Evaluation of Fitness Functions for Evolved Stock Market Forecasting

J. F. Nicholls*, A. P. Engelbrecht, and K. M. Malan

Department of Computer Science, School of Information Technology,
University of Pretoria,
Pretoria, 0002, South Africa
**E-mail: nicholls.jason@gmail.com*
cirg.cs.up.ac.za

This article investigates the impact of different fitness functions on the investment return of a genetically evolved trader. Four different fitness functions are studied and compared. Using historical market data, a population is trained by a simple genetic algorithm using crossover, mutation, elitism and creationism. Four genetic algorithms have been used to evolve an agent to trade, where each genetic algorithm used a different fitness function. The best individual from each evolved population is compared using an out-of-sample data set. Results show a significant difference in performance between the four fitness functions. Populations evolved using the total area under the asset value graph as a function of fitness, produced a higher return on investment on volatile stocks than those populations evolved using the final asset value as a function of fitness. Agents subjected to complexity penalization (Ockham's razor) did not produce significantly better results than agents that were not penalized.

Keywords: Evolutionary Computing; Fitness Function; Market; Forecast; Technical Analysis.

1. Introduction

Technical analysis has become a widely used market analysis tool made famous by the editor and founder of the Wall Street Journal, Charles Dow.[1,2] Technical Analysis employs statistical functions on current and historical stock data to find the current trend or signal a change in the trend. A change in the trend may indicate a buy or sell signal. Technical analysis requires specific parameters that may change for any given stock. Each parameter has an effect on the forecast ability of the function. The number of combinations of parameters and functions creates an NP complete problem. This large search space is ideal for stochastic search algorithms such

as evolutionary algorithms.[3,4]

Schoreels *et al.*[5] showed that a simplistic genetic program using technical analysis performs comparably to investment funds run by human professionals. In 2007 Papadamou *et al.*[6] successfully produced an evolved trading tool called GATradeTool that out-performed two leading stock tools MetaStock and FinTradeTool. Lam,[7] Li and Tsang[8] showed that a genetic algorithm in-conjunction with technical analysis could out-perform a buy and hold strategy. Mahfoud and Mani[9] showed that a genetic algorithm using technical analysis could out-perform a neural network and a buy and hold strategy. Ghandar *et al.*[10] showed that genetic algorithms and technical analysis could be used on a portfolio of stocks to successfully out-perform the benchmark indices. Different approaches are used to calculate the fitness of individuals in evolutionary algorithms. Schoreels *et al.*[5] implemented an accumulated return on investment (AROI) as a fitness function, while Lam[7] and Papadamou *et al.*[6] used the final return on investment (ROI). Mahfoud and Mani[9] introduced a more complex fitness function, using a credit system that penalizes complexity and poor performance while rewarding generality. Ghandar *et al.*[10] implemented the AROI but penalized individuals for the number of active rules.

The aim of this paper is to evaluate these different fitness functions. Four functions are used: ROI, AROI, ROI with a complexity penalty and AROI with a complexity penalty. These four fitness functions are tested against a selection of stocks to see which fitness function yields the greatest profit. The rest of the paper is divided into three sections. The first section covers the design of the genetic algorithm. The second section highlights the experimental set-up and results, followed by the conclusion.

2. System Design

A simple genetic algorithm is used to determine the parameter values of thirteen technical analysis functions. Functions may be enabled or disabled by the genetic algorithm. Using the enabled technical analysis functions, each individual examines the current market information (volume, closing, highest and lowest price of the day) to determine an action (buy, hold or sell). The action is executed by a virtual banker. Each purchase and sale incurs charges (brokerage fee: 0.006% - min of R70, strate fee: R10.92, insider trading fee: 0.000007% and Value Added Tax: 14%, where R is the symbol for the South African Rand). All individuals are rated using one of four fitness functions. The higher the fitness value, the higher the probability an individual has of being selected for crossover. The best individual

survives to the next generation. New individuals are introduced during each generation. After a set number of generations the evolutionary process is stopped. The best individual from a population is then tested against an out-of-sample test set.

2.1. *Genetic Algorithm Representation and Operations*

Genotypic information is encoded such that each individual (genome) is made up of an array of nodes (chromosome). Each node (gene) is linked to a technical analysis function listed in Table 4. A node has a list of variables (DNA) used as input to the function. The functions return a value that may indicate a change in the trend. This change results in a buy, hold or sell action (encoded as a floating point number). The actions of all the enabled nodes are summed resulting in a single action for any individual.

Four simulated evolutionary operations are implemented: elitism,[3,4] creationism, crossover[3,4] and mutation.[3,4] Elitism is the selection of the best individual within a generation and transferring that individual un-altered to the next generation. Creationism introduces a specific number of new randomly created individuals into the next generation. Crossover selects a set of the best individuals, selected from a randomly selected set of individuals. Two individuals are randomly selected as the parents from this set for crossover. Nodes from the parents are randomly selected at a specified selection probability to take part in crossover. A randomly selected variable within each of the selected nodes of one parent is swapped with the corresponding variable in the other parent, resulting in two new offspring that share the genotypic information of their parents. Based on a mutation probability, offspring are mutated by altering a single variable within a single node. The variable type will denote the change. A number becomes a new randomly selected number and a boolean state is switched.

2.2. *Fitness Functions*

The fitness function represents the probability of, or propensity for, survival of an individual or population.[3] Those individuals that maximize profit will have a greater probability of genetic survival. Profit is generally seen as the final ROI of the stock. Schoreels *et al.*[5] argued that by using the final ROI, the genetic algorithm may evolve an over-fitted individual that will perform poorly on unseen data. Schoreels *et al.* therefore suggested to rather use an AROI. This conflict produces two potential fitness functions: ROI (F_1) or AROI (F_2). Ghandar *et al.*[10] suggested that the more complex the

make-up of an individual the higher the chance of over-fitting the training data. They suggested penalizing those individuals that have a higher complexity (Ockham's razor). Using the two fitness functions above (F_1 and F_2) and adding a complexity penalization adds two additional fitness functions, namely AROI with penalty (F_3) and ROI with penalty (F_4). The four fitness functions are expressed in Table 1.

3. Experimental Set-up and Results

Fifty populations with 100 individuals in each were evolved using 50 unique random seeds. Ten parents were selected from each iteration to create 89 offspring. Fifty percent of the function parameters were crossed over. One out of 2 individuals were selected for mutation. In addition to the 89 offspring, 10 randomly created individuals were added to the new generation. The best performing individual from the current generation was carried over to the new generation. The result was a new generation with 100 individuals. Populations are evolved over 50 iterations. The process is repeated for each fitness function defined in Table 1. The entire experiment was repeated for a number of different stocks collected from different sectors within the Johannesburg Stock Exchange (JSE), namely Anglo American (AGL), ABSA (ASA), BHP Billiton (BIL), Gold Fields (GFI), MTN (MTN), Murray and Roberts (MUR), Remgro (REM), Richmond (RCH), Standard Bank (SBK), SASOL (SOL) and Satrix 40 Index Fund (STX40). Each genetic algorithm was trained on data from the 3$^{\text{rd}}$ of April 2007 to the 22$^{\text{nd}}$ of January 2008 which included 200 trading days. Testing was done on data from the 22$^{\text{nd}}$ of January 2008 to the 18$^{\text{th}}$ of June 2008 which included 100 trading days.

The results from the 11 stock runs are depicted in Table 2. Each run started with an initial cash value of R100,000. Trades were made on the stock market using a virtual banker. At the end of the 100-day test period all stocks were sold resulting in the final asset value. The table shows the stock code followed by the mean final asset value of 50 runs using one of the four fitness functions (highest means are marked with an asterisk). The buy and hold column shows the final asset value after purchasing the stock at the start of the period and selling it at the end. The final column shows the p-value result of a Kruskal-Wallis statistical test comparing the mean values of the four fitness function tests. A 95% significance has a 0.05% chance of rejecting the null hypothesis. This results in an α value of 0.05. Using Kruskal-Wallis, a significant difference (p-values less than the α value is obtained) in 6 (AGL, ASA, MUR, RCH, SOL, STX40) of

the 11 experiments. Further investigation found that stocks with a higher significant difference had higher volatility.

Stocks which showed a significant difference using the Kruskal-Wallis test were analysed further using the Mann and Whitney U test. Results are shown in Table 3. Bonferroni correction was made for the Mann and Whitney U tests resulting in an α value of 0.0125. Those stocks that showed a significant difference in the Kruskal-Wallis test, showed in the Mann and Whitney U test that F_1 and F_4 are not significantly different and F_2 and F_3 are not significantly different. This means that the use of Ockham's razor to penalize complexity had no significant effect on the performance of stocks. Most of the stocks that showed a significant difference (AGL, MUR, RCH, SOL and STX40) performed better in the case of F_2 or F_3. Only the ASA stock performed significantly better in the case of F_1 and F_4.

4. Analysis and Conclusion

Stocks with a lower volatility showed no significant difference in the use of different fitness functions. This is expected as the stock does not change a lot resulting in less opportunity for large profit or loss. The stocks that showed a significant difference showed that using the area under the asset graph (F_2, F_3) as a fitness function in most cases out-performed those stocks that used the final asset value (F_1, F_4). Using the final asset value as a function of fitness on a volatile bullish stock would result in an evolved trader that capitalises on the ups but makes larger losses on the downs. If the stock ends in the green the trader would appear to have done well. If the stock was bearish the trader would have lost more. Using a function of area forces a trader to remain constant. Any great gains or losses are averaged and reflected in the final fitness value. Fitness functions penalizing complexity (F_3, F_4) did not produce significantly different results to those functions that did not (F_1, F_2). This could mean that the functions were not complex enough for a complexity penalization or the penalty value was not high enough. Further work is required to test if different penalization values would produce a significant difference. In conclusion a fitness function that incorporates the entire performance of an evolved trader, such as the accumulated asset value produces a significantly better performing trader on volatile stocks than a fitness function that focuses on the final result or final asset value of a trader.

References

1. A. Cowles, *Econometrica* **1**, 309 (1933).

2. W. I. King, *Journal of the American Statistical Association* **29**, 323 (1934).
3. D. B. Fogel, *Evolutionary Computation, Toward a New Philosophy of Machine Intelligence* (IEEE, 2006).
4. A. P. Engelbrecht, *Computational Intelligence, An Introduction* (Wiley, 2003).
5. C. Schoreels, B. Logan and J. M. Garibaldi, Agent based genetic algorithm employing financial technical analysis for making trading decisions using, historical equity market data, in *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, (3)2004.
6. S. Papadamou and G. Stephanides, *Mathematical and Computer Modelling* **46**, 189 (2007).
7. S. S. Lam, *Evolutionary Computation* **1**, 410 (2001).
8. J. Li and E. P. K. Tsang, Improving technical analysis predictions: An application of genetic programming., in *Proc. Florida Artificial Intelligence Research Symposium, USA*, 1999.
9. S. Mahfoud and G. Mani, *Applied Artifical Intelligence* **10**, 543 (1996).
10. A. Ghandar, Z. Michalewicz, M. Schmidt, T. To and R. Zurbrugg, *accepted for IEEE Transactions On Evolutionary Computation 2008* (2007).

Table 1. Fitness Functions implemented

Function	Formula	Description
F_1	$ROI = AssetValue_i$	Final asset value
F_2	$AROI = \sum_{t=1}^{n} AssetValue_{i-t}$	Accumulated asset value
F_3	$AROI = (\sum_{t=1}^{n} AssetValue_{i-t}) - Pen\%$	F_2 with complexity penalty
F_4	$ROI = AssetValue_i - Pen\%$	F_1 with complexity penalty

Note: i is end of the period. n is the number of days and Pen is the penalty percentage. (Each enabled node increments the penalty by 0.00001)

Table 2. Comparison of Fitness Functions using Kruskal-Wallis

Stock	F_1 (Mean)	F_2 (Mean)	F_3 (Mean)	F_4 (Mean)	Buy & Hold	Kruskal-Wallis
AGL	119,500	130,000	130,800*	121,200	136,373	3.848×10^{-10}
ASA	99,160	92,220	92,420	99,900*	82,654	4.237×10^{-11}
BIL	142,100	148,700	151,100*	146,000	160,094	0.09327
GFI	94,930	92,020	91,170	95,220*	77,787	0.4728
MTN	121,200	122,100*	120,600	117,500	120,294	0.2799
MUR	99,770	103,800*	102,000	102,100	100,449	0.009018
REM	107,700	107,500	108,800*	106,600	113,929	0.1475
RCH	116,400	119,900	120,100*	117,100	121,958	0.004632
SBK	88,640*	87,400	87,100	87,210	82,539	0.4147
SOL	128,000	136,200	139,100*	127,300	146,443	9.978×10^{-7}
STX40	111,500	117,500*	116,600	112,500	126,337	0.002105

Table 3. Comparison of each Fitness Function using Mann and Whitney U

Stock	F_1 vs F_2	F_1 vs F_3	F_1 vs F_4	F_2 vs F_3	F_2 vs F_4	F_3 vs F_4
AGL	9.783×10^{-7}	1.754×10^{-7}	0.5786	0.4099	1.339×10^{-5}	2.849×10^{-6}
ASA	4.781×10^{-6}	3.948×10^{-7}	0.5876	0.7774	6.143×10^{-7}	3.433×10^{-8}
MUR	0.001465	0.1477	0.07692	0.02534	0.1410	0.539
RCH	0.008196	0.0008165	0.4319	0.4503	0.1536	0.03431
SOL	0.003606	9.301×10^{-5}	0.8013	0.1807	0.0001887	3.273×10^{-6}
STX40	0.0009033	0.007346	0.6537	0.7069	0.01039	0.04292

Table 4. Technical Analysis Functions

Node	Name	Function
1	Bollinger Bands	$+BB(n) = SMA(n) + MD(n)$ $-BB(n) = SMA(n) - MD(n)$
2	Simple Moving Average	$SMA(n) = \frac{P_t + P_{t-1} + \cdots + P_{t-n+1}}{n}$
3	Exponential Moving Average	$\alpha = 1 - \frac{2}{n+1}$ $EMA(n) = \frac{\alpha^n P_t + \alpha^{n-1} P_{t-1} + \cdots + \alpha^2 P_{t-n+2} + \alpha P_{t-n+1}}{\alpha^n + \alpha^{n-1} + \cdots + \alpha^2 + \alpha}$
4	Weighted Moving Average	$WMA(n) = \frac{\omega P_t + (\omega-1) P_{t-1} + \cdots + 2P_{t-n+2} + P_{t-n+1}}{\omega + (\omega-1) + \cdots + 2 + 1}$
5	Relative Strength Index	$TG = \sum_{t=1}^{n} (P_{t+1} - P_t)$ for all t where $P_{t+1} - P_t > 0$ $TL = \sum_{t=1}^{n} (P_t - P_{t+1})$ for all t where $P_t - P_{t+1} > 0$ $RSI(n) = 100 - \frac{100}{1 + \frac{TG/n}{TL/n}}$
6	Money Flow Index	$MF = P_t * V_t$ $MFI(n) = 100 * \frac{+MF}{+MF + -MF}$
7	K-Line	$KLine(n) = \frac{P_t - lowest}{heighest - lowest} * 100$
8	D-Line	$DLine(n) = SMA$ of $KLine$ $DLine(n) = \frac{KLine_t + KLine_{t-1} + \cdots + KLine_{t-n+1}}{n}$
9	On Balance Volume	$OBV_t = OBV_{t-1} + \begin{cases} V & \text{if } P_t > Price_{t-1} \\ 0 & \text{if } P_t = Price_{t-1} \\ -V & \text{if } P_t < Price_{t-1} \end{cases}$
10	Accumulative Distribution Index	$CLV(t) = \frac{(P_{close} - P_{low}) - (P_{high} - P_{close})}{(P_{high} - P_{low})}$ $ADI(t) = ADI_{t-1} + V_t CLV_t$
11	Rate of Change	$ROC(n) = \frac{P_t - P_{t-n}}{P_{t-n}} * 100$
12	Commodity Channel Index	$\theta = \frac{\sum_{t=1}^{n} \lvert SMA_t - P_t \rvert}{n}$ $CCI(t) = \frac{1}{0.015} \frac{P_t - SMA(n)}{\theta}$
13	Moving Average Convergence or Divergence	$MACD(t) = EMA(12) - EMA(26)$

Note: t is the day of trade, P is the closing price, n is the number of previous days. V is the Volume, MD is the mean deviation.