Optimizing an Algorithmic Trading Strategy Using DEAP

Lukas Jakstas
COSC 420
University of Tennessee
Knoxville, TN
ljakstas@vols.utk.edu

Andrew Welden
COSC 420
University of Tennessee
Knoxville, TN
awelden2@vols.utk.edu

***Abstract*—Recent coverage, popularity, and news concerning the stock market and cryptocurrency have introduced more people than ever before to the stock market. However, it is estimated that 70% of people lose in the market whether they are new or experienced [14]. The market is an unforgiving place. A tested trading strategy would in theory be successful in the market and provide consistent returns. Evolutionary and Genetic algorithms can help increase success in financial markets and investment portfolios. This paper presents one such algorithm built using DEAP (Distributed Evolutionary Algorithms in Python) to optimize trading parameters. Performance of the algorithm was evaluated using S&P 500 data from 2009 - 2015. Parameters selected included Simple Moving Average Crossover (SMAC) and Relative Strength Index (RSI). Our Results suggest genetic algorithms can in fact be used to train successful trading agents.**

Keywords— Machine Learning, Stock Market, Genetic Algorithm, Evolutionary Strategies, Technical Trading, Cryptocurrency

I. Introduction

Stock and bond investment portfolios play an important role in retirement planning for millions of Americans. A recent Gallup poll reports over 60% of working Americans under 50 do not believe Social Security will be able to pay practical benefits when they reach retirement age [15]. As public trust in Social Security falls, reliance on a stable market becomes ever-important for the future. However, the stock market is an unforgiving place. Statistics range, but it is estimated that up to 70% of people lose in the stock market - whether new to the market or experienced [14].

It is our goal to create an optimal trading strategy to provide consistent returns and avoid unforeseen financial crashes. Using technical trading rules to analyze parameters of momentum, volatility, trends, and volume, we will first develop a proper strategy that has potential to perform well in the market. Then, we need to optimize this strategy to not only perform well in historic testing, but also in the future. To do this, we will be using Evolutionary Algorithms - more specifically, a genetic algorithm.

Evolutionary algorithms use concepts of biological evolution inspired by Charles Darwin's theory of evolution. Evolutionary algorithms use a computational version of the mechanisms found in nature's process of evolution such as reproduction, mutation, selection, recombination/crossover, etc. In this paper we will discuss and use the bio-inspired genetic algorithms, a subset of evolutionary algorithms, to optimize our trading algorithm.

Ideally, the algorithm will produce consistent returns rather than periods of high and low margins - market stability being the central intent. Because the technical analysis tool used in this experiment can be applied to all trading markets, the algorithm will not only work within the bounds of the stock market, but it will also predict price action in bond, cryptocurrency, FOREX, futures, and index markets. Widespread use of the algorithm would, theoretically, reduce harsh movement in price action, Cutting out both sudden surges and quick crashes in favor of sustained consistent growth that we ultimately already see in long term investment strategies using mutual funds and Exchange Traded Funds. In the future, we'd like to extend this from being isolated in the S&P 500 to different markets as well.

II. Motivation

Creating a successful-performing algorithm as well as optimizing it will provide for consistent and reliable equity growth - a possible alternative to the reliance of questionable and inefficient social security for those in need of retirement insurance. This algorithm could possibly provide a stable, privately managed retirement and sense of comfort from possible financial crashes in an unpredictable time.

We have selected genetic algorithms because of the generational tournament selection and mutation and crossover will ensure not only proper evolution, but a diverse one as well. Optimizing the use of these parameters will generate a buy or sell signal based on the outlook produced by the program consistently throughout different market trends.

Throughout this paper and research we want to conclude whether or not genetic algorithms can optimize and manage trading strategy performance in various markets. We found this topic interesting because we recognize the problem facing our generation when it comes to retirement and the ominous fiscal state of the Federal Social Security System. We selected this question in particular to discover if genetic algorithms can assist in said planning, easing the stress many of our age feel.

III. Related Work

A plethora of similar studies have been performed in the past analyzing different indicators and parameters, many of which show promise and progress toward the end goal. The difference between past research and this proposal is that prior work has been conducted almost solely to increase corporate or personal profits in the stock market, while our program could certainly be used in this may, we intend to use the idea to increase market stabilization and reduce risk for all investors.

In refer. (Subramanian, 2006) Used a modular programming approach to the problem as a way of simplifying the complexity of the algorithm. This technique allows the developer to take the product apart in a way and work on separate components individually without affecting functionality of other components. Iterative development processes like these offer a higher assurance of safety: "Combinations of simple interpretable pieces, such a structure could enable the identification of specific transferable features in the strategy that contribute to the observed success." [19].

Subramanian discusses technical trading rules relaying that up to 90% of foreign exchange traders in London use some form of technical trading rules in daily practice. As such, the popularity of the rules flies in the face of theoretical objections arising from the efficient markets hypothesis, which asserts that stock markets are random processes lacking in systematic trends that may be exploited by mechanized rules. However, with the emergence of high frequency finance it has been demonstrated that the efficient market hypothesis is not entirely valid in the short time scales at which autonomous agents are able to operate." [19] He also notes that simplicity of these rules does not necessarily mean that there is no theoretical justification for their use, and that their profitability can be "rigorously characterized" [19]. Successful use of these indicators does, however, vary depending on the market environment, and that they cannot be successfully implemented in isolation. Rules stack in a way and complement each other in certain circumstances.

Lastly, Subramian recognizes human judgment when it comes to stock trading. Writing, "A key aspect of human judgment is the way we handle asymmetry between upside and downside risks. Traders might welcome the occasional windfall profits but they certainly want to avoid bankruptcy at all costs; and they act in such a way that these preferences are satisfied." [19]

In refer. (Chen, 2022) discusses important assumptions made in use of technical analysis to trade. The first assumption is the market's adherence to General Equilibrium Theory, an idea originally developed by French economist Leon Walras in the late 19th century. The theory proposes that supply and demand interact and tend toward a balance in an economy of multiple markets working at once. This balance of competing levels of supply and demand in different markets ultimately creates a price equilibrium [9]. The second assumption is the idea of the rational economic man. A construct of Adam Smith, an 18th century economist. The rational-economic man makes decisions based on rational analysis of desired outcomes and acts in his own rational self-interest [15]. With these assumptions in mind, Chen approaches the issue with agent-based computational economics.

Chen covers use of agent based computational economics in his work, noting "ACE is a bottom-up approach that uses computational techniques to simulate economic environments. To explain financial phenomena at a macroeconomic level, agent based approaches set specific characteristics of an economic environment, referred to as agents, on a microeconomic level."
Three types of agents." [15] these three types of agents are given below:

1. Fundamental Traders
    a. Focuses on company specific events and news to make buy and sell decisions.
    b. Fundamental Traders believe that expected price of the stock is decided by the dividend discount model [15].
2. Noise Traders
    a. Irrational investors that do not adopt any common stock trading methodologies, and choose to buy and sell based on 'hype' or popularity/trend of a certain stock.
3. Technical Traders
    a. Technical traders use historical market data to analyze current price action and forecast the trend.
    b. Indicators used by technical traders are based on various mathematical formulae that take price and volume into account.

In refer. (Allen 1995) Notes the limitations of Evolutionary algorithms and their application to economics. Computation time presents itself as the first limitation, as training generations of agents takes significant CPU resources. "Maintaining a population of genetic structures leads to an increase in execution time, due to the number of times the objective function must be evaluated." [1] The second limitation mentioned by Allen is the issue of ambiguity. "Since evolutionary algorithms embody very little problem-specific knowledge, they are unlikely to perform better and can be less efficient than special-purpose algorithms in well-understood domains." [1]. As a result, machine learning and evolutionary algorithms are always the second best approach to solving problems.

In refer. (Chen 2002) considers the double auction mechanism defining it as "A process of buying and selling goods with multiple sellers and multiple buyers. Potential buyers submit their bids and potential sellers submit their ask prices to the market institution, and then the market institution chooses some price p that clears the market" [3] Using this process allows the trader to narrow the gap between real and artificial markets. In turn, results from an algorithm working on a simulated market can be easily compared to the behavior of real market data.

In refer. (Conti, 2020) discusses the use of technical indicators in algorithmic trading. He explores four types of indicators: trend, momentum, volatility, and volume based indicators. Because each type of indicator examines a different

parameter of the market, the most reliable results come from incorporating all of them into the genetic algorithm [5].

1. Momentum Indicators
    a. Relative Strength Index (RSI)
        i. Developed by Wilder, the relative strength index (RSI) is a momentum indicator. RSI compares the magnitude of recent gains with the magnitude of recent losses, and represents the relationship as an integer value between 0 and 100. A gain or loss for each day is the difference between the opening and [20]
2. Volatility Indicators
    a. Modified Sortino Ratio (MSR)
        i. Modified Sortino ratio (MSR) rewards consistency and profitability. It penalizes negative volatility without penalizing positive volatility [20]
3. Trend Indicators
    a. SimpleMoving Average (SMA)
        i. The simple moving average is a type of trend indicator. The 10-day simple moving average is the average closing price over a 10-day period [20]
4. Volume Indicators
    a. Volume Trend
        i. Volume can be used as an indicator that complements other types of indicators. When other indicators show upward movement, volume decreasing may indicate that the current movement is not sustainable. A sudden extreme increase in volume can indicate that a stock may be overheated, and thus headed for a sharp reversal [20]

In refer. (Bonde, 2017) discusses use of the Sigmoid Function as a means of classifying input into classes. Used when enough detailed input information is not available to make a confident prediction, the function classifies the price of the specific stock into an increasing or decreasing class. The specific function used in [2] is given below.

$$P(t) = \frac{1}{1+e^{-1}} \qquad (5)$$

Bonde concluded that different varying activation functions for classification may have been helpful, rather than using the one sigmoid function.

Bonde also examines Creep Mutation in his work. This is a way of introducing variability into the model. The creep mutation works by adding a small value to each gene with probability p. The selection method used to select the population is roulette wheel selection. In this method the fitness assigned to each individual is used for the selection process. This fitness is used to associate a probability selection with each individual. [2] given as below:

$$P_i = \frac{f_i}{\sum_{j=1}^{N} f_j} \qquad (6)$$

In refer. (Boyd, 2019) explains feature importance as a tool used to describe which inputs are the most helpful in making an accurate prediction. Knowing which parameters give the most useful output allows for weighted selection, and allows the genetic algorithm to more finely tune its input.

Backtesting was also noted in Boyd's approach, pointing out that running multiple backtests over the same data provides a path for forward information to leak into the model development through the developers actions, and the results of the backtest should be deflated with each backest [12]. He advises a better way to test over historical data is to randomly select portions of the data for training and testing.

In refer. (Simões, 2010) approach takes advantage of Simple Moving Average Crossover (SMAC), as we ultimately decided to as well. With this indicator two ranges of moving averages are selected, one long term and the other short term. A buy signal is produced when the short term moving average crosses over (on top of) the long term moving average. A sell signal is produced in the opposite scenario, when the long term moving average line crosses over the short term. An equation for calculating these buy and sell points is given below.

$$SMAC(d) = \frac{1}{s}\sum_{t=d-s+1}^{d} P(t) - \frac{1}{l}\sum_{t=d-l+1}^{d} P(t) \qquad (8)$$

The agent will buy when the equation is equal to a number greater than zero and sell when it is less than zero. An output of zero will indicate that the agent should hold.

Moving Average Derivate (MAD) was another indicator that Simões examined in his work, although we decided not to use this, its influence is interesting to note. MAD is calculated by subtracting the value of the current Moving Average with the value of the Moving Average in the previous day. Where n is the time period used to calculate the MA and g is the distance between the two days to calculate the secant (the original strategy consists of a fixed g with value 1). Below is the formula is given

$$MAD\ (d) = \frac{\sum_{t=d-n+1}^{d} P(t) - \sum_{t=d-n+1}^{d} P(t-g)}{ng} \qquad (7)$$

Moving past indicators, Simões discusses Sharpe Ratio as a way of measuring the reward-to-variability ratio of

a trading strategy. This measure, created by Nobel Prize William Sharpe, allows us to compare two strategies with different returns, and see if the additional return of one strategy is due to applying a more risky strategy, or to a smarter investment strategy [16]. Formula given below:

$$SR = \frac{R - R_f}{\sigma} \quad (9)$$

In refer. (Garcia-Almanza, 2006) discusses the Repository Method. RM is a method of mining information acquired through an evolutionary algorithm. The objective is to compile several rules that model the rare cases in diverse ways. Since the number of positive examples is very small, it is important to gather all available information about them. The overlearning produced by this method attempts to compensate for the lack of positive cases in the data set [11].

In refer. (Daunhawer, 2018) examines the Efficient Market Hypothesis. Noting how financial markets are self regulating to an extent "A financial market reflects all available information and adapts to new information fully and swiftly" [8]. EMH asserts that the price of a security is a valid and accurate representation of its actual value and that therefore investors will not be able to gain irregular returns in the long term strategies [11]. Lo (2004) puts it this way: "behavioral finance suggests that market efficiency can be reconciled with the existence of individuals' behavioral biases through the framework of the adaptive markets hypothesis." [13].

In refer. (Drezewski, 2018) notes population maintenance. "It was observed that the mechanism of maintaining the population caused the population diversity to lower, and as a result, the population was not tracking the optima of the fitness function. Much better results were obtained when the newly initialized individuals were used after every shift of the time window." [7]. His conclusion is interesting.

Drezewski also discusses the Greedy Strategy in his work, writing that it reveals potential to generate profit. "The GS opens an initial order if there is a gap between current open and the high or low of the previous bar. If open is greater than previous high, strategy goes long, if open is below the low of the previous bar, it opens a short position. After a position is opened, it will continue filling orders in the same direction as long as the color of the candle is consistent with the open position. If the current position is long, new long orders will be created for every consequent green candle and vice versa. This will proceed until the candle of a different color or until the limit of filled orders for a day is reached." [10]. Drezewski writes that a complete investment approach using GS needs more research to implement. His findings were that the system was not able to overcome a simple buy and hold strategy, but it could produce large profit margins in long term trials.

In refer. (Sipper, 2018) covers parameter selection and what factors go into selecting successful ones. Sipper writes that good parameters range over the entire spectrum, allowing the algorithm to optimize at a more in depth level [18]. Sipper recounts typical effects of changing certain parameters as well, noting that increasing population size and generation count do not necessarily result in better output. Overfitting the output to historical data is a risk in experiments with too many generations. Regarding mutation Sipper recounts that Crossover and mutation rates can range widely. Crossover-mutation pairs showed no tendency to converge. At most, such pairs should not both be low (which makes sense given that an evolutionary algorithm requires inter-generational variation)." [18]

IV. Approach

A. Abbreviations and Acronyms

1) MA: Moving Average
2) SMA: Simple Moving Average
3) SMAC: Simple Moving Average Crossover
4) RSI: Relative Strength Index
5) MAD: Moving Average Derivate
6) MSR: Modified Sortino Ratio
7) ACE: Agent-Based Computational Economics
8) FOREX: Foreign Currency Exchange
9) EMH: Efficient Market Hypothesis
10) DEAP: Distributed Evolutionary Algorithms in Python
11) GS: Greedy Strategy
12) CLI: Command Line Interface
13) ETF: Exchange-Traded Fund

B. Units

All units are given in United States Dollars (USD$). The scale and iterative unit used in the strategy are in single days - 24 hour periods.

C. Equations

1. RSI = 100 – (100 / (1 + (μ)))
   a. μ *is the Avg Net Profit*
2. MSR = $\frac{\mu}{\sigma}$
   a. μ *is the Avg Net Profit*
   b. σ *refers to standard deviation of returns*
3. SMA = $\sum_{index = endIndex - 9}^{endIndex} \frac{Price[Index]}{10}$
4. $VT = \frac{V_f - V_i}{V_i}$
   a. V_f refers to final volume
   b. Vi refers to initial volume
5. $P(t) = \frac{1}{1 + e^{-1}}$
   a. P(t) the value of the Index at day "t".
6. $P_i = \frac{f_i}{\sum_{j=1}^{N} f_j}$
   a. f_i *refers to fitness of the ith individual*

b. $N\ refers\ to\ population\ size$
c. P_i refers to the value of the Index at day "i".

7. $MAD\ (d) = \frac{\sum_{t=d-n+1}^{d} P(t) - \sum_{t=d-n+1}^{d} P(t-g)}{ng}$
   a. P(t) the value of the Index at day "t".
   b. d = day of calculation
   c. $n = refers\ to\ population\ size$
   d. g refers to the distance between the two days to calculate the secant (the original strategy consists of a fixed g with value 1).

8. $SMAC(d) = \frac{1}{s}\sum_{t=d-s+1}^{d} P(t) - \frac{1}{l}\sum_{t=d-l+1}^{d} P(t)$
   a. d = day of calculation
   b. P(t) refers to the value of the Index at day "t".

9. $SR = \frac{R-R_f}{\sigma}$
   a. R refers to the average return of the strategy
   b. Rf refers to the risk free rate (normally the rate of the US treasuries security)
   c. σ refers to standard deviation of the strategy

10. $f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$
    a. x refers to the sum of previous gene value and some small random value calculated using the above equation

D. Strategy

An algorithmic trading strategy for the stock market utilizes automated instructions, usually including mathematics, indicators, and statistics, to buy and sell. The benefits of algorithmic trading are speed, computational power, and blind execution without emotion - a fault that many humans have when trading.

The strategy used in this paper consists of two different moving average indicators as well as an RSI indicator. An algorithmic trading strategy for the stock market utilizes automated instructions, usually including mathematics, indicators, and statistics, to buy and sell. The benefits of algorithmic trading are speed, computational power, and blind execution without emotion - a fault that many humans have when trading.

The strategy used in this paper consists of two different moving average indicators as well as an RSI indicator – three parameters total. The moving average of a stock is calculated by finding an n number of closing prices and plotting the average of those previous prices as a point. For example, if we wanted to find a 50-day moving average on day 100 of a stock, we would sum the previous 50 days' closing price (day 50 to day 99), and plot that average price on day 100. The 50-day moving average on day 111 would consist of averaging day 51 to day 100 and so on. The moving average of a stock helps smooth out trends and mitigate random, "chaotic" short-term movement. The larger the moving average, the less impact chaotic movement has on the indicator – it takes more and more of a trend to cause movement in the indicator.

The conditions of our strategy are as follows. We had a fast moving average x and a slow moving average y, where $y > x$. Alongside this was our RSI indicator. When the fast moving average crossed (had a value greater than) the slow moving average, our algorithm placed a market order to full-buy its position. We would expect the market to be in an uptrend here. Our sell execution had two conditions – whichever one was met first. A sell instruction would be executed if the fast moving average crossed back under the slow moving average (AKA the slow moving average crossover the fast moving average), or if the stock's current RSI fell below the RSI selling parameter – indicating that a downtrend was possibly occurring. If one of these two conditions are met, then we would liquidate all of our holdings and wait for the next buy signal again. In the figure below, the slow moving average (50 day) is in purple, while the fast moving average is in blue (10 day)

Figure 1: Moving Average Strategy Visual

E. Packages and Frameworks

Three main packages and frameworks were used in this paper: QuantConnect, LEAN Engine, and DEAP. Quantconnect is a browser/cloud-based, open-source, algorithmic trading platform where engineers can develop, execute, and backtest their strategies for a multitude of equities. QuantConnect provides their own trading API for market access and trading manipulation. Alongside QuantConnect is their LEAN Engine: QuantConnect's CLI backtesting framework. This allowed us to script the backtests and incorporate the Genetic algorithm alongside it, which was done through DEAP. Distributed Evolutionary Algorithms in Python (DEAP) is an evolutionary computation framework that allows for accelerated testing/"prototyping" of ideas. I used genetic algorithms from this framework. With these

packages we were able to successfully develop, backtest, and evolve our trading strategy.

F. Testing Process and Setup

We first needed to successfully set up the LEAN Engine for backtesting and develop a skeleton for our trading strategy. The framework's structure requires a single class – your strategy. There are two functions inside this class: *Initialize* and *OnData*. The *Initialize* function is executed once during a backtest and initializes specific data and options for your backtest such as starting and ending date, "iteration" (minute, day, month, year, etc), starting cash, and other indicator initializations. The *OnData* function is executed every set iteration from *Initialize*, and this is where money orders are executed based on conditions set. Once done, we needed to now implement the genetic algorithm.

The genetic algorithm needed to evolve a set of three parameters of the strategy: fast moving average, slow moving average, and the RSI threshold. The strategy class in the LEAN Engine is quite rigid, so in order to incorporate scripted parameters through the genetic algorithm, we needed to edit a specific JSON file for the parameters for each entity of the population. A function we wrote would handle this. The genetic algorithm is now set to evolve the parameters. We saved each entity of the population along with their fitness throughout the generations and saved them to a .csv file.

We needed to both train and evolve our genetic algorithm's parameters for the strategy as well as test the results. In training/evolution, we used the historic market data for "S&P 500" ("SPY") from January 1, 2009 to January 1, 2015. Once trained, we would then test the generation's parameters on a different time period to see the effectiveness. The testing time period was January 1, 2015 to January 1, 2021.

G. Genetic Algorithm Implementation

Through the DEAP's evolutionary algorithm framework, we are hoping to utilize the iterative process of a genetic algorithm to find a global optimum for a best set of parameters for our strategy. The genetic implementation consists of a six step process.

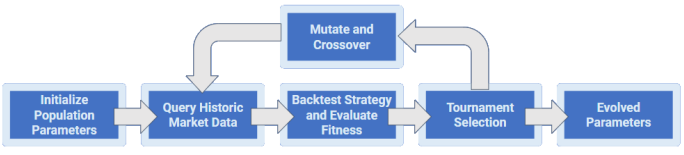

Figure 2: Genetic Algorithm Implementation Flow Chart

Step one is to initialize the population (100) with a random set of parameters: fast moving average from 1-51, slow moving average from 10-151, and RSI threshold from 1-101. With these parameters set, the second step consists of querying market data for backtesting using QuantConnect. The ticker we tested was "SPDR S&P 500 ETF Trust." Step three includes using the LEAN Engine to backtest a strategy with the market data and extracting a fitness value for each entity. The fitness value we used was the percent net return at the end of the date period. In step four the genetic algorithm used tournament selection for determining the most fit individuals. Step five is how each generation passed on traits to the children – using mutation and crossover. The crossover probability was 50% and the mutation probability was 30%. The algorithm then loops through steps 2-5 until a global optimum is found.

V. Results

A. Fitness

The genetic algorithm used converged over 10 iterations. This was relatively quick, especially compared to brute forcing all of the millions of combinations of parameters for this strategy – a very computational heavy process. Evolutionary algorithms allow for much faster convergence. As you can see from the table below, the Average Fitness of the population converges closely toward the Best Fitness on iteration 8.

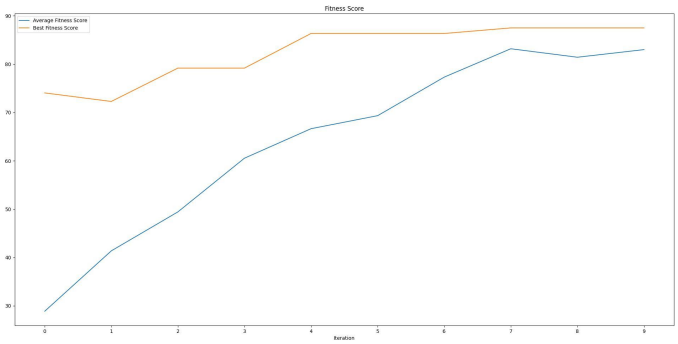

Figure 3: Average and Best Fitness Graph

TABLE I.

ITERATION	BEST FITNESS	AVERAGE FITNESS
0	74	28
1	73	42
2	78	49
3	79	61
4	86	66
5	86	69
6	87	75
7	88	84

8	89	82
9	89	83

Fig. 1. Shown above, Average fitness converges to Best fitness rather quickly, in 8-10 iterations.

B. Training

Training consisted of 10 generations through the genetic algorithm. Each individual's fitness was determined based on the trading strategy's percent net return after the training period from January 1, 2009 to January 1, 2015. [stats]. Each generation had a greater fitness score than the previous, implying optimal evolution. The final evolved parameter set had a fast moving average of 42, a slow moving average of 66 and an RSI threshold of 9. The figures below display the backtesting results from generation 1, 5, and 10 and their corresponding statistics. Percent net return (fitness value) increased throughout from 24.71% to 62.89% to finally 88.33%.

Figure 3: Generation 1 Training Results (Fast Moving Average - 20, Slow Moving Average - 120, RSI Threshold - 50)

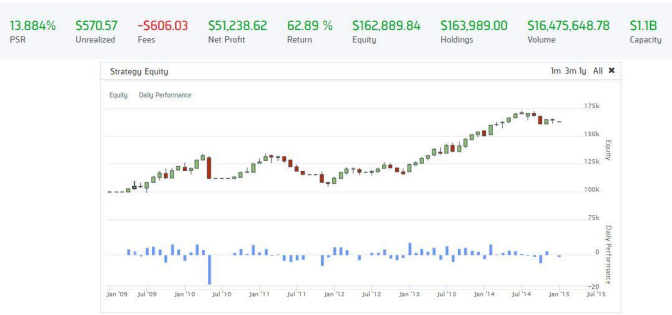

Figure 4: Generation 5 Training Results (Fast Moving Average - 35, Slow Moving Average - 60, RSI Threshold - 33)

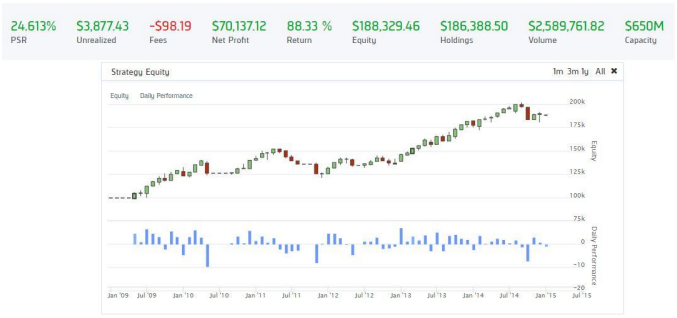

Figure 5: Generation 10 Training Results (Fast Moving Average - 42, Slow Moving Average - 66, RSI Threshold - 9)

One interesting thing we noticed was that the converged solutions for the parameters had very low RSI thresholds- values so low that they would not get triggered in the expected course of the market trends, making the condition essentially useless to our goal. We expected the RSI threshold values to range from 25-55, however, the converged solution had a value of 9. This is most likely due to the loose condition of the parameter. RSI tends to fluctuate sporadically whether the market is on an uptrend or a downtrend – this fluctuation could have triggered an unwanted condition to sell. The expected condition for the RSI threshold would be right before a crossing of the fast moving average to the slow moving average. This would mean that many people are selling, causing the beginning of a downtrend and an eventual cross of the fast moving average over the slow moving average. The RSI threshold condition would queue, and the portfolio would mitigate losses. However, the condition would trigger too early, and result in missing potential gains – decreasing the fitness score of an individual.

Figure 6: Visual example of RSI threshold parameter condition failing - Selling during uptrend

Figure 7: Visual example of expected RSI threshold parameter condition - Selling before downtrend

A. Testing

In the testing period. An individual's parameters were kept the same, but the backtesting date period was different. The tested date period was January 1, 2015 to January 1, 2021. Optimally, a training set of parameters would perform relatively similar on the testing dates. This would mean that the parameters are not overfit to the training date period, but rather can be applied to different time periods – holding true to future market ranges as well.

The figures displayed below show the percent of net return for the testing periods for generations 1 and 5 (and inferrably those in between) performed relatively similar to the percent of net return from the training: 24.71% to 22.23% for the first generation and 62.89% to 57.99% for the fifth generation. However, the converged, "global optima" parameters performed much worse in the testing than the training – from a training value of 88.33% to 38.83%. Because of this, we can infer that the converged parameters were overfit to the training period. In algorithmic trading, overfitting occurs when an algorithm adapts and models its training period and historical data that the strategy is not efficient for future, unknown, future data.

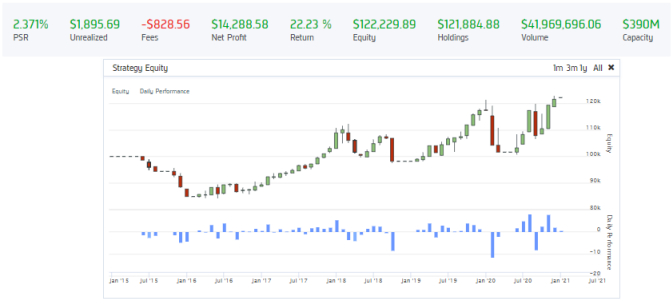

Figure 8: : Generation 1 Testing Results (Fast Moving Average - 20, Slow Moving Average - 120, RSI Threshold - 50)

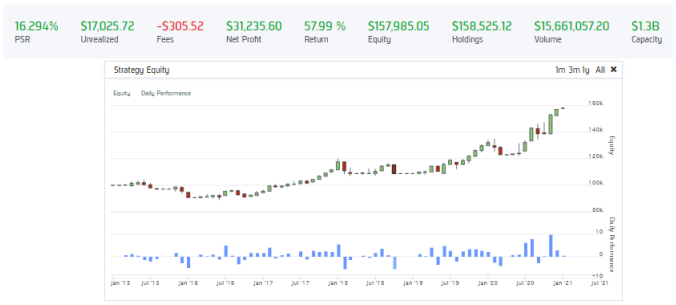

Figure 9: Generation 5 Testing Results (Fast Moving Average - 35, Slow Moving Average - 60, RSI Threshold - 33)

Figure 10: Generation 10 Testing Results (Fast Moving Average - 42, Slow Moving Average - 66, RSI Threshold - 9)

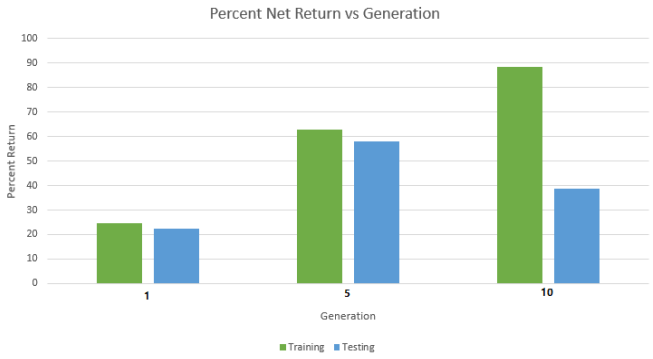

Figure 11: Bar Chart Displaying Generations 1, 5, and 10 Percent Net Return on both Training (green) and Testing (blue)

VI. Conclusion, Future Work, and Application

A. Conclusion

From the results examined above, we were able to conclude that genetic algorithms can indeed converge algorithmic trading parameters and use them to train a technical trading agent to make profitable transactions in the market. Fitness of the agents was equated to the percentage of net return. Over 10 generations fitness increased 200% (55

percentage points, see Table I.) However, evolutionary algorithms tend to overfit to historical data given in training, and the results partially confirm overfit parameters. Although the previous generation's (1-8) parameters allowed for similar testing outcomes when compared to training (< 10%), the converged parameters had a negative change of over 125% (50 percentage points) for its fitness (percent net return). From this, we can infer that the converged parameters were overfit to the training data. There is not enough evidence with this current strategy and setup that our genetic algorithm can successfully optimize trading parameters for future market data.

B. Future Work

Since completing this research we have compiled several ways of increasing the potential of this algorithm. First, a wider, more diverse set of training data from different time periods in the market, as well as possibly markets other than the S&P 500 would grant the agents more potential patterns and scenarios to learn from. Alongside this, we could introduce training data that follows different trends. For example, the 2009-2015 data that we initially used to train is considered a 'bull market', which saw primarily growth as the market recovered from the 2008 financial crisis. It would be interesting to see the algorithm's results if it were trained with a 'bear market', or one which endured a major crash, to see if it could mitigate the amount of loss sustained to its portfolio.

Another way of potentially fine tuning the algorithm would be better optimizing RSI entry and exit points. Historically, RSI has been used to indicate when a stock becomes overbought or oversold. It was thought that when RSI falls below 30 it is considered oversold, and it has reached the best value to buy. When RSI reaches above 70 it is thought to be overbought and at its highest relative price to be sold. We were able to find that instead of buying or selling as soon as RSI reaches those thresholds, wait until it re-enters from outside of the threshold. If we implemented this method, we could see both lower buying prices and higher selling prices, making the algorithm overall more efficient and effective.

C. Application

Real world use of this algorithm could be in financial and retirement planning. Future versions of the program would ideally be able to predict crashes before they occur, giving portfolio managers a major heads up, and allowing them to close their positions before said crash.

In the likely event Social Security runs dry before Millennials and Gen Z reach retirement age, genetic trading algorithms such as this will take the system's place. This would hopefully eliminate inefficient government management of our hard earned tax dollars (6.2% of every citizen's income), in exchange for privately managed funds.

If widespread use of algorithms such as these does become commonplace, it would work to eliminate the boom-bust cycle of the market caused by fundamentalist traders and inefficient portfolio management today. Sudden crashes are often caused by real world news events where fundamentalists panic sell their positions causing a chain reaction. If the ratio of technical traders/algorithms using technical indicators to fundamentalist traders increases, the drastic effect world events will have on the market will become less influential as the news has no effect on technical signals.

VII. References

[1] Allen, Franklin, and Risto Karjalainen. *Using Genetic Algorithms to Find Technical Trading Rules*, Journal of Financial Economics, 12 Oct. 1995, https://www.cs.montana.edu/courses/spring2007/536/materials/Lopez/genetic.pdf

[2] Bonde, Ganesh, and Rasheed Khaled. *Stock Price Prediction Using Genetic Algorithms and Evolution Strategies*, University Of Georgia, 22 Apr. 2017, http://worldcomp-proceedings.com/proc/p2012/GEM4716.pdf.

[3] Chen, Shu-Heng. *Evolutionary Computation in Economics and Finance*. Heidelberg: Physica-Verlag HD, 2002. Web.

[4] Chen, Yan, Zezhou Xu, and Wenqiang Yu. "Agent-Based Artificial Financial Market with Evolutionary Algorithm." *Ekonomska istraživanja* ahead-of-print.ahead-of-print 1–21. Web.

[5] Conti, Jean Pierre Jarrier, and Heitor Silverio Lopes. *Algorithmic Trading Using Genetic Algorithms in the Brazilian Stock Exchange*, Telefonica Brazil, Jan. 2020, https://www.researchgate.net/publication/343688682_Algorithmic_Trading_Using_Genetic_Algorithms_in_the_Brazilian_Stock_Exchange.

[6] Daunhawer, Imant. *Evolutionary Algorithms for the Discovery of Trading Rules in High-Frequency Betting Markets*, University of Konstanz, 1 Mar. 2018, https://www.uni-konstanz.de/bioml/bioml2/publications/Papers2018/Daunhawer18.pdf.

[7] Drezewski, Rafał. *The Bio-Inspired Optimization of Trading Strategies and Its Impact on the Efficient Market Hypothesis and Sustainable Development Strategies*, AGH University of Science and Technology, 7 May 2018, https://www.mdpi.com/2071-1050/10/5/1460/pdf-vor.

[8] Fama, E. F. (1970). "Efficient Capital Markets: A Review of Theory and Empirical Work." Journal of Finance 25.2, pp. 383–417.

[9] Frankenfield, Jake. "How General Equilibrium Theory Works." *Investopedia*, Investopedia, 19 May 2021, https://www.investopedia.com/terms/g/general-equilibrium-theory.asp.

[10] Garcia-Almanza , Alma Lilia, and Edward P.K. Tsang. "Forecasting Stock Prices Using Genetic Programming and Chance Discovery." *Forecasting Stock Prices Using Genetic Programming and Chance Discovery*, University of Essex, Jan. 2006, http://repec.org/sce2006/up.13879.1141401469.pdf.

[11] "Greedy Strategy." *TradingView*, https://www.tradingview.com/support/solutions/43000599876-greedy-strategy/.

[12] Lo, (2004). "The Adaptive Markets Hypothesis: Market Efficiency from an Evolutionary Perspective." Journal of Portfolio Management 30, pp. 15–29.

[13] Harrington, Peter Boyd. *Evolutionary Computation for Rule Discovery in Algorithmic Stock Trading.* Master's thesis, Harvard Extension School. 2019. https://dash.harvard.edu/bitstream/handle/1/37365093/HARRINGTON-DOCUMENT-2019.pdf?sequence=1&isAllowed=y

[14] Long, Heather. "Nearly 70% of Investors Lost Money in 2015." *CNNMoney*, Cable News Network, 31 Dec. 2015,

https://money.cnn.com/2015/12/31/investing/stocks-market-2015/index.html

[15] N. Sam M.S., "RATIONAL-ECONOMIC MAN," in *PsychologyDictionary.org*, April 28, 2013, https://psychologydictionary.org/rational-economic-man/ (accessed May 6, 2022).

[16] Newport, Frank. "Many Americans Doubt They Will Get Social Security Benefits." *Gallup.com*, Gallup, 22 May 2021, https://news.gallup.com/poll/184580/americans-doubt-social-security-benefits.aspx.

[17] Simões, Adriano, et al. "Evolutionary Computing Applied to Stock Market Using Technical Indicators." *Evolutionary Computing Applied to Stock Market Using Technical Indicators* , Instituto Superior Técnico, Jan. 2010, http://web.ist.utl.pt/adriano.simoes/tese/docs/icec_v1.pdf.

[18] Sipper, Moshe, et al. *Investigating the Parameter Space of Evolutionary Algorithms*, University of Pennsylvania, 17 Feb. 2018, https://biodatamining.biomedcentral.com/track/pdf/10.1186/s13040-018-0164-x.pdf.

[19] Subramanian, Harish, et al. *Designing Safe, Profitable Automated Stock Trading Agents Using Evolutionary Algorithms*, The University of Texas at Austin, 2006, https://www.cs.utexas.edu/~pstone/Papers/bib2html-links/GECCO06-trading.pdf.

[20] Teeples, Allan W. *An Evolutionary Approach to Optimization of Compound Stock Optimization of Compound Stock Trading Indicators Used to Confirm Buy Signals* , Utah State University, Dec. 2010, https://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=1816&context=etd

[21] Michalewicz, 1994. Genetic Algorithms + Data Structures = Evolution Programs. Springer-Verlag, 2nd edition, 1994

[22] Yan, W. and Clack, C., Evolving Robust GP Solutions for Hedge Fund Stock Selection in Emerging Markets, in Proceedings of the 2007 GECCO conference companion on Genetic and evolutionary computation. 2007, London, UK, 2234-2241.