Investigating the Viability of Generating Trading Strategies with Multi-Objective Evolutionary Algorithms

Candidate Number: 019203

Abstract

Trading strategies are commonly used by professionals to try to gain an advantage in the stock market by exploiting short-term trends in price data. They consist of a number of technical indicators that can be used to generate buy and sell signals at the most profitable times. However, finding optimal trading strategies is challenging and complex. This report proposes a multi-objective evolutionary approach to search for successful trading strategies using a selection of appropriate indicators and objectives that have been summarised in this report. After reviewing the literature, several key techniques and practices have been identified for using this approach. A genetic programming model with the Non-dominated Sorting Genetic Algorithm II (NSGA-II) was used to search for optimal strategies. This research analyses the overall generalisation performance of the generated strategies and investigates the effects of changing three elements when running the algorithm: the combinations of objectives used; the number of objectives used; and the length of data window used for training and testing the data. The results have enabled analysis on the generalisation performance of the strategies which show promising results. This report highlights some insights which can help improve the performance of generating successful trading strategies and provides directions for future research.

Keywords: Evolutionary algorithms; Multi-objective optimisation; Technical Analysis; Genetic Programming; NSGA-II.

I certify that all material in this dissertation which is not my own work has been identified.

Contents

1 Introduction and Motivation

Achieving consistent profits in the stock market is a challenge that many investors face and often fail to achieve. Due to the random nature of the markets, it is difficult to find successful trading strategies which are profitable [1]. A trading strategy is a set of predefined rules and instructions that are used to inform buy, sell and hold positions [2]. Investors can devise trading strategies using a selection of technical indicators and use them to apply technical analysis on past price data for various equities or stocks. The aim of technical analysis is to find trends and patterns in previous data to predict what the price will do in the future so an investor can profit from rising or falling prices.

This process can be difficult due to the complexity of using multiple technical indicators at once and requires sophisticated knowledge and experience to use them correctly. Ultimately, the success of a trader is down to the correct selection and execution of a suitable strategy [3]. Less than 20% of traders are able to consistently make profits in the stock market signifying the challenge of being successful in this discipline [4]. Many of the reasons for this high failure rate is due to a poor choice of strategy and psychological factors that are inherent to humans, such as overconfidence and fear which can lead to poor decisions [4]. This leads to the conclusion that there is value in designing a system that can find successful trading strategies that avoid these common pitfalls.

The Efficient Market Hypothesis is an accepted theory that argues that the current price of a stock is efficient, meaning that its value is being reflected by all the currently available data [5]. This implies that if all past information is included in the value of the current price then it is a pointless task attempting to predict future prices using past data [6]. Prices are therefore believed to follow a random walk and are inherently unpredictable [7].

It would appear that it is a futile endeavour to engage in further research into finding technical analysis trading strategies for markets that are efficient and many are sceptical about the validity of using technical analysis to make trading decisions. However, there is research providing evidence that there are times when the Efficient Market Hypothesis does not hold and the market is inefficient, particularly when the market is driven by psychological factors [8]. Also, if a large portion of investors follow established trading practices then this can influence the price of a security and can allow other investors to make decisions based on these established practices [6]. This suggests that there are still opportunities to use technical analysis to exploit deviations from the efficient market hypothesis that can allow investors to benefit from using these techniques. On the contrary, if the market is efficient then it is still possible to be able to predict prices by analysing time-varying equilibrium returns [9]. Evidently, there are still opportunities to use technical analysis to make profits from the markets which provide grounds for this research into finding successful technical analysis trading strategies. Therefore, this project seeks to look into the performance of using a multi-objective evolutionary approach for solving this problem.

2 Literature Review and Specification Summary

2.1 Related Work

Due to the factors mentioned previously explaining the legitimacy of technical analysis trading, there has been extensive recent research into using algorithms to generate trading strategies. The abundance of financial data has made it increasingly compelling to provide more accurate and faster analysis of this data to provide profitable solutions. Evolutionary algorithms have been utilised by many researchers as this stochastic search operation has a strong ability in dealing with large and complex search spaces [10]. The search space proposed for finding trading strategies fits these criteria as there are numerous technical indicators, each with multiple parameters, which can be combined to form strategies.

In the real world when investors make trading decisions, they do not just consider one objective. They will base their decisions on taking into account multiple criteria which may often be conflicting. In the world of finance, profit and risk are often conflicting in nature [11]. Therefore, when designing systems to find trading strategies there is a need to take into account multiple objectives for the fitness criteria of the strategies.

Evolutionary algorithms provide a more suitable multi-objective method for this project as they can develop a set of solutions, called the non-dominated set, rather than just a single candidate solution. There are a number of multi-objective genetic algorithms that can be used to find the non-dominated solution set. Evolutionary algorithms are commonly used to solve multi-objective problems as they can produce a population of solutions as the final outcome. This unique ability

of evolutionary algorithms to generate multiple optimal solutions in one simulation is what makes the process suitable for analysing the trade-offs between multiple objectives [12].

Work by Lohpetch & Corne [13], shows that multi-objective optimisation can be more effective for finding successful trading strategies than single-objective optimisation. They conducted a series of comparisons of single-objective algorithms against various combinations of multiple objectives, and in all cases, results with multiple objectives outperformed the single objective ones. They also found that the multi-objective formulations which do not contain elements of risk as an objective tend to perform worse and were less likely to out-perform the Buy and Hold Strategy (B&H). The B&H strategy is defined as buying at the start of the trading period and selling at the end. This work indicates the importance of introducing multiple objectives into finding trading strategies, with particular importance in using risk for an objective.

Much of the literature using evolutionary techniques to evolve trading strategies have found the genetic programming approach to be more appropriate for finding trading strategies as it can allow the strategies to be developed and represented in a more unique and natural way [1]. The works in [7], [13]–[15] were successfully able to use genetic programming to find profitable strategies. They found that the representation is more suitable for allowing more dynamic trading strategies to be evaluated so therefore this approach will be adopted for this research.

Lohpetch and Corne [16] also found that using training, validation and evaluation method can help to make the solutions that are found more robust. This is a method where the strategy is first evolved and found using the training data window. After this, the found strategy is then run on the validation data window and its performance is recorded. The strategy that performed the best on the validation window is then tested on the evaluation data window, and the results from using this data window are the final results that are used for the analysis [13]. This technique has been proven to be an effective method for analysing the results of the genetic algorithm so the technique will be incorporated into this project however these data windows have been named training, testing and validation for the purposes of this project.

Becker and Seshadri [17] and Lohpetch and Corne [16] incorporated a performance consistency objective into the fitness function which benefits the strategies that can consistently perform well. They both found that incorporating this objective into the fitness function helped the found strategies to regularly outperform the B&H strategy on unseen data. It helps to reduce the risk of the trading strategies because it prevents any anomalies or volatile strategies from being rewarded a higher fitness value. It encourages the population to consistently perform rather than just performing very well on one occasion, which could have led to higher returns. Due to the success that previous research has achieved using a performance consistency factor, this project will adopt this objective.

The purpose of this research is to extend the findings in the literature and analyse the difference in performance when the number of objectives are changed and used in different combinations. After reviewing the literature, it has been decided that to carry out research into the problem of finding efficient trading strategies, an elitist non-dominated sorting genetic algorithm-II (NSGA-II) [18] in conjunction with genetic programming [19] will be used.

2.2 Project Specification

The research hypothesis as stated in the literature review is as follows:

"Can multi-objective evolutionary algorithms generate trading strategies that are successful by outperforming the B&H strategy?".

This question will be the over-arching aim for this project and the goal will be to evaluate the viability of generating trading strategies using NSGA-II with genetic programming. To carry out this research a program will be produced that is able to generate readable trading strategies that are valid and can be used on the stock market. This involves implementing NSGA-II with genetic programming and modelling each of the technical indicators to accurately generate trading signals. Additionally, the program needs to be able to visually display the results of the evolutionary algorithm. The Pareto front needs to be identified and analysed for the performance of the algorithm to be monitored and the set of trade-off solutions to be found. The system should also be easily customizable to allow experimentation with different data, parameters and objectives.

As well as answering the research hypothesis, the project will perform some research into altering a range of parameters and objectives to see if any further contributions to this field can be established and if they can aid the ability of future systems in finding strategies that outperform

the B&H. To be summarised, the areas to be researched in this work are:

1. Analysing the performance of the strategies when changing the number of objectives used.
2. Analysing the performance of the strategies, using 11 different multi-objective combinations.
3. Analysing the effects of using different timescales for training and testing data windows.
4. Evaluating the generalisation performance for the proposed system to see if strategies can be found that outperform the B&H strategy.

3 Finding Successful Trading Strategies

3.1 Technical Indicators

Some technical indicators are more popular than others due to their known ability to generate successful trading signals. For this project, our set of technical indicators are compromised as five technical indicators that have been used in previous research [11] [10] [20]. These indicators are outlined in Table 1.

Technical Indicators	Type
Simple Moving Average (SMA)	Trend
Exponential Moving Average (EMA)	Trend
Moving Average Convergence Divergence (MACD)	Trend
Relative Strength Index (RSI)	Momentum
Stochastic Oscillator (SO)	Momentum

Table 1: Selection of technical indicators used to find efficient strategies.

The technical indicators that are being used in the project fall into two categories, trend indicators and momentum indicators. A trend indicator analyses the overall underlying trend of the price and predicts that the price will continue to move in this trend. They are known to be effective because a price usually trends whenever it is not reverting back to the mean which can allow the indicator to identify valuable insights into the future direction of the price [8]. A momentum indicator can identify the speed of price movements by analyzing the price over certain time periods [21]. Momentum indicators typically appear as an oscillator in a range from 0 to 100. The relative strength index (RSI) and Stochastic Oscillator (SO) are momentum indicators and the remaining are trend indicators.

Each of the indicators in table 1 will be pre-computed and appended to a CSV file. This is to improve the runtime of the algorithm as every value for each technical indicator on each day will only need to be calculated once and there will be no repeat indicator calculations. The following subsections contain a summary of how each technical indicator functions and the formulas for how the program will precompute their values:

3.1.1 Simple Moving Average (SMA)

As the name suggests, the SMA calculates a weighted average for a period of time. An SMA for day i is the average of the closing price for the last w days from day i. w is the time period (or window) for the SMA and is a parameter for this indicator. A shorter time period will make the SMA more responsive to changes in price. The SMA is useful because it removes the noise from the price data and can allow the trader to focus on the underlying trend and make trades based on this trend. An SMA crossover strategy will be used where two different SMA's are selected, one with a shorter time-period and one with a longer time period. When the shorter SMA crosses above the longer SMA then a buy signal is generated. When it crosses below it a sell signal is generated [8] [22]. The SMA can be defined by the following, where p_i is the closing price at time i, t is the current time and w is the length or window of the moving average:

$$SMA(t, w) = \frac{\sum_{i=t-w}^{t} p_i}{w} \tag{1}$$

3.1.2 Exponential Moving Average (EMA)

The EMA is similar to the SMA but more recent data has a greater weight in affecting the value of the moving average. Therefore, an EMA is more responsive to recent changes in price data than an SMA. This property makes the EMA a useful technical indicator as it can allow the trading signals to be generated even faster allowing for a more profitable trade. A crossover strategy, similar to the one used by the SMA crossover strategy, is used with the EMA to generate buy and sell signals [23]. The EMA can be defined as the following where p_t is the closing price at time t, $EMA_{t-1,w}$ is the EMA of the previous day, t is the current time, w is the length or window of the moving average and x is a smoothing multiplier that is used to calculate the EMA.

$$\begin{aligned} &EMA(t,\ w) = p_t x + EMA_{t-1,w} \cdot (1-x) \\ &\text{where } x = 2/(w+1) \end{aligned} \tag{2}$$

3.1.3 Moving Average Convergence Divergence (MACD)

This indicator has been selected as it can identify the strength of trends in either direction and generate useful trading signals in response to the strength of these trends. The MACD line is generated by subtracting a longer EMA from a shorter EMA. A shorter EMA is one that has a smaller window and a longer EMA has a longer window. A signal line (SIG) is then generated to be used as a comparison against the MACD line. The SIG is a certain length EMA of the MACD line. A crossover strategy is then used with the MACD line and the SIG [24]. When the MACD line crosses above the SIG then a buy signal is generated and a sell signal is generated when it crosses below it. The following defines the *MACD* and *SIG*, where w is the length/window of the moving average, t is the current time and x is the smoothing multiplier.

$$\begin{aligned} &MACD(t) = EMA(t,f) - EMA(t,s) \\ &SIG(t,\ w) = MACD_t\ x + SIG_{t-1,w} \cdot (1-x) \\ &\text{where } x = 2/(w+1) \end{aligned} \tag{3}$$

3.1.4 Relative Strength Index (RSI)

This indicator measures the momentum of share price movements. It evaluates whether the share is overbought or oversold which can identify profitable times to buy and sell. The RSI value is presented as an oscillator and has a value between 0 and 100. When this value goes over or under certain thresholds a buy or sell signal is produced [25]. It can be defined by the following where, w is the number of days representing the window, $D^+(w)$ is the set of days where the closing price **rises** over the last w days, $D^-(w)$ is the set of days where the closing price **falls** over the last w days, p_i is the closing price on day i and *p(i-1)* is the closing price on the previous day before day i.

$$\begin{aligned} &RSI(w) = 100 - (\frac{100}{1+RS(w)}) \\ &\text{where } RS(w) = \frac{\sum_{i \in D^+(w)}\ p_i - p_{i-1}}{-\sum_{i \in D^-(w)}\ p_i - p_{i-1}} \end{aligned} \tag{4}$$

3.1.5 Stochastic Oscillator (SO)

This indicator is another momentum indicator which provides buy or sell signals when its value goes above or below certain thresholds. It is different from the RSI because it is based on the principle that the closing price should be on the same path as the current trend whereas the RSI is based on the speed of price movements [26]. It can be defined by the following where w is the window to find the SO, p_i is the closing price on day i, l_w is the lowest price during the last window w and h_w is the highest price during the last window w.

$$SO(w) = \frac{p_i - l_w}{h_w - l_w} 100 \tag{5}$$

3.2 Technical Indicator Parameters

Each technical indicator can have an associated set of parameters that can alter the performance of the particular indicator. The parameter values have been selected by using popular parameters

Indicator	Parameters
SMA(w)	$w \in \{5, 10, 20, 30, 50, 100, 200\}$
EMA(w)	$w \in \{5, 12, 26, 30, 50, 100, 200\}$
MACD(s,f,w)	$s \in \{26, 35\}, f \in \{5, 12\}, w \in \{5, 9\}$
RSI(w)	$w \in \{7, 14, 28\}$
SO(w)	$w \in \{7, 14, 28\}$

Table 2: The parameter sets chosen for each technical indicator.

that are conventionally used by traders [22]–[26]. The parameters that have been chosen can be seen in table 2.

In table 2, the w argument is the window size in days for the particular indicator. For the MACD indicator, the s and f arguments are the slower and faster window sizes for the EMA's used to calculate the MACD. The w argument, in this case, is the window size for the signal line used to compare with the MACD. Details of how each indicator operates can be seen in section 3.1 for further clarification.

3.3 The Objectives

The trading strategies are optimised using a set of different objectives which inherently transforms the problem into a multi-objective optimisation problem. The technical indicators that have been outlined above are combined and varied to form trading strategies to achieve two, three and four objectives. The background of the multi-objective approach will be discussed in further detail in Section 4. We considered eleven instances, with either two, three and four objectives, outlined in Table 3. These objectives are non-commensurable as each is measured in different units and they are also conflicting in nature. The aim is to compare the results of all eleven instances against a baseline solution. The five objectives used are as follows:

3.3.1 Profit

$$\text{Maximise: } P = \sum_{i=1}^{n} P_{i,sell}/P_{i,buy} \cdot A_i \tag{6}$$

where n is the number of trades, $P_{i,sell}$ is the closing price when sold for the ith trade, $P_{i,buy}$ is the closing price when bought for the ith trade, A_i is the amount invested at the ith trade. Profit is an important objective as it is often the sole purpose as to why one engages in investing in the stock market. A successful strategy is one that can generate as higher a return as possible. We considered adding a cost per trade, however, as it is becoming increasingly common that brokers are offering commission free trading, we decided not to include a transaction cost.

3.3.2 Risk Exposure

$$\text{Minimise: } RE = \frac{1}{T} \sum_{i=1}^{n} t_{i,sell} - t_{i,buy}, \tag{7}$$

where T is the trading window, n is the number of trades, $t_{i,sell}$ is the date when sold for the ith trade and $t_{i,buy}$ is the date when bought for the ith trade. Risk exposure represents the amount of time that a trader is invested in the stock market. The longer a trader has a position in the market the greater the risk for the trader to obtain financial losses from unexpected events such as market crashes so it is desirable to minimise the risk exposure [10].

3.3.3 Performance Consistency

$$\text{Maximise: } PC(K) \tag{8}$$

The performance consistency (PC) value is achieved by splitting the training period into k successive periods and counting the number of times that the strategy performed well in each of these periods. The strategy is said to have performed well if its return has outperformed the B&H for the period. For my model, the k value will be set to 24 for all runs.

3.3.4 Number of Trades

$$\text{Minimise: } n, \text{ where } n > 0 \tag{9}$$

A trade is either a buy or a sell action. The number of trades, n, should be minimised as it can prevent over-trading from occurring and encourage the strategies to be disciplined with the signals that they indicate [8].

3.3.5 Sharpe Ratio

$$\text{Maximise: } SR = \frac{\sqrt{N}Rx}{StdDev(x)} \tag{10}$$

where N is the number of trading days for the strategy, Rx is the daily average return of the strategy which is in excess of the risk-free daily return and *StdDev(x)* is the standard deviation of the strategies returns. The risk-free return is will be set to 5% a year which is a low rate similar to a US Treasury Bond which is recognised as the safest investments that can be made. The Sharpe Ratio is used to evaluate the returns of a strategy in relation to the risk involved with it. A higher value for the Sharpe Ratio indicates that the strategy can achieve higher returns with a lower risk. The risk is calculated using the volatility of the returns [20], [27].

3.4 Objective Options

All 11 objective combinations considered in this work can be seen in table 3. For clarification, P stands for profit, PC for performance consistency, RE for risk exposure, NT for number of trades and SR for Sharpe ratio. For the remainder of this report, each combination of objectives will be referred to by the option name that is presented in table 3.

Option Name	No. Objectives	Objectives
P_PC	2	Profit and PC
PC_RE	2	PC and Risk Exposure
P_SR	2	Profit and Sharpe Ratio
P_RE	2	Profit and Risk Exposure
P_NT	2	Profit and no. trades
P_PC_RE	3	Profit, PC and Risk Exposure
P_PC_SR	3	Profit, PC and Sharpe Ratio
P_RE_NT	3	Profit, Risk Exposure and no. trades
P_PC_RE_NT	4	Profit, PC, Risk Exposure and no. trades
P_SR_RE_NT	4	Profit, Sharpe ratio, Risk Exposure and no. trades
PC_SR_RE_NT	4	PC, Sharpe ratio, Risk Exposure and no. trades

Table 3: Combinations of objectives used in this work.

4 Methodology

This research seeks to find efficient trading strategies by using elitist non-dominated sorting genetic algorithm-II (NSGA-II) [18] embedded with genetic programming [19]. How these techniques will be implemented and applied to finding trading strategies will be explained in the following section.

4.1 Genetic Programming Representation

Genetic programming evolves decision vectors in the form of variable-length strings that form tree-like structures. In this project trading strategies can be represented by these tree structures. Each tree structure during an evolution is known as an individual. A singular tree structure represents a program with each tree containing terminal nodes and operator nodes. The terminal nodes are the ends of the tree which can either be a constant or an argument. The constants are usually selected at random from a predefined set and the arguments are the program inputs [28]. The operator nodes process the data and can either be Boolean operators or a function, such as a technical indicator in this case [29]. Tables 4 and 5, show all the possible terminal and operator nodes that the genetic programming procedure can select from in the program developed for this project.

Name	Terminal type	Data types
Technical Indicator parameters	Constant	Integer
Date	Argument	Datetime
Position	Argument	Boolean

Table 4: The types of terminal node that can be selected during the genetic programming process.

Boolean Operators
gt
lt
and_
or_
not_
if_then_else

Technical Indicators
ma
ema
macd
rsi
so

Table 5: The possible operator nodes that can be selected during the genetic programming process. The left table shows the Boolean operator nodes and the table on the right displays the technical indicator operator nodes. The values in the tables represent the notation that has been used for the tree diagrams. For example, 'gt' is greater than, 'lt' is less than etc.

In table 4, the Date and Position are the arguments that an individual tree evaluates. These arguments change as the strategy progresses forwards in time. The Date argument is the current date that the strategy is making a trading signal for and the Position argument is the Boolean representing whether that strategy currently holds a position in the stock (owns the stock).

Strongly-typed genetic programming will be implemented for this project as it can ensure that only certain data types can be passed to the operator nodes [30]. The specific terminal nodes that relate to the various operator nodes can be explicitly forced to connect to each other enforcing the type constraints. This is important for this problem as it is how we can ensure that the pre-defined parameters are matched to their relevant technical indicators. This form of strongly-typed genetic programming will be implemented using the python library DEAP [28].

To help explain how the tree structures can be represented, I have included some examples. The representation of a tree structure for two example trading strategies can be seen in figure 1. From the tree structure on the left in figure 1, we can observe the various operator and terminal nodes and their notations as defined in tables 4 and 5. This tree structure can be described in more readable terms by the following:

If the position is true, sell if the 7-day RSI is greater than 64,
Else if the position is false, buy if the 7-day RSI is less than 54

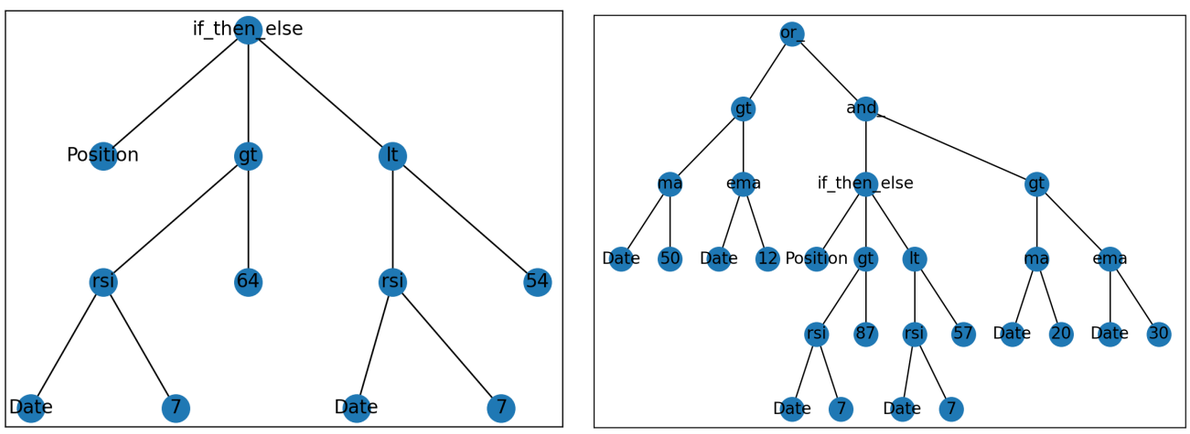

Figure 1: Two examples of genetic programming tree structure representations. Two random trees have been selected to display the varying complexity and size of trees that can be generated.

4.2 Tree Size

The tree size is known as the depth of the tree or the number of layers that can be present in a tree. For instance, the tree on the right in figure 1 has a depth of 6. In DEAP the size of tree structures that are developed can be limited so that they can only be produced up to a defined depth during the evolution process. Lohpetch & Corne [13] found improvements to their results when they

introduced a factor that promotes trees with smaller depths to be evolved. These improved results occurred because the factor was able to reduce the complexity of the trees which in turn reduced the effects of overfitting from occurring. Therefore, for the system that has been designed for this project, a limit of depth 6 for the tree size has been chosen.

4.3 Genetic Programming Operation

The process of the evolution of genetic programs follows a distinct flow of operation. The flow of operation is outlined by the following [29] [15]:

1. **Initialisation**. This randomly generates the initial population of tree structures. The size of the population will be pre-defined and the operation of randomly generating tree structures is controlled using the library, DEAP.

2. **Evaluation**. This evaluates the population and obtains the fitness values of each tree structure. In multi-objective scenarios, there will be multiple fitness measurements calculated for each individual at this stage. The NSGA-II selection operation as described in section 4.4 will be used to select individuals from the population.

3. **Modification**. This creates the offspring population using crossover and mutation. Both crossover **and** mutation are applied to each individual depending on the probability that is defined.

   (a) **Crossover** is the process of combining two different individuals in the population to form two new offspring. Random nodes in the tree structures are selected to form the crossover point. Anything proceeding these nodes in the tree structure is swapped with the corresponding individual [15]. A crossover probability is pre-defined. This is the probability for each individual that the crossover operation will occur.

   (b) **Mutation** is the operation that randomly makes modifications to the tree structures. This helps to encourage the diversity of the population. Random nodes in the tree are selected and then the sub-tree preceding that node is removed and then a new randomly generated sub-tree replaces it [29]. A mutation probability is also pre-defined which controls the likelihood of each individual mutating.

Steps 2 and 3 are repeated until the max number of generations has been reached.

4.4 NSGA-II

The genetic programming operation is to be combined with the NSGA-II selection operation. NSGA-II is a multi-objective genetic algorithm that was developed by Deb et al. [18]. NSGA-II has been chosen as it is computationally efficient and has a strong ability to achieve convergence near the theoretical Pareto-optimal front [31]. Additionally, the ability of the algorithm to achieve a good diversity and range of solutions has been found to be better than most other multi-objective evolutionary algorithms such as SPEA-2 and PAES [32]. It will be implemented using the DEAP library as they provide a selection method that can perform the relevant procedures of NSGA-II. The NSGA-II selection process that will be used for this research is described by the following points:

- The NSGA-II ranks and classifies each of the solutions in the population into an exclusive non-dominated set. It achieves this by first ranking the original non-dominated set. This original set is ignored then the next non-dominated set is classified. This process is repeated until every solution has been classified into a set (also called fronts).

- It then uses a diversity-preserving strategy by using the crowding distance concept. The density of solutions surrounding a solution is estimated. This is done by calculating the average distance of the two nearest solutions, in the same set, on either side of the solution. This value is the crowding distance. The crowding distance operation is used to ensure the diversity of the solution.

- The NSGA-II then takes into account the non-domination rank and the crowding distance of the solution during the selection process. This process uses a binary tournament selection between individuals which is based on the non-dominated rank and the crowding distance estimate.

- The population that has been selected then produces offspring using the genetic programming modification operations as described in section 4.3.
- Finally, the current population and the offspring that have just been produced are ranked again using the non-domination ranking procedure and only the best individuals up to the population size limit are chosen to produce the next generation.

This whole operation can help to maintain a diversity of solutions whilst providing good performance and the elitism helps to prevent already found Pareto-optimal solutions from being removed from the population [33].

4.5 Fitness Function Implementation

The fitness function for the trading strategies returns the objective values that evaluate the performance of the individual. The fitness function acts as a measure of performance for the strategies. It returns a fitness value for each objective that is used by NSGA-II to optimise the strategies. For every generation, the fitness function takes in each individual tree structure as an argument and evaluates the tree structure program for each day during a pre-defined time period.

When the tree is evaluated on a particular date, a Boolean is returned from the tree. If the tree equates to true then the trading signal is to buy and if it equates to false the trading signal is to sell. If the strategy already has a position in the market and the tree is indicating a buy signal, then the respective action is a hold and the strategy will not execute a trade. Additionally, if the strategy does not have a position in the market and the strategy is indicating a sell signal, then the respective action is a wait and the strategy will not execute a trade. A tree is then evaluated on every trade-able day within a window and subsequently, the trading performance of an individual tree can be recorded. These performance metrics are what forms the basis of the fitness function and objectives.

5 Experimental Design

5.1 Data Windows

The data has been retrieved from Yahoo Finance via the yfinance API and is the daily closing price of Microsoft (MSFT). The API allows the timescales of this data to be varied according to the tests that are being deployed. All of the technical indicators can then be derived from the daily closing price of MSFT stock. The technical indicators are pre-computed and appended to a CSV file in order to improve the runtime of the algorithm. Throughout all tests in this project, trading decisions are made daily as the resolution of the data that is downloaded is one day (the closing price).

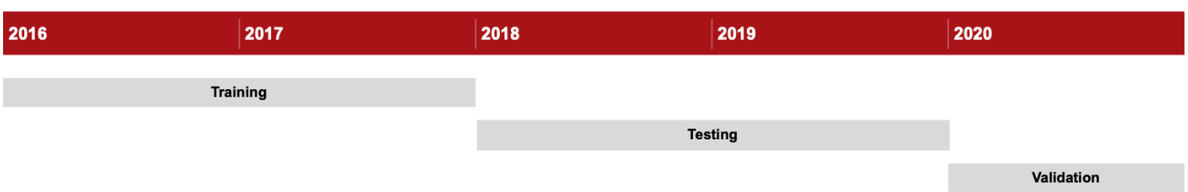

Figure 2: Example representation of the training, testing and validation data windows.

The program splits the data windows into three distinct groups: training, testing and validation. Strategies are developed using the evolutionary process (NSGA-II with genetic programming) as described in section 4 during the training window. The output from this data window is a Pareto front which indicates the trade-offs between multiple objectives. All of the strategies located on the Pareto front are then run on the testing window. This window is an unseen window and the performance of each strategy is recorded. Finally, the best strategy from the testing window is selected to be run on the validation window. The result of this final strategy is used for the results to be analysed in the results section of this report. The reasoning for the testing and validation data windows is explained in the following subsection (section 5.2).

5.2 Multi-Criteria Decision Making

An issue with multi-objective optimisation is that it still leaves a decision for the user to make from a set of solutions which can be a difficult task. The size of the Pareto front and the number of strategies on it can increase when more objectives are used. This makes it even harder to pick a relevant strategy as there are more options and more trade-offs to consider. There are several techniques that can be deployed to help make this process easier. This is known as multi-criteria decision making [34].

Allowing a singular strategy to be developed allows the generalisation performance to be evaluated. The process of developing a single final trading strategy is important as it can identify how applicable these techniques can be for developing strategies in a real-world scenario and on unseen data. The Pareto front may often be very large, particularly when there are a higher number of objectives, which makes the process of choosing a strategy from the front more complicated and difficult to do effectively. Whilst it can be a useful tool to aid a trader in selecting a strategy depending on the preferences of the objectives, it does not provide a singular useful strategy that can be used to yield profitable results.

For this work, I have explored using two different objectives for selecting the best strategy from the testing data window. The strategies with the highest profit and the highest performance consistency will be selected and both of these strategies will be run on the validation data window. Often the strategy that has the highest profit is also the strategy with the highest performance consistency, so in these cases, just this one strategy is run against the validation window. The testing and validation process provides an effective method for choosing one strategy from the Pareto front. The process of selecting the best strategy (with regards to profit and PC) on the testing data window is the multi-criteria decision making process that is being deployed for this project. Lohpetch and Corne [13] used this method by selecting the best strategy in terms of profit, which yielded good results for them. This project will extend their approach by looking at both profit and PC as selecting factors and will compare them to see which is more effective.

5.3 Tests

Four separate tests were carried out. They explored using all 11 objective options defined in table 3 and each test varied the training and testing windows. How the training and testing windows were varied can be seen in table 6. The validation window was kept constant for each of the tests mentioned in table 6 in order to provide a reliable comparison between the tests and to measure their performance relatively to each other. This validation window was between 01/01/2020 to 01/01/2021.

Test	Window size	Training window	Testing window
Test 1	6 months	01/01/2019 - 01/06/2019	01/06/2019 - 01/01/2020
Test 2	1 year	01/01/2018 - 01/01/2019	01/01/2019 - 01/01/2020
Test 3	2 years	01/01/2016 - 01/01/2018	01/01/2018 - 01/01/2020
Test 4	3 years	01/01/2014 - 01/01/2017	01/01/2017 - 01/01/2020

Table 6: The data windows for each test conducted in this paper. The validation window was set between 01/01/2020 to 01/01/2021 for all tests.

These tests will allow for analysis of the effects of changing the training and testing window sizes on the generalisation performance of the found strategies. It will also allow analysis for using the different objective options and using different numbers of objectives. Furthermore, the overall generalisation performance of the proposed algorithm can be analysed.

5.4 Parallel Computing

The use of parallel processing was used and optimised to speed up the evolutionary process by using the library, DEAP [28] along with the parallel computing library, Scoop. Using machines with a higher number of CPUs dramatically reduced the runtime of the algorithm. Each individual in a generation could be assigned to a CPU and its fitness could be computed using this CPU. This process could occur concurrently for each individual allowing the runtime to decrease as there was less time for individuals needing to wait for a CPU to become free. This is a major advantage of evolutionary algorithms and can make them more efficient as a result.

5.5 Numerical Settings

Evolutionary computing has a number of parameters that need to be set that can affect the performance of the process. The optimal parameters can vary depending on the problem domain and the search space that is being analysed [35]. The combinations of mutation and crossover probabilities are often considered. The mutation probability allows for greater exploration of the search space whereas the crossover probability allows for greater exploitation, allowing solutions to better converge to a specific point that is ideally an optimum [36]. For instance, in some specific problems having a higher mutation rate can allow for greater diversification in the search space and prevent the chances of premature convergence from occurring where solutions can become stuck at local optima [35]. However, in other problems, a high mutation rate could prevent solutions from converging at an optimum. Evidently, the relevant parameters greatly depend on the particular problem and choosing the correct parameters is a difficult process that takes careful consideration.

Population size	Number of generations	Crossover rate	Mutation rate
150	100	0.4	0.5

Table 7: The parameters for the evolutionary process.

For this project, the crossover probability was set to 0.4 and the mutation probability to 0.5 for all runs. These values were selected using trial and error after looking at a range of parameter combinations and analysing the hypervolume performance to ensure that convergence gets achieved. This combination achieved good results and convergence around a potential near optimum was usually achieved, as can be seen in figure 3 and 4. Future work could look into adopting hyper-parameter optimisation to systematically calculate the optimal parameters for this specific problem.

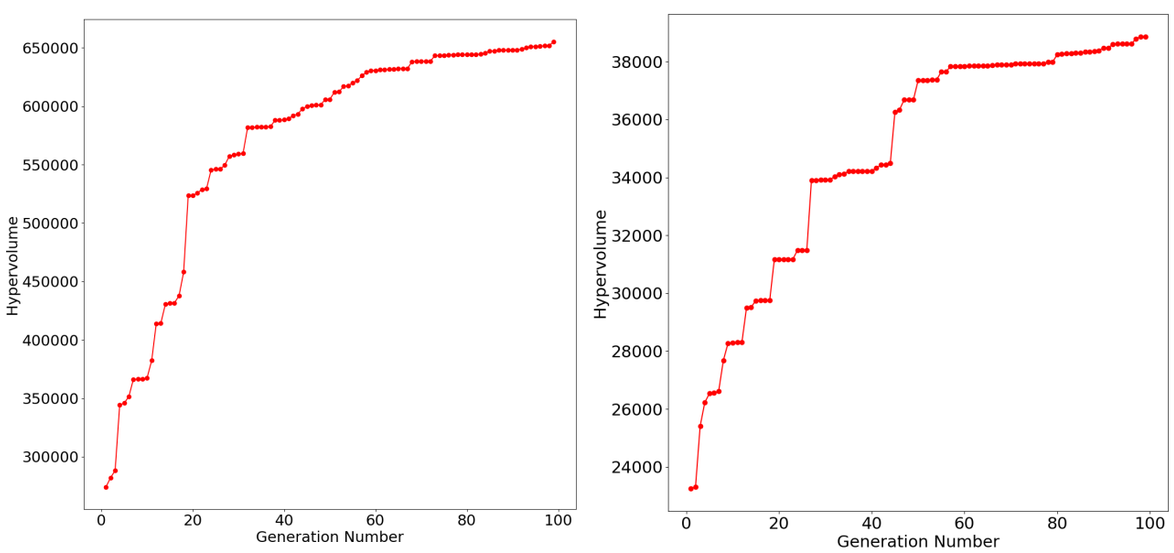

Figure 3: Two examples of hypervolume plots for two separate runs during test 3. They provide clear analysis of the performance of the evolution and enable us to observe and check that convergence has started to occur. The graph on the left was using objective option P_PC_RE (3 objectives) and on the right was with objective option P_RE (2 objectives).

The generation number has been set to 100 and the population size has been set to 150. When running the program the hypervolume is recorded and plotted against the generation number. The hypervolume is the area that is enclosed within the Pareto curve and measured from a defined reference point [37]. Examples of this phenomenon being observed can be seen in figure 3. We can identify that if the hypervolume curve flattens then the strategies are no longer improving. After running the program several times I found that the curve flattened around the 80 to 90 generation mark. Therefore the generation number was set to 100 as this allowed enough generations to allow convergence to occur in most cases. The population size has also been set to 150 as this is a significantly large enough size to allow a range of initial solutions to be developed and evolved. Ideally, these parameters could be set as high as possible, however, computation costs and time also need to be considered when selecting these parameters. This is why these values are not higher. Using the parameters from test 1, option 8 (chosen at random) was run four times repeatedly to get an insight into the reliability and repeatability of the results after an evolution. The results can be seen in figure 4. After 100 generations the hypervolume of all runs arrives at a similar value

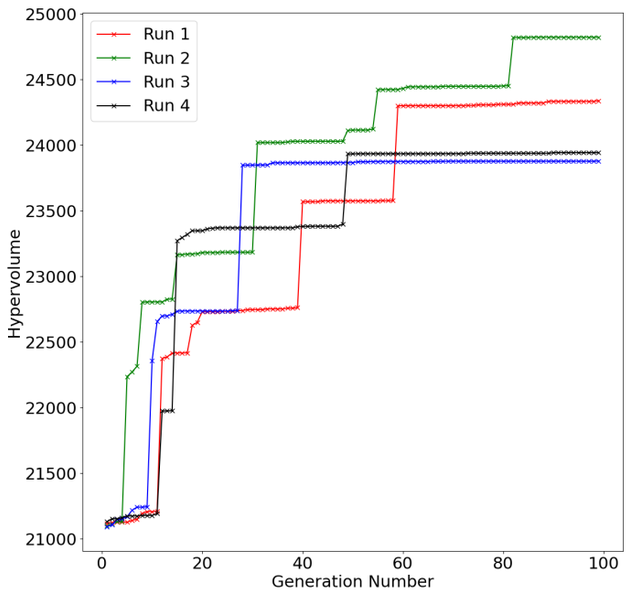

Figure 4: The hypervolume plot displaying four repeat runs of option P_NT for test 1 to test the reliability of the results.

within a range of approximately 750 units. These results and range is satisfactory for the purposes of this project and indicates that it is acceptable to run each option for each test once.

5.6 Analysis Technique

As mentioned previously the performance of the trading strategies will be evaluated against the B&H strategy. Results from the experiments will show the % change in profit from the B&H strategy. Any result with a positive % difference in profit is considered a successful strategy as it has outperformed the B&H. The purpose of using profit as the final analysis measurement is that it is the most important objective and is the predominant reason for trading in the stock market. The method proposed in this project intends to multi-objectivize an originally single objective problem in order to escape local optima. That single-objective being profit. Hence the % difference in profit above the B&H hold provides a suitable performance metric.

Results are collected over the three data windows mentioned previously: training, testing and validation. The performance of the strategies on the validation period are the results that are analysed in the results section. The validation data window is used for the analysis because it provides a good measure for the generalisation performance of the system and how well the system would do in a real-world scenario. This is because the validation data window analyses one single strategy that has been selected as the final result. Professional traders may find the testing data results more useful as they can use the Pareto front to make informed decisions based on their existing knowledge. The results displayed on the Pareto front would highlight the trade-offs between objectives which could help the decision making process. However, for this project, the performance from the validation data window is used to evaluate the performance of the multi-objective evolutionary process. This is because it provides a singular strategy at the end of the process which requires no human input or decision making and also provides a good evaluation for how well the system can find strategies that perform well in unseen environments.

6 Results and Analysis

6.1 Number of Objectives

For each test, the performance of using two, three and four objectives with different combinations mentioned in section 3.3 has been compared. The combinations have been labelled into options which can be seen in Table 3. The results for the average performance of the best strategies for each number of objectives are presented in Figure 5, where the orange line represents the average performance on the testing data and the blue line represents the average performance on the validation data.

We can observe that the performance increases with the number of objectives for the testing data but on the validation data the performance decreases as the number of objectives used

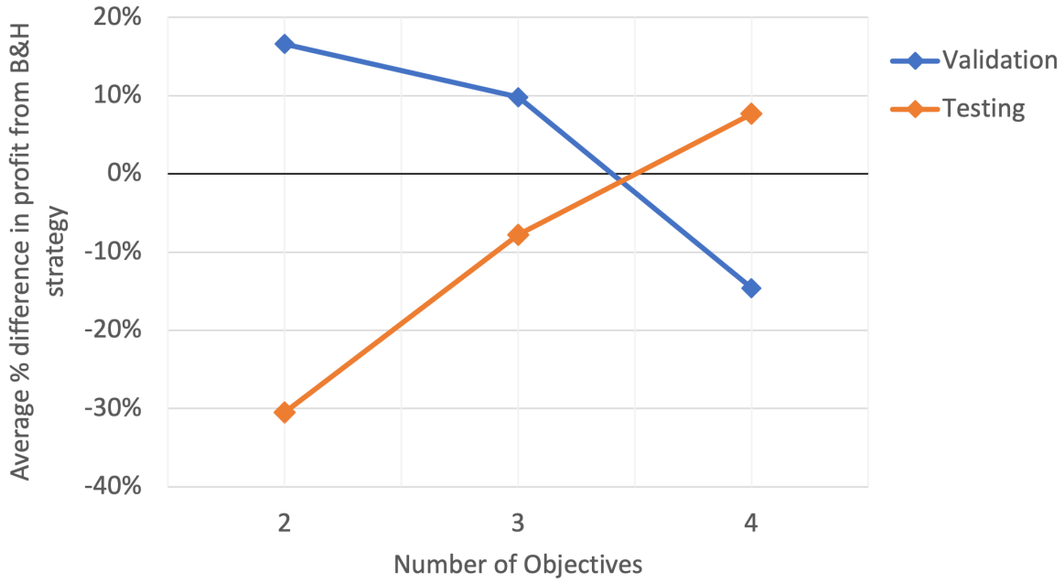

Figure 5: Graph showing how the average performance of all runs for all the tests outlined in table 6 changes when different numbers of objectives are used. The performance on both the testing and validation data sets are displayed.

increases. These results can indicate that using more objectives can aid in finding successful strategies on the Pareto front. However, this is the reverse for finding good results on the validation data window. This can imply that to find strategies that generalise better it would be recommended to use fewer objectives. Whilst it may appear to be useful to use more objectives to find better results on the testing data, it should be noted that the testing data result obtained in this graph is chosen after selecting the very best strategy from the Pareto front. Often the Pareto front can become very large and its size exponentially increases as the number of objectives increases. This effect can be seen visually in figure 6. Therefore, as this front has more options to select the best strategy from, it is more likely that a high-performing strategy can be selected. This is a potential explanation as to why we can observe this trend where the performance increases as the number of objectives increases on testing data. It's possible this would then over-fit strategies on the testing data. These strategies would consequently perform worse on the validation data, explaining why we see a poor performance with four objectives on the validation data.

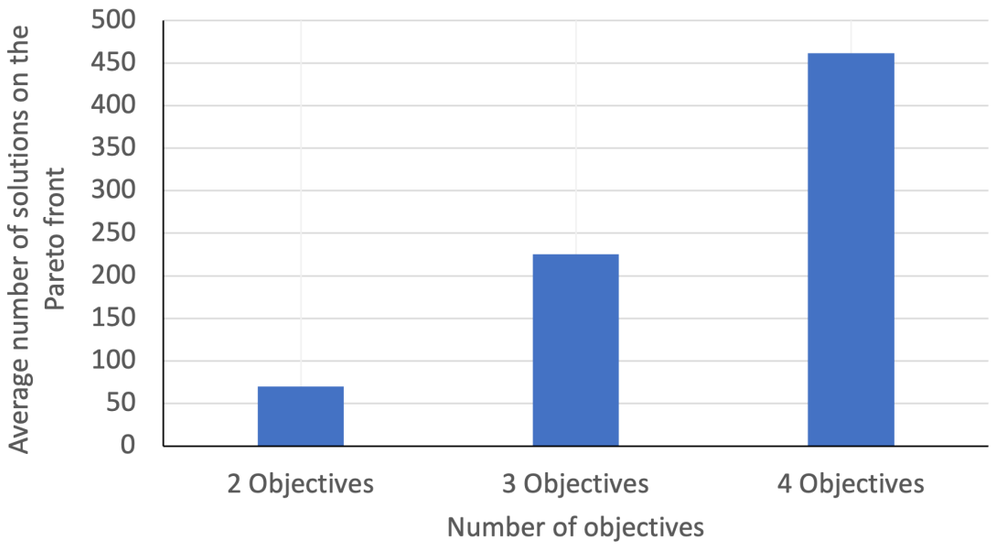

Figure 6: The average number of solutions on the Pareto front for each number of objectives. The averages have been taken from all runs on every test.

Another potential explanation for the reason that the validation data window performance decreases as the number of objectives increases is the effectiveness of NSGA-II on many-objective problems. Many-objectives is defined as being any multi-objective problem that has four or more objectives. The performance of NSGA-II is known to decrease as the number of objectives increases. This is because the majority of the solutions in the population become non-dominated when there are many-objectives, which weakens the dominance-based selection process in NSGA-II

[38]. This lowers the ability for the algorithm to achieve convergence and produces exponentially more solutions on the Pareto front. This reasoning could explain why the four objective options performed worse and perhaps for future work other more suited multi-objective algorithms should be used instead.

Due to the trend in the validation data results in figure 5, it can be suggested that it is more beneficial to use fewer objectives when finding strategies to perform well on unseen data. However, this excludes using just one single objective as Lohpetch and Corne [13] found that multi-objective methods outperformed all single-objective methods on validation data windows. Therefore, two objective methods appear to be the optimal number of objectives to use to evolve trading strategies using NSGA-II.

6.2 Objective Combinations

Each test carried out 11 separate runs, each using different objective options as defined in table 3. Figure 7 shows the average performance on the validation data window for each of these objective options over the four tests that were carried out. As indicated previously, any strategy that can outperform the B&H is a considerable success in terms of generating profitable strategies. From figure 7 we can see that options P_PC, PC_RE, P_NT, P_PC_RE, P_PC_SR and PC_SR_RE_NT were all able to find strategies that on average outperform the B&H. Looking at all options there is not a clear pattern as to which individual objective performs the best. The P_NT option achieved an average performance of 73% above the B&H which was the best out of all objective options. This could highlight the importance for strategies to be able to generate a high profit in few trades.

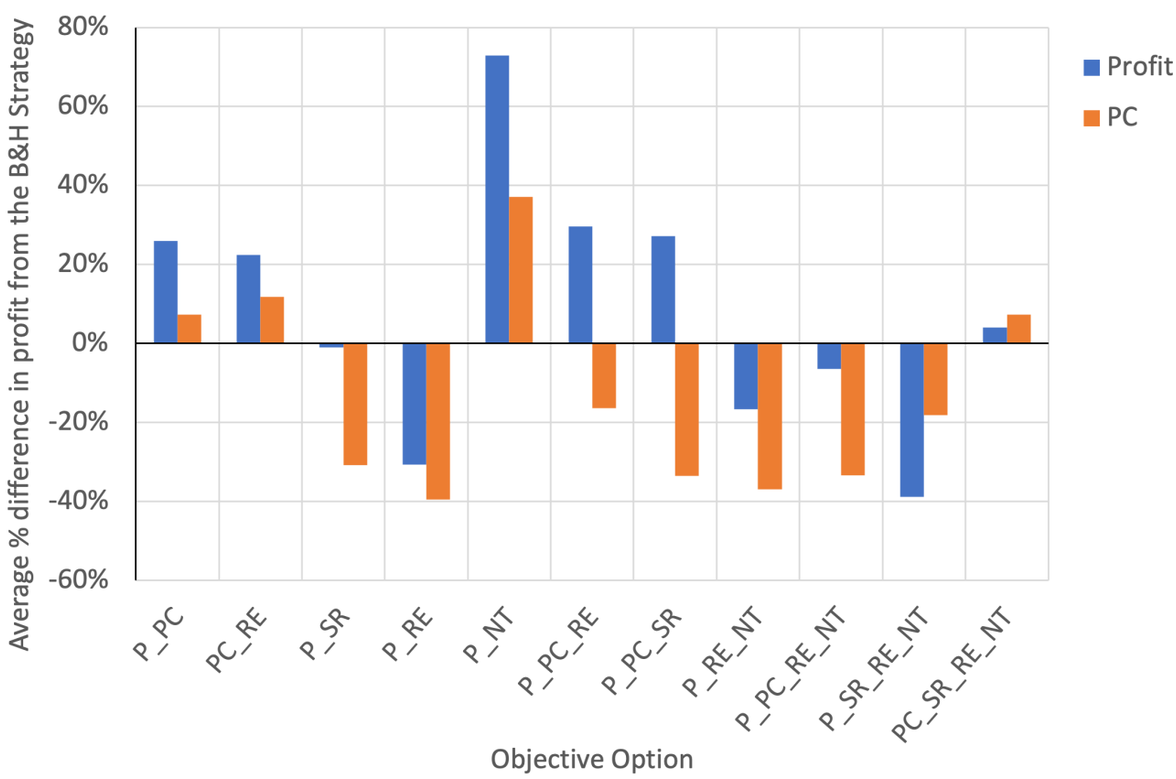

Figure 7: Bar chart showing the average performance of each objective option over the validation data window. The average has been taken from each test outlined in table 6. Both methods of selecting strategies from the Pareto front (either using profit or PC) are displayed.

Furthermore, this figure clearly indicates the better technique for selecting strategies from the Pareto front after they have been run on the testing data. In nearly all cases, except in option P_SR_RE_NT and PC_SR_RE_NT, using profit as a selecting factor can yield superior results on the validation data. This is an unexpected result as I believed that using the PC for this function would select the strategies that generalise better as they are performing more consistently. Evidently, this is not the case and the results suggest that using profit is a better solution. It is worth noting that out of all 44 runs that were carried out in this research, in 22 of them the strategy that achieved the highest profit on the testing data was also the strategy that achieved the highest performance consistency so the performance consistency is still a good option for selecting from the Pareto front.

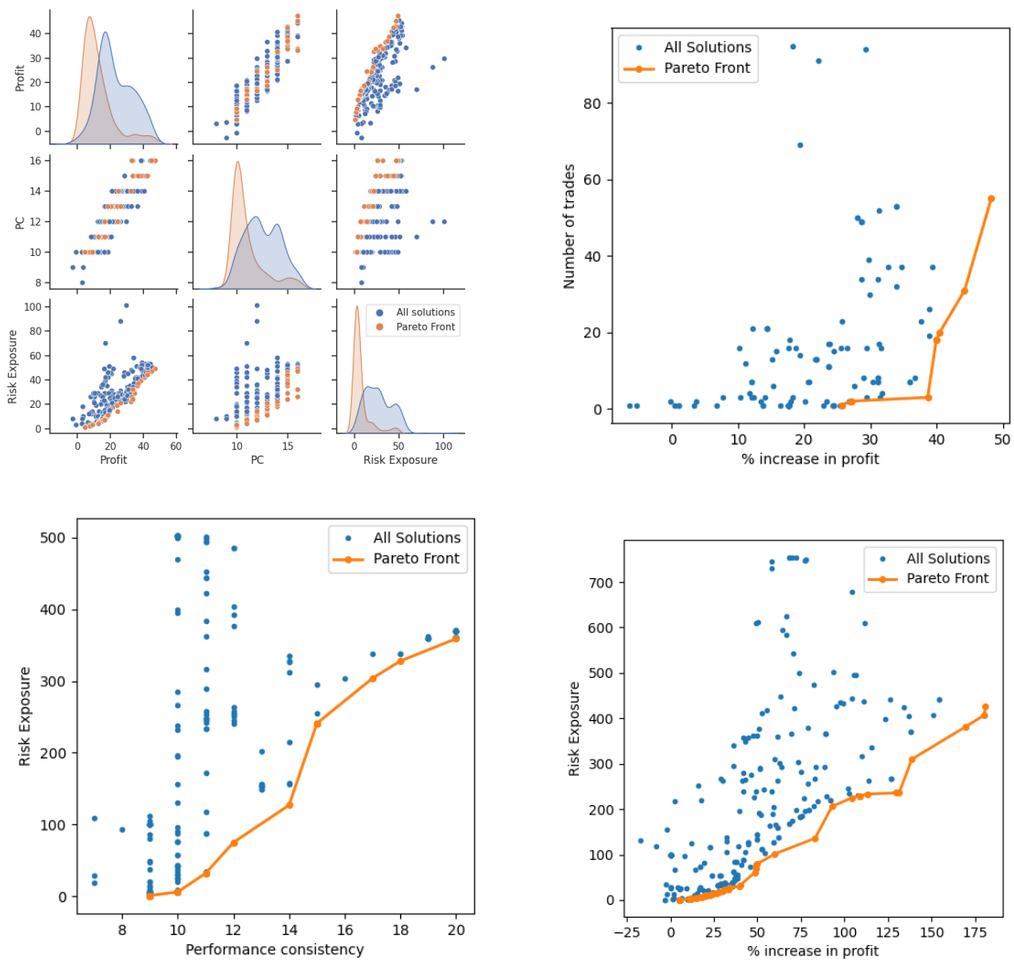

Figure 8: Scatter plots displaying the Pareto front obtained on the training data window for the best run of each test. Option P_PC_RE for test 1 (top left), option P_NT for test 2 (top right), option PC_RE for test 3 (bottom left) and option P_RE for test 4 (bottom right).

Figure 8 shows the solutions obtained after running NSGA-II on the training data for the best objective options in each test. A plot could be drawn for every run carried out in this research, however, due to the practicality of analysis, only four examples have been chosen. These plots can help us analyse the performance of NSGA-II and show the relationships and trade-offs between the objectives. A Pareto front has been obtained for each objective pairing which could allow a user to analyse and select a desired strategy depending on their preferences. From the figure, we can see that the Pareto fronts have a good level of diversity along the front. The Pareto fronts contain solutions that cover wide ranges for each objective. This is a good result for the use of NSGA-II as it is likely due to the crowding distance concept that is used by NSGA-II that enables a diversity of solutions to be explored.

The top left plot shows the best objective option for test 1. From this, we can see that there is a trade-off between profit and risk exposure as expected. An outer curve can be seen which clearly represents the Pareto front. However, with profit and PC, we can see that these objectives are not conflicting but have a positive correlation. Strategies that tend to achieve a higher profit also have a higher PC. In the remaining plots in figure 8: top right, bottom left and bottom right, we can observe that they all have conflicting objectives as clear Pareto fronts can be drawn. The plots in figure 8 show examples of how different objectives can be analysed against each other, which could allow a user to select strategies depending on their preferences.

6.3 Training and Testing Data Windows

From figure 9 we can seen that both 6 month training window and the 1 year training window on average out-performed the B&H. The 1 year window (test 2) significantly outperformed the B&H by 53% which is an impressive result. For test 2, the training window and the testing window was the same length as the validation window which perhaps allowed the strategies to perform better on the unseen validation data as all the windows were appropriately matched.

As markets progress forwards in time the volatility and behaviour of the markets can change. The reason that test 3 and 4 show worse performance could be because their training and testing data windows are further away in time from the validation data window. This means that the

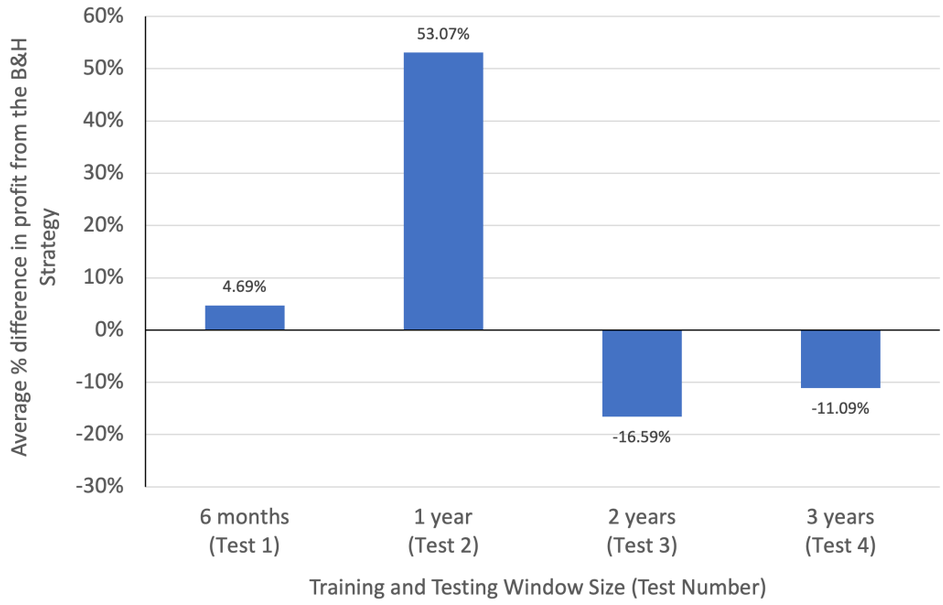

Figure 9: Bar chart displaying the average performance of the strategies for each test. These results are the % change in profit from the B&H strategy on the validation data set.

strategies are being developed in markets that could be showing different behaviour to that of the validation data window. Training and testing data windows that are closer to the validation data window could allow for more effective results as the data windows will be more related and show similar behaviour. If the training and testing data is too far in the past then it is likely that the strategies that worked well in the past no longer apply in the present. The strategies would need be refreshed and evolved with more up-to-date data to improve the performance.

Future work could perhaps explore this idea and match training and testing data to different lengths of validation data to see if similar results are achieved. If a strategy could adapt and continuously evolve then perhaps strategies could provide more robust longer term performance. The results could be highlighting this theory so this is an area that could be researched deeper into for future work in order to validate it.

6.4 Generalisation Performance

The generalisation performance is a measure of how effectively the strategies developed during the training and testing period can perform on out-of-sample, unseen data [39]. In this instance, the validation data window provides a suitable analysis point for measuring the generalisation performance. From figure 7 we can see that on average most of the strategies outperformed the B&H when using profit as a selecting factor. This shows great promise for the use of multi-objective evolutionary algorithms in developing useful strategies that can perform well. These results fall in line with some of the results in previous work [7], [10], [13], [14], as these previous papers are all able to report results where strategies can successfully outperform the B&H on unseen data using NSGA-II.

From the box plot in figure 10, we can see that the results from all the runs are spread above and below the B&H performance. The performance of individual strategies can be very varied as we can see from the wide spread of the results in figure 10. Looking at objective option PC_RE, after one run a strategy could achieve a result of 154% above the B&H but then on the next run the strategy could achieve 97% below the B&H. This indicates that clearly there still is a big risk that individual strategies developed may not perform well and be unable to outperform the B&H on some occasions. This inconsistency of results would be very concerning for someone using these methods and defeats the purpose of finding strategies that could consistently perform.

However, these risks could be mitigated and steps could be carried out to prevent the chances of failure. For instance, as the average for some of the objective options can outperform the B&H, a user could develop a number of strategies and deploy them all at once, splitting the investment value between each strategy. Then on average, there could be higher chances for success. It should also be noted that strategies that perform below the B&H still often make a profit and don't necessarily lose money for the user. From figure 10 we can see that no strategy lost money or made negative returns. We can see this as none of the strategies performed below -100%. Therefore, it can be argued that there is actually less risk as investors are not losing money.

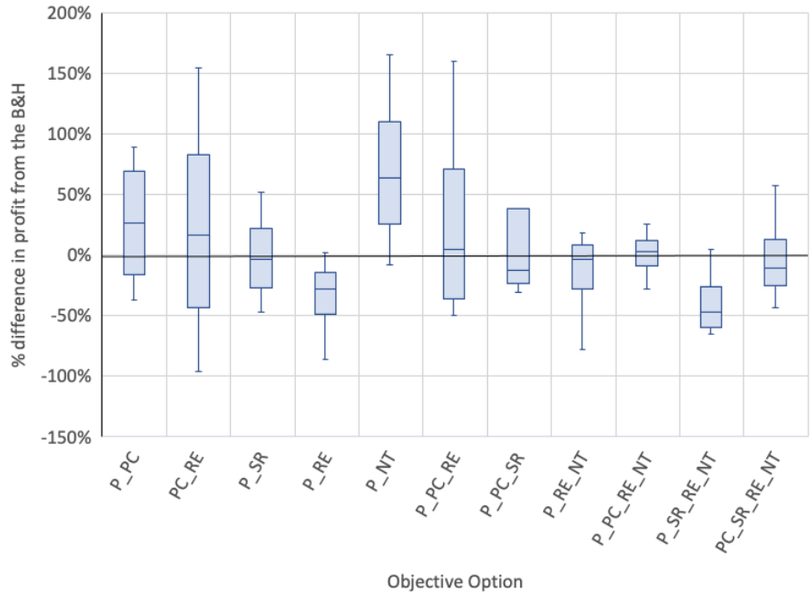

Figure 10: Box plot showing the spread of results for all runs (4 results for each objective option) on the validation data window. The horizontal line in the middle of the box plot represents the median and the upper and lower horizontal lines of the box plot indicate the upper and lower quartiles respectively. The outer whiskers of the box plots represent the maximum and minimum values for each objective option.

Evidently, there is an opportunity cost from missing out on the B&H returns, but this is a better result than achieving negative profits. The B&H % profit for the validation data was 38.48% which is a very commendable return for a year. For some strategies to perform on average 73% above the B&H (as seen in option P_NT), is a very good result for the system.

To further validate these results and the generalisation performance more validation windows and periods could be used. This is important as the specific data window could be very important for how well the strategies can perform against the B&H. In this case, as the validation period was over the year 2020, the strategies were deployed during one of the most turbulent years in the stock market and in this time one of the largest recessions in the history of the stock market occurred [40]. This fact could have been a reason for this success and perhaps the volatility is what makes the strategies perform well. This should be researched further by using a range of validation periods to ensure that anomalies are not occurring.

To further evaluate the generalisation performance, I have selected the best strategy out of all runs (Option P_NT, test 2). This strategy can be seen in figure 11.

if the 7-day stochastic oscillator and the 14-day stochastic oscillator are not greater and 0 then buy, otherwise sell.

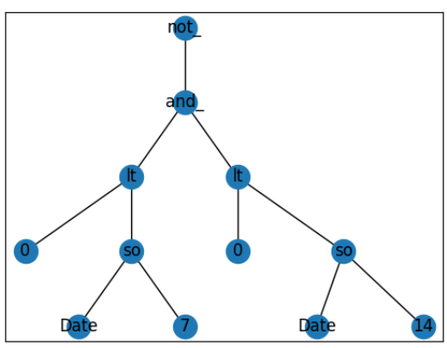

Figure 11: The strategy that performed the best on the validation data window out of all runs for all tests. This strategy was developed from test 2 using option P_NT.

This strategy managed to achieve a profit that was 164% above the B&H on the validation data window which equated to a profit increase of 101.92%. This is a very high return on investment within one year for a stock. With the B&H for MSFT during the validation data window being 38.48%, clearly the developed strategy has a considerably higher performance.

Figure 12 displays the buy and sell actions of this strategy. The graph is divided into coloured sections in relation to the performance consistency splits. The green areas indicate where the

strategy outperformed the B&H and the red areas show where it did not. There were a lot of trades during this window which makes figure 12 harder to decipher. Therefore figure 13 shows a closer look at this strategy over a smaller window so we see more clearly when trades were executed. The green ticks represent buy signals and the red ticks represent sell signals. Looking at this figure we can see how well the strategy is able to time trades and make profits even in a downward moving market. This window was during the recession in March 2020, and clearly this strategy was able to profit from this period. If an investor adopted the B&H strategy for this particular window in figure 13 then they would have made 0% returns on their investment. If we tally the profits from the strategies trades the sum is above 0%. Only one trade in this window made a loss.

These results show that the strategies found using NSGA-II can be highly successful and are able to suitably and efficiently generalise on unseen data. It shows that NSGA-II and genetic programming is able to find strategies that can perform very well.

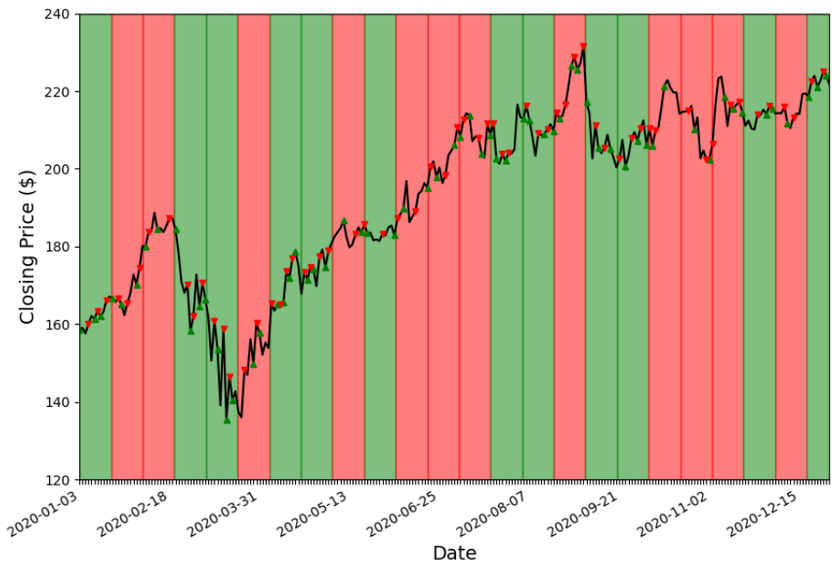

Figure 12: The trading performance on the validation data window for the best strategy (test 2, P_NT). The green regions indicate when the strategy out-performed the B&H and the red regions show when the strategy did not out-perform it. The green and red ticks indicate when buy and sell signals are generated respectively.

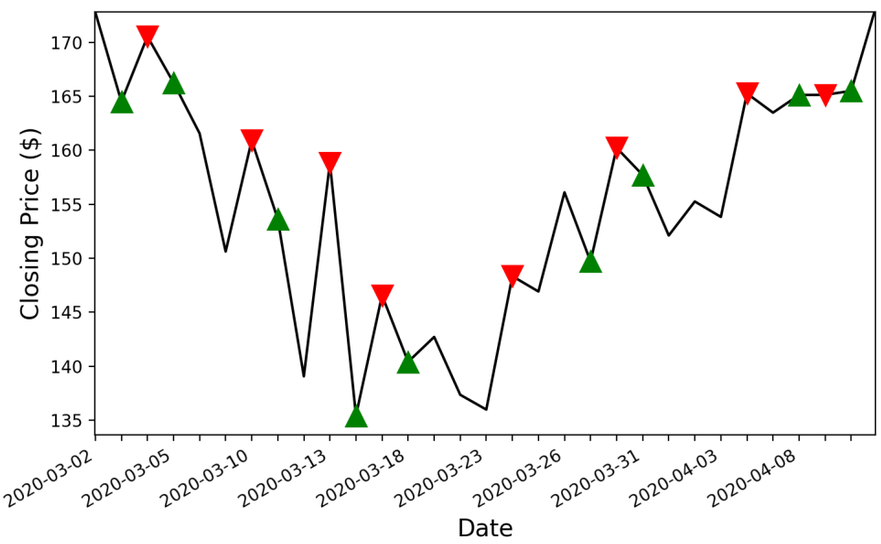

Figure 13: A closer view of the trading performance on a section of the validation data window for the best strategy (test 2, P_NT). The green and red ticks indicate when buy and sell signals are generated respectively.

7 Project Evaluation

7.1 Research Evaluation

The hypothesis for the project and the intended research question, as defined in my initial literature review, was *"can multi-objective evolutionary algorithms generate trading strategies that are successful by outperforming the B&H strategy?"*

From the results in this report, I can confidently answer this question and conclude that it is possible to use multi-objective evolutionary algorithms to produce trading strategies that can outperform the B&H, however not on a consistent basis. Extending the work of previous researchers and selecting techniques and practices that they found successful has likely aided the success of the system in generating trading strategies. Despite the inconsistency of the system in generating successful strategies, on average the strategies can outperform the B&H which shows great promise and steps can be used to mitigate the risks from the inconsistencies of the strategies.

As mentioned in my specification in my literature review, I had planned to experiment with the resolution of the data and to explore the performance when changing this parameter. The system that I have developed only analyses and makes decisions based on daily price data. With more time I could have changed this to hourly or even every minute and intra-day strategies could have been developed.

Additionally, I had planned to look at the implications of training strategies on different securities and industries, other than just Microsoft's stock price to see if the performance of the system relied on the underlying stock. This extra analysis is important to answer the research question on this report as a whole and would provide greater validation for the conclusions obtained. However, as the project progressed, I realised that in order to test and evaluate the performance of various parameters, timescale windows and objective combinations, I would need to ensure that the stock used remained constant to provide a fair evaluation. If the runtime of the algorithm was faster or if more time was available this research could have been carried out, however, I made the decision to focus on other topics of research for this problem and ensured that they were researched using an appropriate scientific manner. The runtime could often take approximately 12 hours or more when using a computer with 8 CPUs which made it more difficult to carry out a higher number of tests.

Originally, the effects of adding transaction costs for each trade was intended to be explored. Again however, I made the decision to exclude the transaction costs to keep the experiments fair and due to the increasing availability of commission-free trading.

7.2 Technical Evaluation

The code that I developed in Python has successfully enabled me to carry out this research into this problem. Using the external Python library DEAP, I was able to implement NSGA-II with genetic programming to produce valid trading strategies. Each of the predefined technical indicators were successfully developed and tested. These were then used when pre-computing the data. The objectives were able to be successfully calculated in the custom fitness function.

I developed a custom back-testing environment for my trading strategies to be used for the fitness function. I considered finding and using a external library that performs back-testing on stocks, and I believe that if I had opted for this option I would have saved more time during the development stage and reduced the risk of errors. However, I believe I made the correct decision to write my custom back-testing environment as it gave me far greater customisation opportunities which became very useful for implementing various objectives and testing procedures.

Results from the experiments and the representations of the strategies can be visually displayed which has helped aid the analysis of the results and performance of the system. The system has been optimised to have a short runtime. I successfully implemented parallel processing techniques in Python which dramatically improved the runtime of the system. Each individual in the population of a generation could be assigned to a CPU and each individual's fitness could be calculated concurrently. Using machines with a higher number of CPU's was very beneficial during the result collecting stage.

8 Conclusion and Future Work

This report looked into the performance of using genetic programming to evolve trading strategies using NSGA-II. It used a range of objectives and has analysed the performance of each objective option. At the beginning of the report, in section 2.2, four research areas were indicated. The following paragraphs highlight the conclusions made in each of these areas and the future research ideas that has arisen as a result:

The performance of the strategies when changing the number of objectives used. The results indicated that using fewer objectives could help find strategies that can perform well on the validation data window, meaning that they better find strategies that can generalise better. It is suggested that this is because using more objectives creates Pareto fronts with a greater number of solutions on it, which then causes overfitting when selecting from the Pareto front using the testing, validation approach. Future research could aim to confirm this theory and then perhaps look for ways to reduce this problem. For instance the number of strategies from the Pareto front that are run on the testing data window could be limited. Additionally, other multi-objective algorithms that maintains performance with a higher number of objectives could be explored as the performance of NSGA-II decreases as the number of objectives increases [38]. This could be limiting the performance of the 4 objective results, explaining the trend that our results show.

The performance of the strategies, using 11 different multi-objective combinations. The performance of each combination has been analysed in this report and plots showing the trade-offs between some of the objectives has been displayed. The results found that six out of the eleven combinations were able to outperform the B&H on average and the multi-objective combination of profit and number of trades performed the best.

Using different timescales for training and testing data windows. Tests with a shorter training and testing windows performed better than the splits which were longer. It is potentially because the training window is closer in time to the validation window which means that the strategies are trained on more related data. Future work could test this idea and match different length training and testing windows to the validation window. If this theory is correct, a system could be developed that regenerates the strategies after set intervals in order to ensure that the strategies remain up-to-date. This could potentially yield very successful long-term results and would be interesting to research further.

The generalisation performance and if strategies can be found that outperform the B&H strategy. Results collected suggest that strategies on the Pareto front can considerably outperform the B&H strategy on unseen testing data for various objective option combinations. Additionally, the best strategies from the Pareto front are able to outperform the B&H on further unseen validation data confirming the strong generalisation performance. This shows the potential for using these techniques to help aid the selection of relevant strategies. However, a limitation of this work is the consistency of performance of the strategies that are found. On average, the majority of the objective options can outperform the B&H, however, there is a large spread of results within this average. Techniques have been suggested that could help mitigate the effects of these inconsistencies, such as choosing the entire range of found strategies and splitting the value invested between each of them. These suggestions should be explored in future work to confirm if this is a viable method.

Future research could also test different multi-criteria decision making techniques when selecting strategies from the Pareto Front. Although the technique used in this research provided effective and useful results, it was a basic approach and was limited in scope: this project only used profit and PC as selecting factors after running on testing data. Various techniques could be deployed such as the Utopian point method [41] or fuzzy rule-based selection criteria [42]. Using different methods may be able to select strategies that can generalise better so this is an area that is worth researching. Finally, to validate the performance of the system, repeat tests on different stocks in different sectors of the economy should be run. This is because trading strategies may vary their performance as a function of the market considered. Carrying out the tests on more stocks will ensure the system is robust and can provide useful results in a range of market conditions.

References

[1] F. Allen and R. Karjalainen, "Using genetic algorithms to find technical trading rules," *Journal of Financial Economics*, vol. 51, no. 2, pp. 245–271, 1999.

[2] J. Chen, *Trading Strategy*, Apr. 2019. [Online]. Available: `https://www.investopedia.com/terms/t/trading-strategy.asp`.

[3] D. Iskrich and D. Grigoriev, "Generating long-term trading system rules using a genetic algorithm based on analyzing historical data," in *Conference of Open Innovation Association, FRUCT*, IEEE Computer Society, 2017, pp. 91–97.

[4] B. M. Barber, Y.-T. Lee, Y.-J. Liu, and T. Odean, "Do Individual Day Traders Make Money? Evidence from Taiwan," Tech. Rep., 2004.

[5] E. F. Fama, "Efficient Capital Markets: A Review of Theory and Empirical Work," *The Journal of Finance*, vol. 25, no. 2, pp. 383–417, 1970.

[6] C. Fyfe, J. P. Marney, and H. Tarbert, "Risk adjusted returns from technical trading: a genetic programming approach," *Applied Financial Economics*, vol. 15, no. 15, pp. 1073–1077, 2005.

[7] A. Pimenta, C. A. Nametala, F. G. Guimarães, and E. G. Carrano, "An Automated Investing Method for Stock Market Based on Multiobjective Genetic Programming," *Computational Economics*, vol. 52, no. 1, pp. 125–144, 2018.

[8] R. L. Weissman, *Mechanical Trading Systems: Pairing Trader Psychology with Technical Analysis*. Wiley Trading, 2005.

[9] W. Brock, J. Lakonishok, and B. LeBaron, "Simple Technical Trading Rules and the Stochastic Properties of Stock Returns," *The Journal of Finance*, vol. 47, no. 5, pp. 1731–1764, 1992.

[10] S. C. Chiam, K. C. Tan, and A. Al Mamun, "Investigating technical trading strategy via an multi-objective evolutionary platform," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10 408–10 423, 2009.

[11] J. Lee and N. Sabbaghi, "Multi-objective optimization case study for algorithmic trading strategies in foreign exchange markets," *Digital Finance*, vol. 2, no. 1-2, pp. 15–37, 2020.

[12] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Inc., 2001.

[13] D. Lohpetch and D. Corne, "Multiobjective algorithms for financial trading: Multiobjective out-trades single-objective," in *2011 IEEE Congress of Evolutionary Computation*, 2011, pp. 192–199.

[14] G. V. Farnsworth, J. A. Kelly, A. S. Othling, and R. J. Pryor, "Sandia Report Successful Technical Trading Agents Using Genetic Programming," Tech. Rep., 2004, pp. 3–20.

[15] C. S. Lee and K. Y. Loh, "GP-based optimisation of technical trading indicators and profitability in FX market," in *ICONIP 2002 - Proceedings of the 9th International Conference on Neural Information Processing: Computational Intelligence for the E-Age*, vol. 3, 2002, pp. 1159–1163.

[16] D. Lohpetch and D. Corne, "Discovering effective technical trading rules with genetic programming: Towards robustly outperforming buy-and-hold," in *2009 World Congress on Nature and Biologically Inspired Computing, NABIC 2009 - Proceedings*, 2009, pp. 439–444.

[17] L. A. Becker and M. Seshadri, "Comprehensibility & Overfitting Avoidance in Genetic Programming for Technical Trading Rules," Tech. Rep., 2003.

[18] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[19] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge: The MIT Press, 1992.

[20] A. C. Briza and P. C. Naval, "Stock trading system based on the multi-objective particle swarm optimization of technical indicators on end-of-day market data," *Applied Soft Computing Journal*, vol. 11, no. 1, pp. 1191–1201, 2011.

[21] J. Desjardins, *12 Types of Technical Indicators Used by Stock Traders*, May 2017. [Online]. Available: `https://www.visualcapitalist.com/12-types-technical-indicators-stocks/`.

[22] C. Mitchell, *How to Use a Moving Average to Buy Stocks*, Aug. 2020. [Online]. Available: https://www.investopedia.com/articles/active-trading/052014/how-use-moving-average-buy-stocks.asp.

[23] N. Lioudis, *Using EMA in a Forex Trading Strategy*, May 2020. [Online]. Available: https://www.investopedia.com/ask/answers/122314/how-do-i-use-exponential-moving-average-ema-create-forex-trading-strategy.asp.

[24] A. Hayes, *Moving Average Convergence Divergence (MACD) Definition*, Oct. 2020. [Online]. Available: https://www.investopedia.com/terms/m/macd.asp.

[25] J. Chen, *Relative Strength Index – RSI Definition & Calculation*, Aug. 2020. [Online]. Available: https://www.investopedia.com/terms/r/rsi.asp.

[26] A. Hayes, *Stochastic Oscillator Definition*, Jun. 2020. [Online]. Available: https://www.investopedia.com/terms/s/stochasticoscillator.asp.

[27] J. Fernando, *Sharpe Ratio Definition*, 2020. [Online]. Available: https://www.investopedia.com/terms/s/sharperatio.asp.

[28] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP: Evolutionary Algorithms Made Easy," *Journal of Machine Learning Research*, pp. 2171–2175, Jul. 2021.

[29] J. Y. Potvin, P. Soriano, and V. Maxime, "Generating trading rules on the stock markets with genetic programming," *Computers and Operations Research*, vol. 31, no. 7, pp. 1033–1047, 2004.

[30] D. J. Montana, "Strongly Typed Genetic Programming," *Evolutionary Computation*, vol. 3, no. 2, pp. 199–230, Jun. 1995.

[31] K. Deb, "Multi-objective Optimisation Using Evolutionary Algorithms: An Introduction," in *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*, Springer London, 2011, pp. 3–34.

[32] C. Jianling, "Multi-objective optimization of cutting parameters with improved NSGA-II," in *Proceedings - International Conference on Management and Service Science, MASS 2009*, 2009.

[33] C. A. Coello Coello, "Evolutionary Multi-Objective Optimization in Finance," in *Handbook of Research on Nature Inspired Computing for Economy and Management*, vol. 1, Idea Group Reference, 2006, pp. 74–88.

[34] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Second Edition. Springer US, 2007.

[35] O. Boyabalti and B. Sabuncuoglu, "Parameter Selection in Genetic Algorithms," *Journal of Systemics, Cybernetics and Informatics*, vol. 4, no. 2, pp. 78–83, 2004.

[36] S. M. Lim, A. B. M. Sultan, M. N. Sulaiman, A. Mustapha, and K. Y. Leong, "Crossover and Mutation Operators of Genetic Algorithms," *International Journal of Machine Learning and Computing*, vol. 7, no. 1, Feb. 2017.

[37] A. Auger, J. Bader, D. Brockhoff, and E. Zitzler, "Theory of the hypervolume indicator: Optimal μ-distributions and the choice of the reference point," in *Proceedings of the 10th ACM SIGEVO Workshop on Foundations of Genetic Algorithms, FOGA'09*, New York, New York, USA: ACM Press, 2009, pp. 87–102.

[38] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary many-objective optimization: A short review," in *2008 IEEE Congress on Evolutionary Computation, CEC 2008*, 2008, pp. 2419–2426.

[39] C. Sammut and G. I. Webb, "Generalization Performance," in *Encyclopedia of Machine Learning*, Springer US, 2011, pp. 454–454.

[40] A.-L. Jackson and B. Curry, *2020 Stock Market Year in Review – Forbes Advisor*. [Online]. Available: https://www.forbes.com/advisor/investing/stock-market-year-in-review-2020/.

[41] A. Szparaga, M. Stachnik, E. Czerwińska, S. Kocira, M. Dymkowska-Malesa, and M. Jakubowski, "Multi-objective optimization based on the utopian point method applied to a case study of osmotic dehydration of plums and its storage," *Journal of Food Engineering*, vol. 245, pp. 104–111, Mar. 2019.

[42] S. Voget and M. Kolonko, "Multidimensional Optimization with a Fuzzy Genetic Algorithm," *Journal of Heuristics*, vol. 4, no. 3, pp. 221–244, 1998.