tree structure is given in Section II. Section III contains a description of the genetic algorithm including the initialization process and the classical evolutionary steps. Principles of testing the best systems and the analysis of their effectiveness are presented in Section IV.

II. Trading decision tree

A decision tree is a connected acyclic graph with the leaves represented by decisions, in our case these can be signals for opening a long position (LONG) and exit from it (CLOSE LONG). The process of determining decisions starts from the root, then, depending on the node value, the transition into either the right or the left node takes place. As a result, the leaf at the end is taken, i.e. an order to buy a stock and open a long position (LONG) or sell out the purchased volume (CLOSE LONG). The system starts with the CLOSE LONG state allowing the first purchase with the LONG signal. The position size is determined by the maximum possible number of shares that can be purchased at the current balance. Thus, all the received income is reinvested.

A tree node is a predicate defining a transition to one of the two children. In every non-terminal node, the value of the technical indicator is compared to a certain level to see whether they match.

A. Calculation of the indicator level

Most often indicators have real values, but their current state can be associated with a certain level, this approach proved itself in [15]. In other words, at a given time, a technical indicator can have one value from the enumeration: {*Very Low, Low, Medium, High, Very High*}.

To calculate this value, the system uses n previous values of the indicator. In this system, n equals to the duration of the training period. After that the calculation of the ranges for each level takes place:

1) The maximum (Max) and minimum (Min) values of the indicator for the previous n days are computed.
2) On the basis of these indicators, an interval [Min, Max] is generated and divided into 5 equal segments.
3) The current level is determined by the segment into which the real value of the indicator falls.

If the value exceeds Max, the indicator level is assigned $VeryHigh$. Similarly, if the value is less than Min the indicator level will be $VeryLow$.

It should also be noted that the current level of indicator is determined after closing the trading session and it is valid for the whole next trading day.

In the simplest form, the decision tree is shown in Fig. 1.

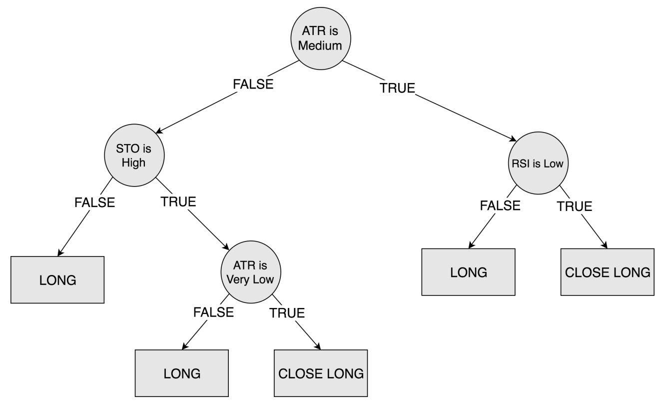

Figure 1. Example of an trading decision tree

III. Genetic algorithm

To identify the best tree the genetic algorithm is used. All steps of the classic evolution are involved.

A. Initialization

The initial population is initialized with a preset number of random generated trees, with each node created by selecting a random indicator and associating it with a random level. The height of the tree is determined by the number of indicators available to the system (in our case 3). After this height has been reached, the generation of non-terminal nodes ceases and random LONG/CLOSE LONG leaves are created.

B. Selection

For the further tree modification with the genetic operators the individuals that fit most are selected and thus the next generation is formed. The selection is done by the roulette wheel method. For each tree the probability of being chosen is:

$$p_i = \frac{f_i}{\sum_{j=1}^{n} f_j} \tag{1}$$

where f_i is the fitness function of tree T_i

C. Fitness function

For the evaluation of the tree the Sharpe ratio (Sharpe Ratio), an indicator of the effectiveness of the strategy, introduced by W.F. Sharpe [16] was chosen. It determines the relationship between the profit and the possible risk. It is calculated as:

$$f = \frac{r_p - R_f}{\delta(x)}, \quad (2)$$

where r_p is the average income, R_f is the income with a zero risk, $\delta(x)$ is the standard deviation of income. As a risk-free rate, the value of 2.5% was taken as an analogue of the 10-year US Treasuries.

D. Genetic operators

Genetic operators are constructed by analogy with paper [11] and include crossover and mutation operators.

Due to these two operations, the principles of inheritance and self-adaptivity can be achieved. Taking into account the specifics of the tree-shape data structure, the basis for the modification consists of manipulations with the terminal and non-terminal nodes.

In the process of crossover, two individuals (in the form of decision trees) are selected for crossing. The probability of selection of each one is consistent with the roulette method and it is computed using formula (1). After that copies of the both trees exchange their randomly selected subtrees and get included into the population (together with the original trees). An example of the operator's action is shown in Fig. 2.

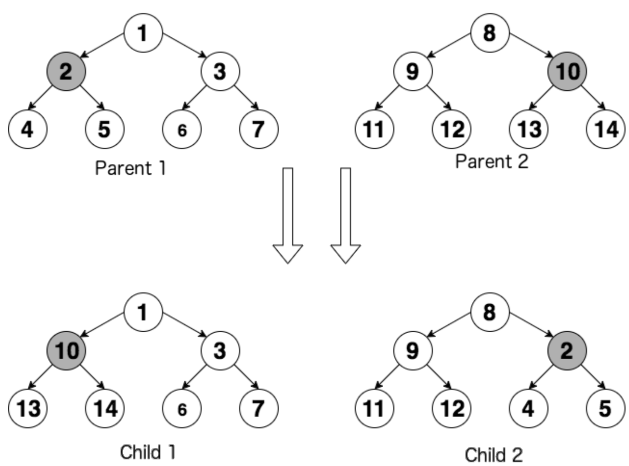

Figure 2. Crossover Operator

The mutation operator is a unary operator, whose purpose is to prevent from premature convergence of the generation to each other. It is based on the same principle as the crossover - two nodes are randomly selected and swapped in the tree.

After these operators are applied, an important step is needed to remove inconsistent nodes. In Fig. 3 a typical case of this kind of errors is presented. As it can be seen, when visiting node *IND1 is LOW* it is already known that this predicate is false, since on the way from it to the root there is node *IND1 is HIGH*. The process of removing excess nodes goes by raising their respective child. In this example, if *IND1 is HIGH* is true, there will be a transition to *IND3 is MEDIUM* (i.e. to the subtree, to which would be a transition in case of falsity of node *IND1 is LOW*).

After creating a new generation, an excess node removal algorithm (Algorithm 1) is run for each tree that had been modified by crossover or mutation. As a result, a tree either retains its structure, or, if an unnecessary predicate is found in a node, the nodes are replaced according to the algorithm suggested below. The output is an optimized tree (*optimizedTree*) that works exactly the same way as the original one.

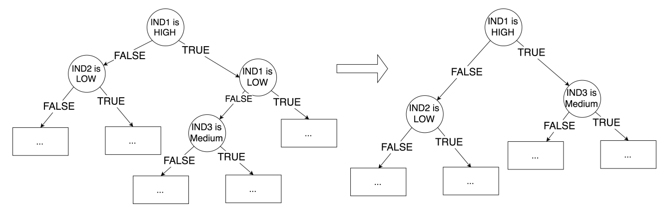

Figure 3. Optimization of tree

As the input the algorithm accepts two nodes which are swapped. In this case the nodes can belong to both different trees and to one tree. So, it can be used both in case of a crossover and a mutation. In this algorithm $replacedNode1$, $replacedNode2$ are the nodes that are swapped around; $RightTree$ is the subtree node in the case of truth of the predicate, $LeftTree$ is the subtree in case of falsity; $Name$ - the name of the indicator in the predicate, $Value$ is its amount; *parent* is the parent of the node.

The estimation of computational complexity is $\mathcal{O}(n \log n)$, where n is the number of nodes in the tree. This complexity allows, in principle, to use a large number of indicators to build a trading system.

E. General algorithm

The general algorithm consists of the steps of the classical genetic algorithm. Iteratively (in 25 steps) new generations (200 individuals) are created, after which their fitness function is calculated and the selection is carried out by means of the roulette method. To a certain number of winners the operators of crossover (35%) and mutation (5%) are applied. The trees that underwent these permutations are optimized by Algorithm 1. The result will be the best tree (by the fitness function value) in the latest generation.

All parameters are chosen empirically to ensure a balance of quality and running time. The algorithm scheme, its parameters and time characteristics are described more in detail in our work [10]. Sometimes sustainability of the result can be required from the genetic algorithm. That is, subsequent start-ups of the system with identical parameters must bring the identical decision tree. In case of three technical indicators, this will require increasing the number of individuals to 800 and for generations to 50.

```
Input   : replacedNode1, replacedNode2
Output: optimizedTree
// Traversal of the children of the changed nodes
TraversalOptimize(replacedNode1);
TraversalOptimize(replacedNode2);
Procedure TraversalOptimize(tree)
    if tree.Left ≠ Null then
        Traversal(tree.Left);
    end
    if tree.Right ≠ Null then
        Traversal(tree.Right);
    end
    FixRepetitions (tree);
/* /* Procedure for checking and removing excess
   nodes                                                  */
Procedure FixRepetitions (node)
    /* Remember the true and false branch of the
       verified node                                      */
    trueWay ⟵ node.RightTree;
    falseWay ⟵ node.LeftTree;
    currentNode ⟵ node.parent;
    /* Going up to the root and checking
       currentNode with each node on the path             */
    while currentNode ≠ Null do
        if currentNode.Name = node.Name then
            if currentNode.Value = node.Value then
                // An identical predicate is found
                if node in currentNode.LeftTree
                 then
                    /* If the node is in the left
                       descendants subtree of the
                       current node, replace it with the
                       false branch.                      */
                    node ⟵ falseWay;
                else
                    /* Otherwise, for the true one        */
                    node ⟵ trueWay
                end
            else
                /* If only the names of the indicators
                   coincide, it is sufficient to check
                   whether the node is in the right
                   descendants subtree of the current
                   one. In this case, the node is
                   obviously false, since the values
                   differ.                                */
                if node in currentNode.RightTree
                 then
                    node ⟵ falseWay;
                end
            end
        end
        /* Going one level up                             */
        currentNode ⟵ currentNode.parent
    end
```

Algorithm 1: Removing excess node

IV. Testing

A. Metrics

In addition to the objective function – the Sharpe ratio, for the analysis of the results, additional metrics were used.

1) Return is a profit rate in percent and it is calculated by trade simulations using historical data. The higher it is, the better.
$$Return = (\frac{FinalBalance}{StartBalance} - 1) * 100, \qquad (3)$$
where $FinalBalance$ is the amount of money after the simulation and $StartBalance$ is the amount of money at the beginning.
2) Max drawdown is the maximum reduction of the account balance value from its local high in percent. It allows to estimate the maximum loss of capital when using this strategy. The lower it is, the better.
3) Trades count is the number of open positions for the period under review. The index is proportional to the possible value of the commission costs.
4) Buy and Hold represents the stock return, which is provided by the market. The essence of this strategy is in the purchase of the instrument at the very beginning and its retention until the end of the given trading period. For a comparison, the profit indices and the Sharpe ratio (B&H Returns, B&H Sharpe) were taken.

B. Indicators

Below there are indicators [1], on which the system builds up its trade recommendations:

1) Relative strength index (RSI), this oscillator compares the number of recent asset price rises with the number of its falls.
2) Stochastic oscillator (STO), shows the position of the current price relative to the price range for a certain period.
3) Average True Range (ATR) is an indicator of the current market volatility.

A greater number of indicators increases the sensitivity of the trading system, which increases the number of transactions.

C. Results of testing

As a testing object, five large IT companies were selected: Citrix Systems (CTXS), Electronic Arts (EA), eBay Inc (EBAY), Intel (INTC), Oracle (ORCL). Consistently for each year a selection of the best trading decision tree and its verification in the next one is carried out. One year has 251 trading signals, each of which corresponds to a certain stock market day.

It should be noted that the time characteristics of our algorithm allow to apply it to the analysis of 5-minute timeframes and higher. However, for liquid shares the greater the timeframe is, the lower the influence of the "noise" component of the market is. This component is

Table I
CTXS results

Year	Sharpe	B&H Sharpe	Return	B&H returns	Max drawdown	Trades Count
2007 Train	2.73	1.26	59.64%	36.16%	6.92%	42
2008 Test	-0.82	-0.73	-20.23%	-35.72%	36.66%	28
2008 Train	1.60	-0.73	26.88%	-35.72%	6.70%	26
2009 Test	-0.05	1.52	1.90%	67.20%	6.57%	16
2009 Train	3.42	1.52	102.75%	67.20%	7.96%	40
2010 Test	1.64	1.34	19.64%	54.37%	4.08%	18
2010 Train	1.82	1.34	64.59%	54.37%	10.23%	15
2011 Test	0.29	-0.07	6.66%	-9.04%	28.50%	14
2011 Train	1.53	-0.07	40.58%	-9.04%	8.58%	14
2012 Test	0.65	0.24	11.06%	5.26%	9.25%	21
2012 Train	1.57	0.24	12.60%	5.26%	2.76%	15
2013 Test	-0.78	-0.32	-3.27%	-8.77%	9.68%	18
2013 Train	2.67	-0.32	37.12%	-8.77%	3.56%	30
2014 Test	1.85	0.16	21.80%	3.66%	2.99%	34
2014 Train	3.84	0.16	46.11%	3.66%	1.51%	37
2015 Test	3.00	0.69	38.48%	19.96%	4.37%	34
2015 Train	1.69	0.69	18.37%	19.96%	1.82%	18
2016 Test	1.16	0.73	10.98%	18.66%	3.88%	35

Table II
EA results

Year	Sharpe	B&H Sharpe	Return	B&H returns	Max drawdown	Trades Count
2007 Train	2.74	0.45	54.25%	11.56%	9.78%	70
2008 Test	-1.30	-1.94	-33.88%	-65.56%	44.81%	38
2008 Train	3.32	-1.94	132.34%	-65.56%	7.19%	54
2009 Test	1.87	0.19	28.07%	1.50%	4.35%	30
2009 Train	2.72	0.19	71.93%	1.50%	8.78%	29
2010 Test	0.54	-0.25	10.76%	-8.73%	9.25%	22
2010 Train	2.78	-0.25	43.72%	-8.73%	4.86%	52
2011 Test	-0.05	0.67	0.11%	23.08%	13.33%	34
2011 Train	1.32	0.67	46.63%	23.08%	20.28%	14
2012 Test	-0.58	-0.95	-16.38%	-27.89%	33.46%	12
2012 Train	1.46	-0.95	24.23%	-27.89%	7.94%	39
2013 Test	1.38	1.30	24.36%	53.26%	4.34%	44
2013 Train	3.62	1.30	58.90%	53.26%	5.01%	57
2014 Test	2.53	2.22	26.16%	97.53%	3.33%	46
2014 Train	5.10	2.22	112.49%	97.53%	4.00%	36
2015 Test	1.46	1.33	32.30%	44.03%	11.97%	41
2015 Train	4.59	1.33	95.09%	44.03%	4.00%	74
2016 Test	0.44	0.71	8.71%	19.65%	12.76%	52

caused by unpredictable actions of large players, as well as by a random combination of actions of many trading robots and traders. Vulnerability of the trading system to such a random component leads to premature signals. Therefore, similarly to works [7] and [11], testing is carried out on daily historical data.

The results are shown in Tables I - V. The lines with the better system performance than the B&H strategy are highlighted in grey.

V. Future work

Obviously, the efficiency of the constructed trading strategies substantially depends on the set of technical indicators and their parameters. In this regard, it is necessary to investigate the applicability of other technical indicators and to automate the selection of optimal parameters of the indicators. Also in the future it is planned to expand the set of nodal predicates. At the moment indicator values are only able to belong to the level but later it is planned to add the relationships between several indicators within a predicate. An important area is the study of applicability of the approach for BUY/SELL/HOLD signals.

Table III
EBAY results

Year	Sharpe	B&H Sharpe	Return	B&H returns	Max drawdown	Trades Count
2007 Train	1.90	0.36	47.86%	9.01%	9.78%	39
2008 Test	-0.77	-1.42	-26.94%	-52.55%	40.55%	24
2008 Train	2.70	-1.42	74.23%	-52.55%	6.64%	33
2009 Test	3.38	1.24	123.44%	55.25%	8.52%	37
2009 Train	2.92	1.24	78.65%	55.25%	8.14%	36
2010 Test	0.83	0.62	11.11%	16.91%	6.39%	12
2010 Train	1.49	0.62	24.08%	16.91%	6.39%	27
2011 Test	0.85	0.24	12.15%	4.68%	7.84%	7
2011 Train	1.45	0.24	25.98%	4.68%	8.23%	30
2012 Test	1.72	1.73	21.82%	58.61%	4.96%	20
2012 Train	2.23	1.73	55.22%	58.61%	7.50%	55
2013 Test	-1.57	0.07	-21.31%	1.61%	27.20%	31
2013 Train	1.36	0.07	15.80%	1.61%	2.86%	39
2014 Test	0.68	0.20	8.71%	4.62%	6.12%	37
2014 Train	3.20	0.20	28.06%	4.62%	3.05%	28
2015 Test	-0.38	-0.69	-28.73%	-45.02%	48.15%	25
2015 Train	3.71	-0.69	55.73%	-45.02%	2.98%	48
2016 Test	0.14	0.45	3.32%	11.61%	18.19%	16

Table IV
INTC results

Year	Sharpe	B&H Sharpe	Return	B&H returns	Max drawdown	Trades Count
2007 Train	1.49	1.00	35.18%	26.54%	11.73%	36
2008 Test	-0.57	-0.93	-24.27%	-39.51%	38.93%	19
2008 Train	3.28	-0.93	68.39%	-39.51%	2.63%	45
2009 Test	1.35	0.90	18.79%	30.67%	5.21%	49
2009 Train	4.09	0.90	94.73%	30.67%	3.22%	36
2010 Test	-1.11	0.01	-7.79%	0.04%	13.79%	11
2010 Train	1.55	0.01	20.00%	0.04%	4.01%	12
2011 Test	0.66	0.55	6.92%	13.98%	1.16%	7
2011 Train	1.23	0.55	24.10%	13.98%	9.28%	41
2012 Test	0.62	-0.86	8.90%	-14.49%	8.15%	30
2012 Train	0.64	-0.86	7.47%	-14.49%	6.22%	47
2013 Test	0.48	0.85	6.76%	17.98%	6.00%	30
2013 Train	2.73	0.85	31.64%	17.98%	4.69%	30
2014 Test	3.53	1.59	51.25%	39.15%	4.03%	38
2014 Train	3.53	1.59	51.25%	39.15%	4.03%	38
2015 Test	3.11	-0.10	45.35%	-2.03%	2.43%	42
2015 Train	3.11	-0.10	45.35%	-2.03%	2.43%	42
2016 Test	-0.35	0.30	-0.91%	6.75%	7.26%	17

Table V
ORCL results

Year	Sharpe	B&H Sharpe	Return	B&H returns	Max drawdown	Trades Count
2007 Train	2.38	0.96	35.94%	26.42%	4.23%	48
2008 Test	0.12	-0.38	2.86%	-20.45%	15.15%	42
2008 Train	2.74	-0.38	46.03%	-20.45%	6.35%	43
2009 Test	4.25	1.04	94.30%	32.90%	2.49%	46
2009 Train	4.30	1.04	118.95%	32.90%	3.29%	68
2010 Test	1.82	0.99	29.85%	24.37%	5.00%	58
2010 Train	1.58	0.99	21.62%	24.37%	3.68%	34
2011 Test	-0.27	-0.50	0.40%	-16.88%	6.75%	12
2011 Train	1.30	-0.50	22.14%	-16.88%	14.05%	40
2012 Test	0.73	1.19	8.22%	25.65%	4.98%	35
2012 Train	1.94	1.19	34.01%	25.65%	8.55%	32
2013 Test	-0.38	0.39	-5.24%	8.61%	17.23%	30
2013 Train	4.10	0.39	35.41%	8.61%	1.92%	43
2014 Test	3.89	0.89	34.14%	18.79%	1.98%	39
2014 Train	4.22	0.89	42.84%	18.79%	2.19%	49
2015 Test	2.19	-0.85	25.82%	-14.37%	5.39%	42
2015 Train	1.97	-0.85	15.50%	-14.37%	2.16%	29
2016 Test	-1.48	0.33	-8.11%	6.92%	9.41%	16

VI. Conclusion

As it can be seen from the presented examples, on average in 6 out of 9 cases the integral Sharpe ratio was better with our system than with the market. At the same time, the maximum loss of capital was 48% against 65% for a simple investment. This means that there are stocks and periods for which this approach can be effective. The number of entries into the long position varies from 58 to 7, which indicates a low sensitivity of the rules received. This may be due to the fact that a period of 1 year was taken as a sliding interval to determine the level ranges. Reducing this interval leads to an increase in the speed of the system reaction, but more false signals emerge.

It should be noted that during the training this trade system is adjusted to a certain ratio of trend and consolidation movements of the share price. If in the next period this ratio changes, then the trader can expect whether profit or loss. This fact is inherent in most algorithmic trading strategies based on methods of the technical analysis.

References

[1] M.J John, "Technical analysis of the financial markets: a comprehensive guide to trading methods and applications (2nd ed.)", New York Institute of Finance, 1999.

[2] R. Morta, E. Dadios "Proposed system for predicting Buy, Hold and Sell recommendations for a publicly listed Philippine company using computational intelligence", *TENCON 2015-2015 IEEE Region 10 Conference*, 2015, pp. 1-8.

[3] S. Yodphet, S. Rimcharoen, and Leelathakul, "LARG: Loss avoidance technical trading rules using genetic algorithm. In Knowledge and Smart" *Technology (KST), 2016 8th International Conference*, 2016, pp. 33-38.

[4] F. Giacomel, R. Galante, and A. Pereira. "An Algorithmic Trading Agent based on a Neural Network Ensemble: a Case of Study in North American and Brazilian Stock Markets.", *In Web Intelligence and Intelligent Agent Technology (WI-IAT)*, 2015, IEEE/WIC/ACM International Conference on (Vol. 2, pp. 230-233). IEEE.

[5] B.A. Sheldon, "Binary Decision Diagrams", *IEEE Transactions on Computers*, pp. 509–516, 1978.

[6] L. Rokach and O. Maimon, "Data mining with decision trees: theory and applications", 2014.

[7] F. X. Satriyo D. Nugroho, T. Bharata Adji, S. Fauziati, "Decision support system for stock trading using multiple indicators decision tree", *The 1st International Conference on Information Technology, Computer, and Electrical Engineering*, 2014.

[8] C. N. Ochotorena, C. A. Yap , E. Dadios and E. Sybingco, "Robust stock trading using fuzzy decision trees", *Computational Intelligence for Financial Engineering and Economics (CIFEr)*, pp. 1-8, 2012.

[9] R. Dash and P.K. Dash, "A hybrid stock trading framework integrating technical analysis with machine learning techniques", *The Journal of Finance and Data Science*, pp. 42-57, 2016.

[10] D. Iskrich, D. Grigoriev "Generating Long-Term Trading System Rules Using a Genetic Algorithm Based on Analyzing Historical Data", *Proceedings of the International FRUCT Conference on Intelligence, Social Media and Web*, 2017

[11] J. Potvin, P. Soriano, M. Vallée "Generating trading rules on the stock markets with genetic programming", *Computers & Operations Research*, 2004.

[12] Iba, Hitoshi, and Claus C. Aranha "Practical Applications of Evolutionary Computation to Financial Engineering", Springer, 2012.

[13] Nur Aini Rakhmawati, Erma Suryani "DECISION FOR BUYING AND SELLING STOCK WITH DECISION TREE ALGORITHM", Sepuluh Nopember Institute of Technology.

[14] Al-Radaideh, A. Qasem, Aa Assaf, and Eman Alnagi. "Predicting stock prices using data mining techniques.", *The International Arab Conference on Information Technology* (ACIT'2013), 2013.

[15] A. Ghandar, Z. Michalewicz, M. Schmidt, T. To, and R. Zurbrugg, "Computational Intelligence for Evolving Trading Rules", *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION 1*, pp. 71-86, 2009.

[16] W.F. Sharpe, "The sharpe ratio", *The journal of portfolio management*, 1994, pp. 49-58.