
Fuzzy-Evolutionary Modeling for Single-Position Day Trading

Célia da Costa Pereira and Andrea G. B. Tettamanzi

Università Degli Studi di Milano
Dipartimento di Tecnologie dell'Informazione
Via Bramante 65, I-26013 Crema (CR), Italy
E-mail: celia.pereira@unimi.it, andrea.tettamanzi@unimi.it

Summary. This chapter illustrates a data-mining approach to single-position day trading which uses an evolutionary algorithm to construct a fuzzy predictive model of a financial instrument. The model is expressed as a set of fuzzy IF-THEN rules. The model takes as inputs the open, high, low, and close prices, as well as the values of a number of popular technical indicators on day t and produces a *go short, do nothing, go long* trading signal for day $t + 1$ based on a dataset of past observations of which actions would have been most profitable.

The approach has been applied to trading several financial instruments (large-cap stocks and indices): the experimental results are presented and discussed.

A method to enhance the performance of trading rules based on the approach by using ensembles of fuzzy models is finally illustrated.

The results clearly indicate that, despite its simplicity, the approach may yield significant returns, outperforming a buy-and-hold strategy.

1 Introduction

Single-position automated day-trading problems (ADTPs) involve finding an automated trading rule for opening and closing a single position¹ within a trading day. They are a neglected subclass of the more general automated intraday trading problems, which involve finding profitable automated technical trading rules that open and close positions within a trading day.

An important distinction that may be drawn is the one between static and dynamic trading problems. A *static* problem is when the entry and exit strategies are decided before or on market open and do not change thereafter. A *dynamic* problem allows making entry and exit decisions as market action unfolds.

¹ A *position* is a non-zero balance for a financial instrument in a trader's account. Therefore, a long (short) position is opened by buying (short-selling) some quantity of a given financial instrument, and is closed by selling (buying back) an equal quantity of the same financial instrument.

Dynamic problems have been the object of much research, and different flavors of evolutionary algorithms have been applied to the discovery and/or the optimization of dynamic trading rules [21, 1, 15, 11, 12, 18, ?, 23, 28, 34].

Static problems are technically easier to approach, as the only information that has to be taken into account is information available before market open. This does not mean, however, that they are easier *to solve* than their dynamic counterparts.

This chapter will focus on solving a class of static single-position automated day-trading problems by means of a data-mining approach which uses an evolutionary algorithm to construct a fuzzy predictive model of a financial instrument. The model takes as inputs the open, high, low, and close prices, as well as the values of a number of popular technical indicators on day t and produces a *go short, do nothing, go long* trading signal for day $t+1$ based on a dataset of past observation of which actions would have been most profitable.

The chapter is organized as follows: Section 2 discusses how trading rules are evaluated and compared by investors; Section 3 states the problem addressed by this chapter, namely the static single-position day trading problem, and situates it within the context of trading problems. Next, the two basic tools used for approaching such problem are introduced: Section 4 provides a gentle introduction to fuzzy logic, with a particular reference to fuzzy rule-based systems, and Section 5 introduces evolutionary computation in general and distributed evolutionary algorithms in particular. Section 6 describes a fuzzy-evolutionary modeling approach to single-position day trading and the data used for modeling; in particular, several technical analysis indicators used by the approach are defined and briefly discussed. Section 7 reports the protocol and the results of experiments carried out to assess the approach and Section 8 discusses an ensemble technique to improve the performance of trading rules discovered by the approach. Section 9 concludes.

2 Evaluating Trading Rules

Informally, we may think of a trading rule R as some sort of decision rule which, given a time series $X = \{x_t\}_{t=1,2,\dots,N}$ of prices of a given financial instrument, for each time period t returns some sort of trading signal or instruction. The reason one might want to write such a trading rule is to consistently operate a strategy on a market, with the intent of gaining a profit. Instead of considering absolute profit, which depends on the quantities traded, it is a good idea to focus on returns.

2.1 Measures of Profit

For mathematical reasons, it is convenient to use log-returns instead of the usual returns, because they are additive under compounding. The average log-return of rule R when applied to time series X of length N is

$$r(R; X) = \frac{Y}{N} \sum_{t=1}^N r(R; X, t), \quad (1)$$

where $r(R; X, t)$ is the return generated by rule R in the t^{th} day of time series X , and Y is the number of market days in a year.

This is the most obvious performance index for a trading rule. However, as a performance measure, average log-return completely overlooks the risk of a trading rule.

2.2 Measures of Risk-Adjusted Return

Following the financial literature on investment evaluation [7], the criteria for evaluating the performance of trading rules, no matter for what type of trading problem, should be measures of risk-adjusted investment returns. The reason these are good metrics is that, in addition to the profits, consistency is rewarded, while volatile patterns are not. Common measures within this class are the Sharpe ratio and its variations, the Treynor ratio, Jensen's performance index, and the upside-potential ratio.

The main ingredients of all these measures are:

- the risk-free log-return r_f (in practice, one can use the log-return of short-dated government bonds of the currency in question);
- the average log-return of a rule, defined in Equation 1;
- the average log-return of time series X ,

$$r(X) = \frac{Y}{N-1} \sum_{t=2}^N r(X, t) = \frac{Y}{N-1} \sum_{t=2}^N \ln \frac{x_t^C}{x_{t-1}^C}, \quad (2)$$

where x_t^C is the closing price of the t^{th} day;

- the standard deviation of the log-returns (i.e., the risk) of rule R on X ,

$$\sigma(R; X) = \sqrt{\frac{Y}{N} \sum_{t=1}^N [r(R; X, t) - r(R; X)]^2}; \quad (3)$$

- the downside risk [13, 27] of rule R on X ,

$$\text{DSR}_\theta(R; X) = \sqrt{\frac{Y}{N} \sum_{t=1}^N [r(R; X, t) < \theta] [\theta - r(R; X, t)]^2}; \quad (4)$$

Sharpe Ratio and Variations

The Sharpe ratio [26] is probably the most popular measure of risk-adjusted returns for mutual funds and other types of investments; in the case of trading rules, the Sharpe ratio of rule R on time series X is

$$\text{SR}(R; X) = \frac{r(R; X) - r_f}{\sigma(R; X)}, \quad (5)$$

where we are making the assumption that the risk-free rate r_f is constant during the timespan covered by X .

A critique to this measure is that it treats positive excess returns, i.e., windfall profits, exactly the same way as it treats negative excess return; in fact, traders, just like investors, do not regard windfall profits as something to avoid as unexpected losses. A variation of the Sharpe ratio which acknowledges this fact is Sortino ratio [27], which in our case is

$$\text{SR}_d(R; X) = \frac{r(R; X) - r_f}{\text{DSR}_{r_f}(R; X)}. \quad (6)$$

Unlike the Sharpe ratio, the Sortino ratio adjusts the expected return for the risk of falling short of the risk-free return; positive deviations from the mean are not taken into account to calculate risk.

3 The Trading Problem

Static ADTPs can be classified according to the type of positions allowed, entry strategy, profit-taking strategy, and stop-loss or exit strategy. By combining these options (and perhaps more), one can name all different types of trading problems. In the rest of this chapter, we will focus on a particular class of static ADTP, namely the class of problems whereby the trading strategy allows taking both long and short positions at market during the opening auction, a position is closed as soon as a pre-defined profit has been reached, or otherwise at market during the closing auction as a means of preventing losses beyond the daily volatility of an instrument.

Such problems make up the simplest class of problems when it comes to rule evaluation: all is required is open, high, low, and close quotes for each day, since a position is opened at market open, if the rule so commands, and closed either with a fixed profit or at market close.

3.1 The BOFC Problem

Static ADTPs can be classified according to

- the type of positions allowed;
- the entry strategy;
- the profit-taking strategy;
- the stop-loss or exit strategy.

This chapter focuses on a particular ADTP which can be described as follows: take a long or short position on open, take profit if a fixed return is achieved, otherwise close the position at the end of the day.

We will refer to this problem as the BOFC problem, because:

- both (B) long and short positions are allowed;
- the entry is on open (O) at market price;
- the profit-taking strategy is on a fixed (F) return;
- the exit strategy is on close (C) at market price.

3.2 Trading Rules for Static ADTPs

In its most general form, a trading rule for a static ADTP should specify:

1. a sign (long or short) for the position;
2. a limit price for entry, including market price;
3. a take-profit level;
4. a stop-loss level.

All prices should be expressed relative to some other price: the limit price for entry might be expressed relative to the previous close, the take-profit and stop-loss levels might be expressed relative to the price at which the position is opened.

Formally, we can think of a trading rule R as a function of a day in a time series whose value is a 4-tuple

$$R(X, t) = \langle s, r_L, r_{TP}, r_{SL} \rangle, \quad (7)$$

where

1. $s \in \{-1, 1\}$ is the sign of the position to be opened: -1 for a short position, 1 for a long position;
2. $r_L \in \mathbb{R}$ is used to calculate the limit price for entry according to the formula $p_L = x_{t-1}^C e^{r_L}$;
3. $r_{TP} > 0$ is the log-return target from which a take-profit limit of $x e^{r_{TP}}$ is calculated, where x is the price at which the position is opened;
4. $r_{SL} > 0$ is the stop-loss log-return; the stop-loss level is $x e^{-r_{SL}}$.

Although this general framework is capable of accommodating all static ADTPs, for the BOFC problem, $s \in \{-1, 1\}$, $r_L \in \{-\infty, +\infty\}$ (i.e., a binary entry condition: $+\infty = \text{enter}$, $-\infty = \text{do not enter}$), $r_{TP} = \text{constant}$, given by the user, $r_{SL} = +\infty$, meaning a stop-loss price level of 0, which is never triggered. Given these constraints, the result of a trading rule is just a ternary decision: *go short* ($s = -1$, $r_L = -\infty$), *do nothing* ($s \cdot r_L = -\infty$), or *go long* ($s = +1$, $r_L = +\infty$).

3.3 Rule Evaluation Requirements of the BOFC Problem

While for the purpose of designing a profitable trading rule R , i.e., solving any static ADTP, the more information is available the better, whatever the

trading problem addressed, for the purpose of evaluating a given trading rule the quantity and granularity of quote information required varies depending on the problem.

In order to evaluate the performance of rules for the BOFC problem, the open, high, low, and close quotes for each day are needed. The required time series is $X = \{x_t^O, x_t^H, x_t^L, x_t^C\}_{t=1, \dots, N}$.

The log-return generated by rule R in the t^{th} day of time series X is

$$r(R; X, t) = \begin{cases} r_{TP} & \text{if } s \cdot r_L = +\infty \text{ and } \bar{r} > r_{TP}, \\ s \ln \frac{x_t^C}{x_t^O} & \text{if } s \cdot r_L = +\infty \text{ and } \bar{r} \leq r_{TP}, \\ 0 & \text{otherwise,} \end{cases} \quad (8)$$

where

$$\bar{r} = \begin{cases} \ln \frac{x_t^H}{x_t^O} & \text{if } s > 0, \\ \ln \frac{x_t^L}{x_t^O} & \text{if } s < 0. \end{cases} \quad (9)$$

The BOFC problem is therefore among the most complex single-position day-trading problems whose solutions one can evaluate when disposing only of open, high, low, and close quotes for each day. The reason we chose to focus on such problem is indeed that while such kind of quotes are freely available on the Internet for a wide variety of securities and indices, more detailed data can in general only be obtained for a fee.

We approach this problem by evolving trading rules that incorporate fuzzy logic. The adoption of fuzzy logic is useful in two respects: first of all, by recognizing that concept definitions may not always be crisp, it allows the rules to have what is called an *interpolative behavior*, i.e., gradual transitions between decisions and their conditions; secondly, fuzzy logic provides for linguistic variables and values, which make rules more natural to understand for an expert. The next section introduces fuzzy logic, with a particular emphasis on the concepts that are relevant to our approach.

4 Fuzzy Logic

Fuzzy logic was initiated by Lotfi Zadeh with his seminal work on fuzzy sets [35]. Fuzzy set theory provides a mathematical framework for representing and treating vagueness, imprecision, lack of information, and partial truth.

Very often, we lack complete information in solving real world problems. This can be due to several causes. First of all, human expertise is of a qualitative type, hard to translate into exact numbers and formulas. Our understanding of any process is largely based on imprecise, “approximate” reasoning. However, imprecision does not prevent us from performing successfully very hard tasks, such as driving cars, improvising on a chord progression, or trading financial instruments. Furthermore, the main vehicle of human expertise is natural language, which is in its own right ambiguous and vague, while at the same time being the most powerful communication tool ever invented.

4.1 Fuzzy Sets

Fuzzy sets are a generalization of classical sets obtained by replacing the characteristic function of a set A , χ_A which takes up values in $\{0, 1\}$ ($\chi_A(x) = 1$ iff $x \in A$, $\chi_A(x) = 0$ otherwise) with a *membership function* μ_A , which can take up any value in $[0, 1]$. The value $\mu_A(x)$ is the membership degree of element x in A , i.e., the degree to which x belongs in A .

A fuzzy set is completely defined by its membership function. Therefore, it is useful to define a few terms describing various features of this function, summarized in Figure 1. Given a fuzzy set A , its *core* is the (conventional) set of all elements x such that $\mu_A(x) = 1$; its *support* is the set of all x such that $\mu_A(x) > 0$. A fuzzy set is *normal* if its core is nonempty. The set of all elements x of A such that $\mu_A(x) \geq \alpha$, for a given $\alpha \in (0, 1]$, is called the α -cut of A , denoted A_α .

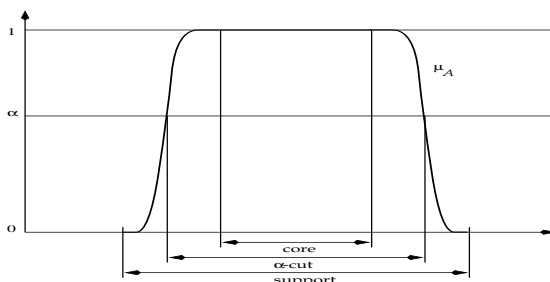


Fig. 1. Core, support, and α -cuts of a set A of the real line, having membership function μ_A .

If a fuzzy set is completely defined by its membership function, the question arises of how the shape of this function is determined. From an engineering point of view, the definition of the ranges, quantities, and entities relevant to a system is a crucial design step. In fuzzy systems all entities that come into play are defined in terms of fuzzy sets, that is, of their membership functions. The determination of membership functions is then correctly viewed as a problem of design. As such, it can be left to the sensibility of a human expert or more objective techniques can be employed. Alternatively, optimal membership function assignment, of course relative to a number of design goals that have to be clearly stated, such as robustness, system performance, etc., can be estimated by means of a machine learning or optimization method. In particular, evolutionary algorithms have been employed with success to this aim. This is the approach we follow in this chapter.

4.2 Operations on Fuzzy Sets

The usual set-theoretic operations of union, intersection, and complement can be defined as a generalization of their counterparts on classical sets by introducing two families of operators, called triangular norms and triangular co-norms. In practice, it is usual to employ the min norm for intersection and the max co-norm for union. Given two fuzzy sets A and B , and an element x ,

$$\mu_{A \cup B}(x) = \max\{\mu_A(x), \mu_B(x)\}; \quad (10)$$

$$\mu_{A \cap B}(x) = \min\{\mu_A(x), \mu_B(x)\}; \quad (11)$$

$$\mu_{\bar{A}}(x) = 1 - \mu_A(x). \quad (12)$$

4.3 Fuzzy Propositions and Predicates

In classical logic, a given proposition can fall in either of two sets: the set of all true propositions and the set of all false propositions, which is the complement of the former. In fuzzy logic, the set of true proposition and its complement, the set of false propositions, are fuzzy. The degree to which a given proposition P belongs to the set of true propositions is its degree of truth, $T(P)$.

The logical connectives of negation, disjunction, and conjunction can be defined for fuzzy logic based on its set-theoretic foundation, as follows:

$$\text{Negation } T(\neg P) = 1 - T(P); \quad (13)$$

$$\text{Disjunction } T(P \vee Q) = \max\{T(P), T(Q)\}; \quad (14)$$

$$\text{Conjunction } T(P \wedge Q) = \min\{T(P), T(Q)\}. \quad (15)$$

Much in the same way, a one-to-one mapping can be established as well between fuzzy sets and fuzzy predicates. In classical logic, a predicate of an element of the universe of discourse defines the set of elements for which that predicate is true and its complement, the set of elements for which that predicate is not true. Once again, in fuzzy logic, these sets are fuzzy and the degree of truth of a predicate of an element is given by the degree to which that element is in the set associated with that predicate.

4.4 Fuzzy Rule-Based Systems

A prominent role in the application of fuzzy logic to real-world problems is played by fuzzy rule-based systems. Fuzzy rule-based systems are systems of fuzzy rules that embody expert knowledge about a problem, and can be used to solve it by performing fuzzy inferences. The ingredients of a fuzzy rule-based systems are *linguistic variables*, *fuzzy rules*, and *defuzzification* methods.

The Mamdani Model

The type of fuzzy rule-based system just described, making use of the min and max as the triangular norm and co-norm, is called the Mamdani model. A Mamdani system [16] has rules of the form

$$\text{IF } x_1 \text{ is } A_1 \text{ AND } \dots \text{ AND } x_n \text{ is } A_n \text{ THEN } y \text{ is } B, \quad (19)$$

where the A_i s and B are linguistic values (i.e., fuzzy sets) and each clause of the form “ x is A ” has the meaning that the value of variable x is in fuzzy set A .

Defuzzification Methods

There may be situations in which the output of a fuzzy inference needs to be a crisp number y^* instead of a fuzzy set R . Defuzzification is the conversion of a fuzzy quantity into a precise quantity.

At least seven methods in the literature are popular for defuzzifying fuzzy outputs [14], which are appropriate for different application contexts. The *centroid method* is the most prominent and physically appealing of all the defuzzification methods. It results in a crisp value

$$y^* = \frac{\int y\mu_R(y)dy}{\int \mu_R(y)dy}, \quad (20)$$

where the integration can be replaced by summation in discrete cases.

The next section introduces evolutionary algorithms, a biologically inspired technique which we use to learn and optimize fuzzy rule bases.

5 Distributed Evolutionary Algorithms

Evolutionary algorithms (EAs) are a broad class of stochastic optimization algorithms, inspired by biology and in particular by those biological processes that allow populations of organisms to adapt to their surrounding environment: genetic inheritance and survival of the fittest.

An evolutionary algorithm (EA) maintains a population of candidate solutions for the problem at hand, and makes it evolve by iteratively applying a (usually quite small) set of stochastic operators, known as *mutation*, *recombination*, and *selection*.

Mutation randomly perturbs a candidate solution; recombination decomposes two distinct solutions and then randomly mixes their parts to form novel solutions; and selection replicates the most successful solutions found in a population at a rate proportional to their relative quality.

The initial population may be either a random sample of the solution space or may be seeded with solutions found by simple local search procedures, if these are available.

The resulting process tends to find, given enough time, globally optimal solutions to the problem much in the same way as in nature populations of organisms tend to adapt to their surrounding environment.

Books of reference and synthesis in the field of EC are [10, 5, 4]; recent advances are surveyed in [33].

Evolutionary algorithms have enjoyed an increasing popularity as reliable stochastic optimization, search and rule-discovering methods in the last few years. The original formulation by Holland and others in the seventies was a sequential one. That approach made it easier to reason about mathematical properties of the algorithms and was justified at the time by the lack of adequate software and hardware. However, it is clear that EAs offer many natural opportunities for parallel implementation [19]. There are several possible parallel EA models, the most popular being the fine-grained or *grid* [17], the coarse-grain or *island* [31], and the master-slave or *fitness parallelization* [9] models. In the grid model, large populations of individuals are spatially distributed on a low-dimensional grid and individuals interact locally within a small neighborhood. In the master-slave model, a sequential EA is executed on what is called the *master* computer. The master is connected to several *slave* computers to which it sends individuals when they require evaluation. The slaves evaluate the individuals (fitness evaluation makes up most of the computing time of an EA) and send the result back to the master.

In the island model, the population is divided into smaller subpopulations which evolve independently and simultaneously according to a sequential EA. Periodic migrations of some selected individuals between islands allow to inject new diversity into converging subpopulations. Microprocessor-based multicomputers and workstation clusters are well suited for the implementation of this kind of parallel EA. Being coarse-grained, the island model is less demanding in terms of communication speed and bandwidth, which makes it a good candidate for a cluster implementation.

6 The Approach

Data mining is a process aimed at discovering meaningful correlations, patterns, and trends between large amounts of data collected in a dataset. A model is determined by observing past behavior of a financial instrument and extracting the relevant variables and correlations between the data and the dependent variable. We describe below a data-mining approach based on the use of evolutionary algorithms, which recognize patterns within a dataset, by learning models represented by sets of fuzzy rules.

6.1 Fuzzy Models

A model is described through a set of fuzzy rules. A rule is made by one or more antecedent clauses (“IF ...”) and a consequent clause (“THEN ...”).

Clauses are represented by a pair of indices referring respectively to a variable and to one of its fuzzy sub-domains, i.e., a membership function.

Using fuzzy rules makes it possible to get homogenous predictions for different clusters without imposing a traditional partition based on crisp thresholds, that often do not fit the data, particularly in financial applications. Fuzzy decision rules are useful in approximating non-linear functions because they have a good interpolative power and are intuitive and easily intelligible at the same time. Their characteristics allow the model to give an effective representation of the reality and simultaneously avoid the “black-box” effect of, e.g., neural networks.

The output of the evolutionary algorithm is a set of rules written in plain consequential sentences. The intelligibility of the model and the high explanatory power of the obtained rules are useful for a trader, because the rules are easy to interpret and explain. An easy understanding of a forecasting method is a fundamental characteristic, since otherwise a trader would be reluctant to use forecasts.

6.2 The Evolutionary Algorithm

The described approach incorporates an EA for the design and optimization of fuzzy rule-based systems that was originally developed to automatically learn fuzzy controllers [29, 24], then was adapted for data mining, [6] and is at the basis of MOLE, a general-purpose distributed engine for modeling and data mining based on EAs and fuzzy logic [30].

A MOLE classifier is a rule base, whose rules comprise up to four antecedent and one consequent clause each. Input and output variables are partitioned into up to 16 distinct linguistic values each, described by as many membership functions. Membership functions for input variables are trapezoidal, while membership functions for the output variable are triangular.

Classifiers are encoded in three main blocks:

1. a set of trapezoidal membership functions for each input variable; a trapezoid is represented by four fixed-point numbers, each fitting into a byte;
2. a set of symmetric triangular membership functions, represented as an area-center pair, for the output variable;
3. a set of rules, where a rule is represented as a list of up to four antecedent clauses (the IF part) and one consequent clause (the THEN part); a clause is represented by a pair of indices, referring respectively to a variable and to one of its membership functions.

An island-based distributed EA is used to evolve classifiers. The sequential algorithm executed on every island is a standard generational replacement, elitist EA. Crossover and mutation are never applied to the best individual in the population.

The recombination operator is designed to preserve the syntactic legality of classifiers. A new classifier is obtained by combining the pieces of two parent

classifiers. Each rule of the offspring classifier can be inherited from one of the parent programs with probability $1/2$. When inherited, a rule takes with it to the offspring classifier all the referred domains with their membership functions. Other domains can be inherited from the parents, even if they are not used in the rule set of the child classifier, to increase the size of the offspring so that their size is roughly the average of its parents' sizes.

Like recombination, mutation produces only legal models, by applying small changes to the various syntactic parts of a fuzzy rulebase.

Migration is responsible for the diffusion of genetic material between populations residing on different islands. At each generation, with a small probability (the migration rate), a copy of the best individual of an island is sent to all connected islands and as many of the worst individuals as the number of connected islands are replaced with an equal number of immigrants.

A detailed description of the evolutionary algorithm and of its genetic operators can be found in [24].

6.3 The Data

In principle, the modeling problem we want to solve requires finding a function which, for a given day t , takes the past history of time series X up to t and produces a trading signal *go short*, *do nothing*, or *go long*, for the next day.

One could therefore try to approach this problem directly, by evolving trading rules in the most general form by means of genetic programming [21, 18, 23, 34]. The search space for such trading rules is, clearly, incredibly huge. However, there exists an impressive body of expertise used everyday by practitioners in the financial markets about summarizing all important information of the past history of a financial time series into few relevant statistics. That body of expertise is called *technical analysis*.

Technical analysis is the study of past financial market data, primarily through the use of charts, to forecast price trends and make investment decisions. In its purest form, technical analysis considers only the actual price behavior of the market or instrument, based on the premise that price reflects all relevant factors before an investor becomes aware of them through other channels [32].

The idea is then to reduce the dimensionality of the search space by limiting the inputs of the models we look for to a collection of the most popular and time-honored technical analysis statistics and indicators.

A technical indicator is a short-term statistics of a time series used by technical analysts to predict future price movements of financial instruments. Price data include any combination of the open, high, low, and close, plus volume information, over a (generally short) period of time. As we will see in the next sections, some indicators use only the closing prices, while others incorporate other information like volume and opening prices into their formulas. Some others consist of a combination of two or more different indicators.

The advantage of using different technical indicators, is that we can dispose of different ways to analyze price movements. Some indicators, like moving averages, are defined by using simple formulas and their mechanics are thus relatively easy to understand. Others, like stochastic oscillators, have more complex definitions. Regardless of the complexity of their definition, technical indicators can provide valid elements for predicting the direction of market price movements.

Popular Statistics and Technical Indicators

In this section, we provide definitions and a basic discussion of several popular technical indicators that have been adopted as inputs to the data mining process and to the models such process looks for.

A *moving average* is the average of the closing values of a financial instrument over a given time period. Generally speaking, moving averages tend to smooth out the short-term oscillations of a time series and identify longer-term trends. The two most popular types of moving average indicators are the *simple* moving average (SMA) and the *exponential* moving average (EMA). The difference between these indicators is uniquely on the weight each of them carry to recent prices compared to old prices. SMA considers equally important all the prices in a considered period, whereas EMA considers that recent prices are more important than the old ones. In general, a buy signal is generated when the instrument's price rises above a moving average and a sell signal is generated when the instrument's price falls below a moving average.

Simple and exponential moving averages are used in our approach by combining them with the closing price value at current day t . For both forms, we consider the moving average of closing values over the n days before t , with $n \in \{5, 10, 20, 50, 100, 200\}$ noted $SMA_n(t)$ and $EMA_n(t)$ respectively, and we then consider the position of the closing price x_t^C with respect to the simple (exponential) moving average, noted $x_t^C : SMA_n(t)$ ($x_t^C : EMA_n(t)$), as follows:

$$x_t^C : SMA_n(t) = \ln \frac{x_t^C}{SMA_n(t)} \quad (21)$$

and

$$x_t^C : EMA_n(t) = \ln \frac{x_t^C}{EMA_n(t)}. \quad (22)$$

If the above logarithms are negative, i.e., the closing price at t is lower than the moving average/exponential moving average value for the previous n days, this means that the price tends to decrease. Otherwise, it means the price tends to increase. In order to get further information about the market trend, we analyze also how the above result evolves from day $t - 1$ to t . The formulas for these differences are:

$$\Delta(x_t^C : SMA_n(t)) = x_t^C : SMA_n(t) - x_{t-1}^C : SMA_n(t-1). \quad (23)$$

$$\Delta(x_t^C : EMA_n(t)) = x_t^C : EMA_n(t) - x_{t-1}^C : EMA_n(t-1). \quad (24)$$

The *Moving Average Convergence/Divergence* (MACD) is a technical indicator which shows the difference between a fast and a slow EMA of closing prices. The simplest version of this indicator is composed of two lines: the MACD line, which is the difference between the two EMAs, and a *signal line*, which is an EMA of the MACD line itself. We use the standard periods recommended back in the 1960s by Gerald Appel, which are 12 and 26 days [20]:

$$\text{MACD}(t) = \text{EMA}_{26}(t) - \text{EMA}_{12}(t) \quad (25)$$

The signal line we have used is also standard. It consists of an EMA_9 of the MACD line. This allows to further smooth the trend of market price values:

$$\text{signal}(t) = \text{EMA}_9(t; \text{MACD}) \quad (26)$$

We also consider Thomas Aspray's difference between the MACD and the signal line, which is often plotted as a solid block *histogram* style:

$$\text{histogram}(t) = \text{MACD}(t) - \text{signal}(t). \quad (27)$$

Its change from the previous day is given by:

$$\Delta(\text{MACD}(t) - \text{signal}(t)) = \text{histogram}(t) - \text{histogram}(t - 1). \quad (28)$$

The *rate of change* (ROC) is a simple technical indicator showing the difference between the closing price on a current day t and the corresponding price n days before day t . Because this indicator is measured in terms of the old closing price, it represents the increase as a fraction,

$$\text{ROC}_n(t) = \frac{x_t^C - x_{t-n}^C}{x_{t-n}^C}, \quad (29)$$

which is positive when the price value is increasing over the last n days, negative otherwise. We use the popular version with $n = 12$.

Developed by George C. Lane in the late 1950s, the *Stochastic Oscillator* is a momentum indicator that shows the location of the current close relative to the high/low range over a set number of periods. Closing levels that are consistently near the top of the range indicate accumulation (buying pressure) and those near the bottom of the range indicate distribution (selling pressure). The fast stochastic oscillator, $\%K_n$ calculates the ratio of two closing price statistics — the difference between the latest closing price and the lowest closing price in the last n days over the difference between the highest and lowest closing prices in the last n days:

$$\%K_n(t) = \frac{x_t^C - \min_{i=t-n}^t \{x_i^C\}}{\max_{i=t-n}^t \{x_i^C\} - \min_{i=t-n}^t \{x_i^C\}} \cdot 100. \quad (30)$$

We use the popular time period n of 14 days. $\%K_n = 0$ when the current closing price is a low for the last n days; $\%K_n = 100$ when the current closing price is a high for the last n days.

The slow stochastic oscillator $\%D_n$ calculates the simple moving average of $\%K_n$ across a period of k days:

$$\%D_n(t) = \text{SMA}_k(t; \%K_n). \quad (31)$$

We use the popular value of $k = 3$ days.

The *relative strength index* (RSI) is another oscillator which determines the strength of a financial instrument by comparing upward and downward close-to-close movements in a period of n days. For each day t , the amounts of upward change, $U(t) > 0$, or downward change, $D(t) > 0$, are calculated as follows. On an “up” day t ,

$$\begin{aligned} U(t) &= x_t^C - x_{t-1}^C, \\ D(t) &= 0. \end{aligned}$$

Conversely, on a “down” day t ,

$$\begin{aligned} U(t) &= 0, \\ D(t) &= x_t^C - x_{t-1}^C. \end{aligned}$$

If the closing price in $t - 1$ is the same as the closing price in t , both $U(t)$ and $D(t)$ are zero. An exponential moving average of up days in the period of n days, U , is calculated. Similarly, an exponential moving average of down days in the period of n days, D , is calculated. The ratio of those averages is the relative strength,

$$\text{RS}(t) = \frac{\text{EMA}_n(t; U)}{\text{EMA}_n(t; D)} \quad (32)$$

This is converted to a relative strength index between 0 and 100:

$$\text{RSI}(t) = 100 - 100 \cdot \frac{1}{1 + \text{RS}(t)}. \quad (33)$$

This can be rewritten as follows to emphasize the way RSI expresses the up as a proportion of the total up and down (averages in each case),

$$\text{RSI}(t) = 100 \cdot \frac{\text{EMA}_n(t; U)}{\text{EMA}_n(t; U) + \text{EMA}_n(t; D)}. \quad (34)$$

The basic idea behind the use of RSI is that when there is a high proportion of daily movements in one direction, this suggests an extreme, and prices are likely to “change direction”. More precisely, a financial instrument is considered overbought if $\text{RSI} \geq 70$, meaning that a speculator should consider the option of selling. Conversely, a speculator should consider buying if $\text{RSI} \leq 30$ (oversold).

The *money flow index* indicator (MFI) indicates the balance between money flowing into and out of a financial instrument. The construction and

interpretation of this indicator is similar to RSI, with the only difference that volume is also taken into account.

The *typical price* \bar{x}_t is the average of high, low and close values in day t :

$$\bar{x}_t = \frac{x_t^H + x_t^L + x_t^C}{3}. \quad (35)$$

Money flow is the product of typical price on a day t and the volume on that day:

$$\text{MF}(t) = \bar{x}_t \cdot x_t^V. \quad (36)$$

Totals of the money flow amounts over the given n days are then formed. *Positive* money flow is the total for those days where the typical price is higher than the previous day's typical price, and *negative* money flow is the total of those days where the typical price is below the previous day's typical price. A money ratio is then formed:

$$\text{MR}(t) = \frac{\text{MF}^+(t)}{\text{MF}^-(t)}, \quad (37)$$

from which a *money flow index* ranging from 0 to 100 is formed,

$$\text{MFI}(t) = 100 - \frac{100}{1 + \text{MR}(t)}. \quad (38)$$

When analyzing the money flow index, one needs to take into consideration the following points:

- divergences between the indicator and price movement — if prices grow while MFI falls (or *vice versa*), there is a great probability of a price turn;
- MFI over 80 or under 20 signals a potential peak or bottom of the market.

The *accumulation/distribution* indicator (AccDist) is a cumulative total volume technical analysis indicator created by Marc Chaikin, which adds or subtracts each day's volume in proportion to where the close is between the day's high and low. For example, many up days occurring with high volume in a down trend could signal that the demand for the financial instrument is starting to increase. In practice this indicator is used to find situations where the indicator is heading in the opposite directions than the price. Once this divergence has been identified, traders will wait to confirm the reversal and make their transaction decisions using other technical indicators.

A *close location value* is defined as:

$$\text{CLV}(t) = \frac{(x_t^C - x_t^L) - (x_t^H - x_t^C)}{x_t^H - x_t^L}. \quad (39)$$

This ranges from -1 , when the close is the low of the day, to $+1$ when it is the high. For instance if the close is $3/4$ the way up the range, then CLV is

+0.5. The accumulation/distribution index adds up volume multiplied by the CLV factor, ie.

$$\text{AccDist}(t) = \text{AccDist}(t-1) + x_t^V \cdot \text{CLV}(t). \quad (40)$$

Finally, the *on-balance volume* indicator (OBV) is intended to relate price and volume in the stock market. OBV is based on a cumulative total volume. Volume on an up day (i.e., close value at day t higher than close value at day $t-1$) is added and volume on a down day is subtracted. The formula is

$$\text{OBV}(t) = \text{OBV}(t-1) + \begin{cases} x_t^V & \text{if } x_t^C > x_{t-1}^C, \\ 0 & \text{if } x_t^C = x_{t-1}^C, \\ -x_t^V & \text{if } x_t^C < x_{t-1}^C. \end{cases} \quad (41)$$

The starting point for an OBV total is arbitrary. Only the shape of the resulting indicator is used, not the actual level of the total.

Combining Statistics and Technical Indicators

After a careful scrutiny of the most popular technical indicators outlined in the previous subsection, we concluded that more data were needed if we wanted an evolutionary algorithm to discover meaningful models expressed in the form of fuzzy IF-THEN rules. Combinations of statistics and technical indicators are required that mimic the reasonings analysts and traders carry out when they are looking at a technical chart, comparing indicators with current price, checking for crossings of different graphs, and so on.

Combinations may take the form of differences between indicators that are pure numbers or that have a fixed range, or of ratios of indicators such as prices and moving average, that are expressed in the unit of measure of a currency. Following the use of economists, we consider the natural logarithm of such ratios, and we define the following notation: given two prices x and y , we define

$$x : y \equiv \ln \frac{x}{y}. \quad (42)$$

Eventually, we came up with the following combinations:

- all possible combinations of the Open (O), High (H), Low (L), Close (C), and previous-day Close (P) prices: $O : P$, $H : P$, $L : P$, $C : P$, $H : O$, $C : O$, $O : L$, $H : L$, $H : C$, $C : L$;
- close price compared to simple and exponential moving averages, $C : \text{SMA}_n$, $C : \text{EMA}_n$, $n \in \{5, 10, 20, 50, 100, 200\}$;
- the daily changes of the close price compared to simple and exponential moving averages, $\Delta(C : \text{SMA}_n)$, $\Delta(C : \text{EMA}_n)$, where $\Delta(x) \equiv x(t) - x(t-1)$;
- the MACD histogram, i.e., MACD – signal, and the daily change thereof, $\Delta(\text{Histogram})$;

- Fast stochastic oscillator minus slow stochastic oscillator, $\%K - \%D$, and the daily change thereof, $\Delta(\%K - \%D)$.

The full list of the statistics, technical indicators, and their combinations used as model inputs is given in Table 1.

6.4 Fitness

Modeling can be thought of as an optimization problem, where we wish to find the model M^* which maximizes some criterion which measures its accuracy in predicting $y_i = x_{im}$ for all records $i = 1, \dots, N$ in the training dataset. The most natural criteria for measuring model accuracy are the mean absolute error and the mean square error.

One big problem with using such criteria is that the dataset must be *balanced*, i.e., an equal number of representative for each possible value of the predictive attribute y_i must be present, otherwise the underrepresented classes will end up being modeled with lesser accuracy. In other words, the optimal model would be very good at predicting representatives of highly represented classes, and quite poor at predicting individuals from other classes.

To solve this problem, MOLE divides the range $[y_{\min}, y_{\max}]$ of the predictive variable into 256 bins. The b^{th} bin, X_b , contains all the indices i such that

$$1 + \lfloor 255 \frac{y_i - y_{\min}}{y_{\max} - y_{\min}} \rfloor = b. \quad (43)$$

For each bin $b = 1, \dots, 256$, it computes the mean absolute error for that bin

$$\text{err}_b(M) = \frac{1}{\|X_b\|} \sum_{i \in X_b} |y_i - M(x_{i1}, \dots, x_{i,m-1})|, \quad (44)$$

then the total absolute error (TAE) as an integral of the histogram of the absolute errors for all the bins, $\text{tae}(M) = \sum_{b: \|X_b\| \neq 0} \text{err}_b(M)$. Now, the mean absolute error for every bin in the above summation counts just the same no matter how many records in the dataset belong to that bin. In other words, the level of representation of each bin (which, roughly speaking, corresponds to a class) has been factored out by the calculation of $\text{err}_b(M)$. What we want from a model is that it is accurate in predicting all classes, independently of their cardinality.

The fitness used by the EA is given by $f(M) = \frac{1}{\text{tae}(M)+1}$, in such a way that a greater fitness corresponds to a more accurate model.

7 Experiments

The approach described above has been applied to trading several financial instruments. This section reports some of the experiments that have been carried out and their results.

Table 1. The independent variables of the dataset.

Name	Formula	Explanation
Open	x_t^O	the opening price on day t
High	x_t^H	the highest price on day t
Low	x_t^L	the lowest price on day t
Close	x_t^C	the closing price on day t
Volume	x_t^V	the volume traded on day t
O:P	$x_t^O : x_{t-1}^C$	opening price on day t vs. previous-day closing price
H:P	$x_t^H : x_{t-1}^C$	high on day t vs. previous-day closing price
L:P	$x_t^L : x_{t-1}^C$	low on day t vs. previous-day closing price
C:P	$x_t^C : x_{t-1}^C$	close on day t vs. previous-day closing price
H:O	$x_t^H : x_t^O$	high on day t vs. same-day opening price
C:O	$x_t^C : x_t^O$	closing on day t vs. same-day opening price
O:L	$x_t^O : x_t^L$	opening price on day t vs. same-day lowest price
H:L	$x_t^H : x_t^L$	high on day t vs. same-day low
H:C	$x_t^H : x_t^C$	high on day t vs. same-day closing price
C:L	$x_t^C : x_t^L$	closing price on day t vs. same-day low
dVolume	$x_t^V : x_{t-1}^V$	change in volume traded on day t
C:MAN	$x_t^C : \text{SMA}_n(t)$	n -day simple moving averages, for $n \in \{5, 10, 20, 50, 100, 200\}$.
dC:MAN	$\Delta(x_t^C : \text{SMA}_n(t))$	daily change of the above
C:EMAN	$x_t^C : \text{EMA}_n(t)$	n -day exponential moving averages, for $n \in \{5, 10, 20, 50, 100, 200\}$.
dC:EMAN	$\Delta(x_t^C : \text{EMA}_n(t))$	daily change of the above
MACD	$\text{MACD}(t)$	Moving average convergence/divergence on day t
Signal	$\text{signal}(t)$	MACD signal line on day t
Histogram	$\text{MACD}(t) - \text{signal}(t)$	MACD histogram on day t
dHistogram	$\Delta(\text{MACD}(t) - \text{signal}(t))$	daily change of the above
ROC	$\text{ROC}_{12}(t)$	rate of change on day t
K	$\%K_{14}(t)$	fast stochastic oscillator on day t
D	$\%D_{14}(t)$	slow stochastic oscillator on day t
K:D	$\%K_{14}(t) - \%D_{14}(t)$	fast vs. slow stochastic oscillator
dK:D	$\Delta(\%K_{14}(t) - \%D_{14}(t))$	daily change of the above
RSI	$\text{RSI}_{14}(t)$	relative strength index on day t
MFI	$\text{MFI}_{14}(t)$	money-flow index on day t
AccDist	$\Delta(\text{AccDist}(t))$	The change of the accumulation/distribution index on day t
OBV	$\Delta(\text{OBV}(t))$	The change of on-balance volume on day t
PrevClose	x_{t-1}^C	closing price on day $t - 1$

7.1 Aims of the Experimental Study

There would be many interesting questions to ask of a data-mining approach applied to single-position day trading like the one described above. Here, we address the following:

1. What are the generalization capabilities of the models obtained? This question has to do with being able to correctly model the behavior of the financial instrument used for learning for a timespan into the future: here the basic question to ask is, how well can we expect the model perform tomorrow with today's data, if we have used data up to yesterday to learn it? However, another relevant question is, how does the model's performance decay as we move into the future, i.e., the day after tomorrow, and then in three days, and so forth?
2. How much historical data is needed to obtain a reliable model? This is a critical issue, as for many financial instruments the history available may be limited: think for example of the stock of a new company resulting from the merger of two pre-existing companies, or of the stock of an old company which starts being traded as a result of an initial public offering. Furthermore, it could be argued that markets evolve as the context in which they function changes; therefore, very old data about a given financial instrument might not be of any help for learning a model that reflects current (and possibly future) behavior of that instrument. The issue has thus two faces, which are distinct but related:
 - a) ideally, for a data-mining approach, the more data is available the better; however, finance is a domain where only historical data is available; there is no way of obtaining more data by performing experiments: so the question is, how much data is required for learning a meaningful model?
 - b) if too little data exists to allow a meaningful model to be learned, too much data (especially, data that goes too deep into the past) might lead to a model which does not correctly reflect current behavior: how old data can be before it is useless or counterproductive for learning a reliable model?

7.2 Experimental Protocol

The following financial instruments have been used for the experiments:

- the Dow Jones Industrial Average index (DJI);
- the Nikkei 225 index (N225);
- the common stock of Italian oil company ENI, listed since June 18, 2001 on the Milan stock exchange;
- the common stock of world's leading logistics group Deutsche Post World Net (DPW), listed since November 20, 2000 on the XETRA stock exchange;

- the common stock of Intel Co. (INTC), listed on the NASDAQ.

For all the instruments considered, three datasets of different length have been generated, in an attempt to gain some clues on how much historical data is needed to obtain a reliable model:

- a “long-term” dataset, generated from the historical series of prices since January 1, 2002 till December 31, 2006, consisting of 1,064 records, of which 958 are used for training and the most recent 106 are used for testing;
- a “medium-term” dataset, generated from the historical series of prices since January 1, 2004 till December 31, 2006, consisting of 561 records, of which 505 are used for training and the most recent 56 are used for testing;
- a “short-term” dataset, generated from the historical series of prices since January 1, 2005 till December 31, 2006, consisting of 304 records, of which 274 are used for training and the most recent 30 are used for testing;

The validation dataset, in all cases, consists of records corresponding to the first half of 2007, which require a historical series starting from March 17, 2006 (200 market days before January 2, 2007) to be generated, due to the 200-day moving averages and their changes that need to be computed. Having six months of validation data allows us to evaluate the rate of decay of model performance as it “gets stale”.

7.3 Results

For each combination of instrument and dataset, ten runs of the MOLE data mining engine with four islands of size 100 connected according to a ring topology and with a standard parameter setting have been performed. Each run lasted as many generations as required to reach convergence, defined as no improvement for 100 consecutive generations. The results are summarized in Table 2. Figure 2 shows how the automated trading strategy based on the models found by each run fared when applied to the validation set.

To correctly interpret the fitness values appearing in Table 2, the following considerations may be of interest:

- a hypothetical model M_{rnd} that returns a completely random (i.e., uniformly distributed) value for the Action variable in the interval $[-1, 1]$ will have an expected mean absolute error of 1 in the -1 bin, of $\frac{2}{3}$ in the 0 bin, and of 1 in the $+1$ bin, according to Equation 45, and thus an expected total average error of 2 and $\frac{2}{3}$, corresponding to an expected fitness $E[f(M_{\text{rnd}})] = \frac{3}{11} = 0.2727\dots$ — model M_{rnd} is representative of a situation of complete ignorance;
- a “constant” model M_0 that always returns zero for the Action variable, which is equivalent to a *do nothing* trading signal, will have an expected mean absolute error of 1 in the -1 bin, of 0 in the 0 bin, and of 1 in the $+1$ bin, according to Equation 45, and thus an expected total average error of 2, corresponding to an expected fitness $E[f(M_0)] = \frac{1}{3} = 0.3333\dots$;

Table 2. Summary of experimental results. Minimum, average, and maximum values are over the best models produced in ten independent runs of the island-based evolutionary algorithm, when applied to the corresponding validation set. When infinite Sharpe ratios occur, the average is computed over the finite values only.

Performance Measure	Dataset								
	Long-Term			Medium-Term			Short-Term		
	min	avg	max	min	avg	max	min	avg	max
Dow Jones Industrial Average									
Fitness	0.3297	0.3394	0.3484	0.3188	0.3327	0.3457	0.3183	0.3398	0.3671
Return*	0.1618	0.2303	0.4017	0.1062	0.2280	0.5503	0.0996	0.3225	0.5416
Sharpe Ratio	1.2250	2.0073	3.6700	0.6123	2.0901	4.9866	0.6064	3.0687	5.1994
Sortino Ratio	1.5380	2.5572	4.7616	0.7642	2.7949	6.5557	0.7215	4.0799	6.9968
Nikkei 225									
Fitness	0.3211	0.3414	0.3651	0.3241	0.3418	0.3575	0.3205	0.3351	0.3529
Return*	-0.1467	-0.0006	0.2119	-0.1118	0.0006	0.1436	-0.1063	-0.0161	0.1040
Sharpe Ratio	-1.6352	-0.0192	2.8201	-1.2708	-0.0238	1.5947	-1.6385	-0.1828	2.0360
Sortino Ratio	-1.9181	0.0782	4.1485	-1.5070	0.0253	2.1311	-1.9135	-0.1033	3.2197
ENI Stock									
Fitness	0.2459	0.3268	0.3500	0.2475	0.2907	0.3425	0.2402	0.2949	0.3277
Return*	-0.1389	0.0122	0.2120	-0.0856	0.0248	0.1547	-0.1936	-0.0372	0.2643
Sharpe Ratio	-2.0204	-0.2971	2.2852	-2.1096	-0.1935	1.7443	-2.5211	-0.8458	2.4507
Sortino Ratio	-2.3274	-0.2751	3.0867	-2.4578	-0.1799	2.4460	-2.8959	-0.9655	3.2188
Deutsche Post World Net Stock									
Fitness	0.3182	0.3306	0.3451	0.3200	0.3342	0.3506	0.3118	0.3299	0.3403
Return*	-0.0607	0.0476	0.2646	-0.0246	0.0547	0.2480	0.0117	0.1169	0.2820
Sharpe Ratio	$-\infty$	0.1380	3.7798	$-\infty$	0.2476	2.0083	-2.7351	0.5796	3.2487
Sortino Ratio	-15.8114	-2.3809	10.5500	-15.8114	-0.1780	12.7425	-10.2067	0.0920	4.6700
Intel Co. Stock									
Fitness	0.2490	0.3050	0.3443	0.2433	0.2838	0.3658	0.2394	0.2665	0.3333
Return*	0.0247	0.1015	0.1669	0.0131	0.2254	0.4292	-0.0244	0.1252	0.3632
Sharpe Ratio	-0.2128	0.6425	2.0494	-0.3872	2.0846	4.1456	$-\infty$	0.7836	2.7926
Sortino Ratio	-0.2467	0.8624	3.2520	-0.4569	2.9042	6.1129	-15.8114	-0.7107	3.4903

*) Annualized logarithmic return.

- the other two “constant” models M_{-1} and M_{+1} , instead, will have an expected total average error of 3, corresponding to an expected fitness $E[f(M_{-1})] = E[f(M_{+1})] = \frac{1}{4} = 0.25$.

For the calculation of the Sortino ratio, we made the simplifying assumption of a constant risk-free rate of 3.925% (the average three-month Euribor rate in the first half of year 2007²) for the instruments denominated in euros, of 5.175% (the average discount rate of 13-week US Treasury Bills in the first half of 2007³) for those denominated in dollars, and of 0.650% (the yield of 3-month Japanese Government Bills in June 2007⁴) for those denominated in yens.

² Source: Euribor, URL <http://www.euribor.org/>.

³ Source: TreasuryDirect, URL <http://www.treasurydirect.gov>.

⁴ Source: Bloomberg

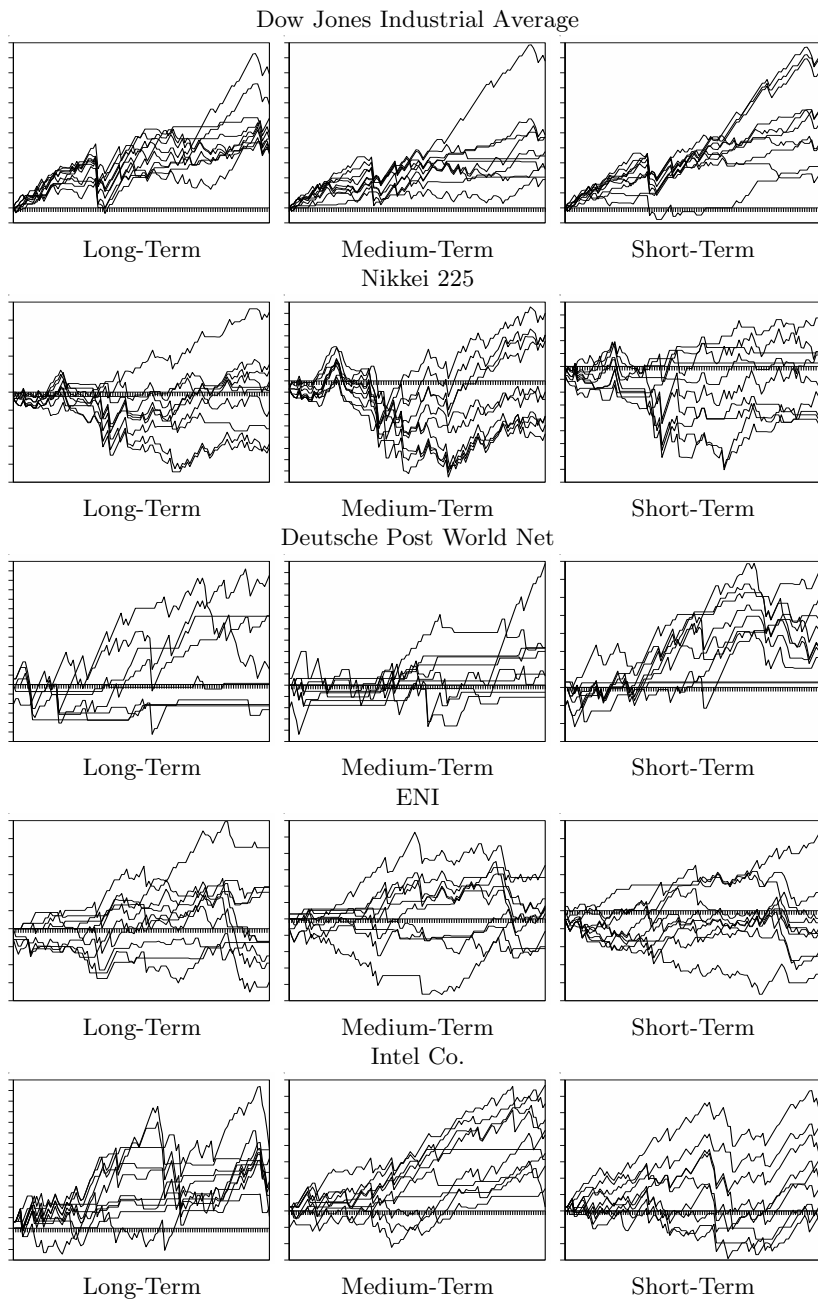


Fig. 2. Graphs of the cumulative returns obtained by the automated trading strategy based on the models generated by ten independent runs, for each instrument and training dataset.

7.4 Discussion of Results

In order to have a better idea of the quality of the results, one can compare them to a buy-and-hold strategy over the same time period, summarized in Table 4.

Table 3. Annualized logarithmic return and Sharpe ratio of a buy-and-hold strategy over the validation period for the instruments considered.

Instrument	Log-Return	Sharpe Ratio
Dow Jones Industrial Average	0.1461	1.4218
Nikkei 225	0.1111	0.7258
ENI	0.1168	0.8074
Deutsche Post World Net	0.1320	0.5509
Intel Co.	0.2365	1.0696

Besides evaluating the results on a quantitative basis, it is also of interest to perform a qualitative assessment of the model obtained, to see if some useful insight is captured by the data mining approach. Figure 3 shows the best model of the Dow Jones Industrial Average obtained from the short-term dataset in Run #1. The model uses in its rules 13 independent variables only, out of the 54 shown in Table 1, namely: Close, O:P, dVolume, C:MA5, C:MA10, dC:MA20, C:MA200, C:EMA10, K, dK:D, MFI, and OBV. Most rules make sense to an expert: for example, the second rule states that a closing price below the 200-day moving average speaks in favor going long, as for a mean-reverting series it can be expected that the price will go back up. Similar *a posteriori* explanations have been given by expert traders for these and other rules on which they were asked for an opinion. Overall, a tendency towards using volume information can be observed, both directly (dVolume) and through indicators, like MFI and OBV, whose definitions depend on volume.

8 An Ensemble Technique for Improving Performance

Ensemble techniques are known to be very useful to boost the performance of several types of machine learning methods and their benefits have recently been proven experimentally [3] on Pittsburgh classifier systems, an approach to which the one proposed in this chapter is very closely related.

Ensemble learning, a family of techniques established for more than a decade in the Machine Learning community, provides performance boost and robustness to the learning process by integrating the collective predictions of a set of models in some principled fashion [2]. In particular, *bagging* [8] (an abbreviation for *bootstrap aggregating*) is an ensemble technique which consists

```

IF MFI is high AND OBV is positive AND dVolume is slightlyNegative AND
dK:D is positive
THEN Action is Short
IF C:MA200 is negative THEN Action is Long
IF MFI is high AND dVolume is slightlyNegative AND C:MA5 is aroundZero
THEN Action is Short
IF MFI is between60and70 AND dC:MA20 is positive THEN Action is Short
IF dK:D is positive AND K is between50and70 AND dC:MA20 is smallPositive
AND O:P is positive
THEN Action is Short
IF C:MA10 is slightlyNegative AND Close is around12000 AND
C:EMA10 is aroundZero
THEN Action is Long
IF dK:D is positive AND K is between50and70 AND dC:MA20 is smallPositive
AND O:P is positive THEN Action is Short
IF MFI is high AND OBV is positive AND dVolume is slightlyNegative
AND dK:D is positive THEN Action is Short
IF MFI is high AND OBV is positive AND dVolume is slightlyNegative
AND dK:D is positive THEN Action is Short

```

Fig. 3. The best model of the Dow Jones Industrial Average obtained from the short-term dataset in Run #1. The definitions of the linguistic values, which are an integral part of the model, are not shown for conciseness; instead, descriptive labels have been assigned by hand to improve clarity.

in training several models on different training sets (called bootstrap samples) sampled with replacement from the original training set, and combining such models by averaging the output or voting.

A simple but effective ensemble technique to greatly reduce the risk of the trading rules, similar to bagging, is a model-averaging approach whereby a number of models, discovered in independent runs of the algorithm, perhaps using different parameter settings but, unlike in bagging, the same training and test sets, are pooled and their output is combined by averaging. Table 5 reports the results obtained by such ensemble technique applied to the models resulting from the ten runs summarized in Table 2 for each combination of instrument and dataset. These results show a dramatic reduction of risk, in the face of a modest reduction of the returns. Overall, the performance measures of the model pool are usually better than their averages over the members of the pool, with notable exceptions on those instruments, like the Dow Jones Industrial Average, where the performance of the individual model was already more than satisfactory on average. In some cases, the benefits of pooling do not manage to turn a negative result into a positive one; however, pooling makes the losses less severe.

Table 4. Results of the ensemble technique applied to the best models produced in ten independent runs of the island-based evolutionary algorithm, when applied to the corresponding validation set.

Performance	Dataset		
Measure	Long-Term	Medium-Term	Short-Term
Dow Jones Industrial Average			
Return*	0.1970	0.1272	0.3958
Sharpe Ratio	1.7094	0.8744	3.8192
Sortino Ratio	2.1619	1.0636	4.9320
Nikkei 225			
Return*	-0.0369	0.0175	0.0164
Sharpe Ratio	-0.4849	0.1238	0.1405
Sortino Ratio	-0.5878	0.1543	0.1767
ENI Stock			
Return*	0.0853	0.0057	-0.0622
Sharpe Ratio	0.9108	-0.7189	-1.2671
Sortino Ratio	1.2734	-1.2080	-1.5334
Deutsche Post World Net Stock			
Return*	-0.0776	-0.0188	0.0781
Sharpe Ratio	-2.2932	-2.4571	0.5558
Sortino Ratio	-2.3166	-2.6190	0.7174
Intel Co. Stock			
Return*	0.0946	0.2317	0.1041
Sharpe Ratio	0.5099	2.9882	0.5273
Sortino Ratio	0.6134	4.3481	0.6269

*) Annualized logarithmic return.

9 Conclusions

This chapter has discussed an approach to a very specific class of day-trading problems, that we have called *single-position*, by means of a fuzzy-evolutionary modeling technique.

A distributed evolutionary algorithm learns models, expressed in the form of sets of fuzzy IF-THEN rules, that relate the optimal trading signal for a given financial instrument on day $t + 1$ to a set of 54 statistics and technical indicators for the same instrument measured on day t , which effectively filter and summarize the previous history of the relevant time series up to day t .

The capability of the approach to discover useful models, which can provide risk-adjusted returns in excess of a buy-and-hold strategy, has been demonstrated through a series of experiments, whose aim was also to assess the generalization capabilities of the models discovered. Finally, an ensemble technique has been suggested to further improve the performance of the trading rules generated by the modeling technique.

The strength of the approach described is its simplicity, which makes it accessible even to unsophisticated traders. Indeed:

- the data considered each day by a trading rule to make a decision about its action is restricted to those freely available on the Internet; visibility of the past time series is filtered through a small number of popular and quite standard technical indicators;
- the underlying trading strategy is among the simplest and most accessible even to the unsophisticated individual trader; its practical implementation does not even require particular kinds of information-technology infrastructures, as it could very well be enacted by placing a couple of orders with a broker on the phone before the market opens; there is no need to monitor the market and to react in a timely manner.

Nonetheless, the results clearly indicate that, despite its simplicity, such an approach may yield, if carried out carefully, significant returns, in the face of a risk that is, to be sure, probably higher than the one the average investor would be eager to take, but all in all proportionate with the returns expected.

To conclude, the issue of transaction costs is worth some discussion. Transaction costs are the sum of two parts: the commissions claimed by the broker or the dealer who execute the orders, and the price slippage that can occur in some circumstances, when the price moves before the order can be executed and the trader has to aggressively move the order price accordingly to get it executed. To begin with, no slippage can occur given the particular structure of the trading strategy (the position is opened with a market order placed before the opening auction, and closed either by the execution of a limit order entered when the position was opened or automatically by the broker at market close). Commissions are the only component that could have an impact on the profitability of the evolved trading rules. However, they can be made as low as desired simply by increasing the amount invested: under most commission structures applied by brokers, commissions are calculated as a percentage (e.g., 0.19%) of the order value with a maximum (e.g., \$19); in other cases they are flat, independent of the order size; many brokers offer substantial discounts on commissions to frequent traders. Therefore, even if transaction costs have not been taken into account in reporting the experimental results, their impact would be negligible for an investor with sufficient capitalization.

References

1. F. Allen and R. Karjalainen. Using genetic algorithms to find technical trading rules. *Journal of Financial Economics*, 51:245–271, 1999.
2. Various Authors. Special issue on integrating multiple learned models. *Machine Learning*, 36(1–2), 2006.
3. Jaume Bacardit and Natalio Krasnogor. Empirical evaluation of ensemble techniques for a pittsburgh learning classifier system. In *Proceedings of the Ninth*

- International Workshop on Learning Classifier Systems*, Seattle, WA, USA, July 8–9 2006. ACM.
4. T. Bäck. *Evolutionary algorithms in theory and practice*. Oxford University Press, Oxford, 1996.
 5. Thomas Bäck, David Fogel, and Zbigniew Michalewicz. *Evolutionary Computation*. IoP Publishing, Bristol, 2000.
 6. M. Beretta and A. Tettamanzi. Learning fuzzy classifiers with evolutionary algorithms. In G. Pasi A. Bonarini, F. Masulli, editor, *Soft Computing Applications*, pages 1–10. Physica Verlag, Heidelberg, 2003.
 7. Zvi Bodie, Alex Kane, and Alan J. Marcus. *Investments (7th Edition)*. McGraw-Hill, New York, 2006.
 8. Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
 9. Erik Cantú-Paz. A survey of parallel genetic algorithms. Technical Report IlliGAL 97003, University of Illinois at Urbana-Champaign, 1997.
 10. Kenneth A. DeJong. *Evolutionary Computation: A unified approach*. MIT Press, Cambridge, MA, 2002.
 11. M. Dempster and C. Jones. A real-time adaptive trading system using genetic programming. *Quantitative Finance*, 1:397–413, 2001.
 12. M. Dempster, C. Jones, Y. Romahi, and G. Thompson. Computational learning techniques for intraday fx trading using popular technical indicators. *IEEE Transactions on Neural Networks*, 12(4), 2001.
 13. H. V. Harlow. Asset allocation in a downside-risk framework. *Financial Analysts Journal*, pages 30–40, September/October 1991.
 14. H. Hellendoorn and C. Thomas. Defuzzification in fuzzy controllers. *Intelligent and Fuzzy Systems*, 1:109–123, 1993.
 15. T. Hellstrom and K. Holmstrom. Parameter tuning in trading algorithms using asta. *Computational Finance*, 1:343–357, 1999.
 16. E. H. Mamdani. Advances in linguistic synthesis of fuzzy controllers. *International Journal of Man Machine Studies*, 8:669–678, 1976.
 17. B. Manderick and P. Spiessens. Fine-grained parallel genetic algorithms. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, San Mateo, CA, 1989. Morgan Kaufmann.
 18. J. P. Marney, C. Fyfe, H. Tarbert, and D. Miller. Risk-adjusted returns to technical trading rules: A genetic programming approach. *Computing in Economics and Finance* 147, Society for Computational Economics, Yale University, USA, June 2001.
 19. H. Mühlenbein. Parallel genetic algorithms, population genetics and combinatorial optimization. In J. D. Schaffer, editor, *Proceedings of the Third International Conference on Genetic Algorithms*, pages 416–421, San Mateo, CA, 1989. Morgan Kaufmann.
 20. John J. Murphy. *Technical Analysis of the Financial Markets*. New York Institute of Finance, New York, 1999.
 21. C. Neely, P. Weller, and R. Dittmar. Is technical analysis in the foreign exchange market profitable? a genetic programming approach. *Journal of Financial and Quantitative Analysis*, 32:405–26, 1997.
 22. Michael O’Neill, Anthony Brabazon, Conor Ryan, and J. J. Collins. Evolving market index trading rules using grammatical evolution. In Egbert J. W. Boers *et al.*, editor, *Applications of Evolutionary Computing, EvoWorkshops 2001: Proceedings*, volume 2037 of *Lecture Notes in Computer Science*, pages 343–352. Springer, 2001.

23. Jean-Yves Potvin, Patrick Soriano, and Maxime Vallée. Generating trading rules on the stock markets with genetic programming. *Computers and Operations Research*, 31(7), June 2004.
24. A. Tettamanzi R. Poluzzi, G. G. Rizzotto. An evolutionary algorithm for fuzzy controller synthesis and optimization based on sgs-thomson's W.A.R.P. fuzzy processor. In L. A. Zadeh E. Sanchez, T. Shibata, editor, *Genetic algorithms and fuzzy logic systems: Soft computing perspectives*. World Scientific, Singapore, 1996.
25. T.J. Ross. *Fuzzy Logic with Engineering Applications*. McGraw-Hill, New York, 1995.
26. William F. Sharpe. The Sharpe ratio. *Journal of Portfolio Management*, 21(1):49–58, 1994.
27. Frank A. Sortino and R. van der Meer. Downside risk — capturing what's at stake in investment situations. *Journal of Portfolio Management*, 17:27–31, 1991.
28. Harish Subramanian, Subramanian Ramamoorthy, Peter Stone, and Benjamin J. Kuipers. Designing safe, profitable automated stock trading agents using evolutionary algorithms. In Mike Cattolico, editor, *Genetic and Evolutionary Computation Conference, GECCO 2006, Proceedings, Seattle, Washington, USA, July 8-12, 2006*, pages 1777–1784. ACM, 2006.
29. A. Tettamanzi. An evolutionary algorithm for fuzzy controller synthesis and optimization. In *IEEE International Conference on Systems, Man and Cybernetics*, volume 5/5, pages 4021–4026. IEEE Systems, Man, and Cybernetics Society, 1995.
30. Andrea G. B. Tettamanzi, Maria Carlesi, Lucia Pannese, and Mauro Santalmasi. Business intelligence for strategic marketing: Predictive modelling of customer behaviour using fuzzy logic and evolutionary algorithms. In Mario Giacobini *et al.*, editor, *Applications of evolutionary computing: Evoworkshops 2007*, volume 4448 of *Lecture Notes in Computer Science*, pages 233–240, Valencia, Spain, April 11–13 2007. Springer.
31. D. Whitley, S. Rana, and R. B. Heckendorn. The island model genetic algorithm: On separability, population size and convergence. *Journal of Computing and Information Technology*, 7(1):33–47, 1999.
32. Wikipedia. Technical analysis — wikipedia, the free encyclopedia, 2007. [Online; accessed 5-June-2007].
33. X. Yao and Y. Xu. Recent advances in evolutionary computation. *Computer Science and Technology*, 21(1):1–18, January 2006.
34. Tina Yu, Shu-Heng Chen, and Tzu-Wen Kuo. Discovering financial technical trading rules using genetic programming with lambda abstraction. In Una-May O'Reilly, Tina Yu, Rick Riolo, and Bill Worzel, editors, *Genetic Programming Theory and Practice II*, pages 11–30. Springer, 2006.
35. L. A. Zadeh. Fuzzy sets. *Information and Control*, 8:338–353, 1965.
36. L. A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning, i–ii. *Information Science*, 8:199–249, 301–357, 1975.
37. L. A. Zadeh. The calculus of fuzzy if-then rules. *AI Expert*, 7(3):22–27, March 1992.