AUTOMATED TRADING WITH BOOSTING AND EXPERT WEIGHTING

Germán Creamer
CENTRUM Católica
Pontificia Universidad Católica del Peru
email: ggc14@columbia.edu

Yoav Freund
Department of Computer Science and Engineering
University of California, San Diego
9500 Gilman Drive, Mail Code 0404
La Jolla, CA 92093-0404
email: yfreund@cs.ucsd.edu

ABSTRACT
We propose a multi-stock automated trading system that relies on a layered structure consisting of a machine learning algorithm, an online learning utility, and a risk management overlay. Alternating decision tree (ADT), which is implemented with Logitboost, was chosen as the underlying algorithm. One of the strengths of our approach is that the algorithm is able to select the best combination of rules derived from well-known technical analysis indicators and is also able to select the best parameters of the technical indicators. Additionally, the online learning layer combines the output of several ADTs and suggests a short or long position. Finally, the risk management layer can validate the trading signal when it exceeds a specified non-zero threshold and limit the application of our trading strategy when it is not profitable.

We test the expert weighting algorithm with data of 100 randomly selected companies of the S&P 500 index during the period 2003-2005. We find that this algorithm generates abnormal returns during the test period. Our experiments show that the boosting approach is able to improve the predictive capacity when indicators are combined and aggregated as a single predictor. Even more, the combination of indicators of different stocks demonstrated to be adequate in order to reduce the use of computational resources, and still maintain an adequate predictive capacity.

KEY WORDS
Automated trading, machine learning, algorithmic trading, boosting.

1 Introduction

The major stock exchanges such as NYSE and NASDAQ are transforming their markets into electronic financial markets. Many traders in these markets must rely on automated trading systems in order to process large amounts of information and make instantaneous investment decisions.

The early automated trading systems embedded classical artificial intelligence approaches such as expert systems, neural networks and genetic algorithms (see Trippi and Turban [64] and Trippi and Lee [63] for a review of these systems).

Generally, the current automated trading systems include a backtest or simulation module. In this respect, the models developed by the agent-based perspective could be useful to explore new ideas without risking any money. The Santa Fe stock market model[1], has inspired many other agent-based financial market models such as Ehrentreich [27] that is based on the Grossman and Stiglitz model [41]. In the Santa Fe stock market agents can classify and explore several forecasting rules that are built using genetic algorithms. Many of the models built in this perspective test the performance of agents or algorithms that have unique characteristics. For example, Lettau [51] builds an agent-based financial market using simple agent benchmarks based on genetic algorithms; Gode and Sunder [39] develop a double action market using zero intelligence traders; Arifovic [4] builds a model of the foreign exchange market using genetic algorithms; Routledge [56] extends the basic framework of Grossman and Stiglitz [41] with agents that can learn using genetic algorithms; Chan et al. [14] and Chan [15] use the artificial market framework to explore the behavior of different trading approaches and their microstructure impact. The application of the Ising model [42] to financial markets has led to several versions of the spin model where a sell position is a spin-up and a buy position is a spin-down. Prices are determined by the aggregation of traders' positions.[2]

These agent-based models are useful for simulation, however they do not automate the trading functions. On the contrary, Seo et al. [58] and Decker [23] describe a multi-agent portfolio management system that automatically classifies financial news. In this line of research, Thomas [61] combines news classification with technical analysis indicators in order to generate new trading rules. Lavrenko et al. [46] describe a system that recommends news stories that can affect market behavior. This is a special case of activity monitoring task as suggested by Fawcett and Provost [32]. In a similar way as fraud detection systems operate, activity monitoring generates alarms

[1] For a presentation of the Santa Fe stock market model see Arthur et al. [5], LeBaron et al. [50], and a later version at Lebaron [49]. Lebaron [48] has also a general review of papers in the area of agent-based finance.

[2] Bornholdt [10] modified this model introducing an anti-ferromagnetic coupling between the global magnetization and each spin, as well as a ferromagnetic coupling between the local neighborhood and each spin.

when an unusual event happens. These signals try to recognize when the trend of the market is positive, and therefore can generate new trading signals. Wuthrich et al. [67] and Cho, Wuthrich, and Zhang [17] weights keywords based on its occurrences to predict the direction of major stock indices. Text classification and retrieval applied to finance is still an area under-explored in the literature. However, several investment banks and hedge funds are developing systems to automatically incorporate the impact of daily news into their trading systems. Our current work does not incorporate the news aspect, however the methods used to automate forecasting, trading or the extraction of new indicators are an important antecedent for our work in automated trading systems.

1.1 Constant rebalanced portfolio and universal portfolio

Constant rebalanced portfolio (CRP), known in the financial world as constant mix, is a well-known strategy in the investment community. Kelly [45] showed that individuals that invest the same proportion of their money on a specific asset–the constant rebalanced portfolio–their portfolio value will increase (or decrease) exponentially. Kelly introduced the log-optimal portfolio as the one which achieves the maximum exponential rate of growth. Algoet and Cover [2] showed that if the market is stationary ergodic, the maximum capital growth rate of a log-optimal portfolio is equivalent to the maximum expected log return. CRP simply requires that traders maintain a fixed proportion of stocks to portfolio value. If stock price increases (decreases), the stock to portfolio value ratio increases (decreases), then part of the stocks must be liquidated (bought). This strategy works better when the stock price is unstable, so the trader is able to sell when the price is high, and buy when the price is low.

Cover [21] and later on many other researchers such as Vovk and Watkins [66], Cover and Ordentlich [20], Blum and Kalai [7], and Kalai and Vempala [43] extended CRP to the concept of universal constant rebalanced portfolio. Cover's "universal portfolio" (CUP) algorithm combines many CRPs each of them with its own probability distribution. The trading algorithm that we introduce in section 3.1 is similar to CUP because of its dependence on the data, and does not depend upon underlying investment theories or statistical assumptions. Our algorithm also combines different recommendations of experts or "artificial" fund managers. However, the definition of experts and their combination is completely different of how it is done in CUP. Our algorithm is based on a learning algorithm and has an online learning capacity that assigns different weights according to the performance of its experts. As a result, the portfolio is reallocated based on the risk adjusted return of its underlying assets, and the combination of experts' forecasts.

1.2 Trading strategies and technical analysis

Another important line of research in the trading algorithmic literature is the use of learning algorithms to generate trading rules using technical analysis indicators. Technical analysis or technical trading strategies try to exploit statistically measurable short-term market opportunities, such as trend spotting, and momentum, in individual industrial sectors (e.g. financial, pharmaceutical etc.).

The presence of technical analysis has been very limited in the finance literature because of its lack of a solid statistical or mathematical foundation, its highly subjective nature, and its visual nature. In the 60s and 70s researchers studied trading rules based on technical indicators and did not find them profitable [1] [31]. These findings led Fama [30] to dismiss technical analysis as a profitable technique and support the efficient market hypothesis. Part of the problem of the studies during the 60s was the *ad hoc* specifications of the trading rules that led to data snooping. Ex post specification of rules may have led to biased studies. Recently, Allen and Karjalainen [3] found profitable trading rules using genetic algorithm for the S&P 500 with daily prices from 1928 to 1995. However, these rules were not consistently better than a buy-and-hold strategy in the out-of-sample test periods.

In the last years, there has been a growing interest on applying machine learning methods to formulate trading strategies using technical indicators such as the following: Lo, Mamaysky, and Wang [53], who used nonparametric kernel regression for technical pattern recognition of a large number of stocks for the period 1962 - 1996, found that technical indicators provide incremental information for investors comparing the unconditional empirical distribution of daily stock returns to the conditional distribution on specific technical indicators such as head and shoulders. Moody and Saffell [54] found that a trading system using direct reinforcement learning outperforms a Q-trader for the asset allocation problem between the S&P 500 and T-bill. Dempster et al. [25] compared four methods for foreign exchange trading (reinforcement learning, genetic algorithm, Markov chain linear programming, and simple heuristic) and concluded that a combination of technical indicators using genetic algorithm leads to better performance than using only individual indicators for the foreign exchange market. Dempster and Leemans. [24] reached a similar conclusion using adaptive reinforcement learning. Bates et al. [6] used Watkin's Q-learning algorithm to maximize profits; these authors compared order flow and order book data, and compared with technical trading rules. They concluded that using order flow and order book data was usually superior to trading on technical signal alone. LeBaron [47] applied bootstrapping to capture arbitrage opportunities in the foreign exchange market, and then used a neural network where its network architecture was determined through an evolutionary process. Finally, Towers and Burgess [62] used principal components to capture arbitrage opportunities.

In this research we follow the tradition of the papers in this section that use machine learning algorithms to find profitable trading strategies, and also build completely automated trading systems.

The rest of the paper is organized as follows: section 2 introduces boosting; section 3 introduces the trading system; section 4 explains in detail the experiments; section 5 presents the results of our trading algorithm, section 6 presents the conclusions and final comments, and the appendix introduces the main investment indicators used in this research.

2 Methods

2.1 Boosting

Adaboost is a general discriminative learning algorithm invented by Freund and Schapire [36].

The basic idea of Adaboost is to repeatedly apply a simple learning algorithm, called the *weak* or *base* learner[3], to different weightings of the same training set. In its simplest form, Adaboost is intended for binary prediction problems where the training set consists of pairs $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$, x_i corresponds to the features of an example, and $y_i \in \{-1, +1\}$ is the binary label to be predicted. A *weighting* of the training examples is an assignment of a non-negative real value w_i to each example (x_i, y_i).

On iteration t of the boosting process, the weak learner is applied to the training set with a set of weights $w_1^t, \ldots, w_m^t$ and produces a prediction rule h_t that maps x to $\{0, 1\}$. [4] The requirement on the weak learner is for $h_t(x)$ to have a small but significant correlation with the example labels y when measured using the *current weighting of the examples*. After the rule h_t is generated, the example weights are changed so that the weak predictions $h_t(x)$ and the labels y are decorrelated. The weak learner is then called with the new weights over the training examples, and the process repeats. Finally, all of the weak prediction rules are combined into a single *strong* rule using a weighted majority vote. One can prove that if the rules generated in the iterations are all slightly correlated with the label, then the strong rule will have a very high correlation with the label – in other words, it will predict the label very accurately.

The whole process can be seen as a variational method in which an approximation $F(x)$ is repeatedly changed by adding to it small corrections given by the weak prediction functions. In Figure 1, we describe Adaboost in these terms. We shall refer to $F(x)$ as the *prediction score* in the rest of the document. The strong prediction rule learned by Adaboost is $\text{sign}(F(x))$.

A surprising phenomenon associated with Adaboost is that the test error of the strong rule (percentage of mistakes made on new examples) often continues to decrease even after the training error (fraction of mistakes made on the training set) reaches zero. This behavior has been related to the concept of a "margin", which is simply the value $yF(x)$ [57]. While $yF(x) > 0$ corresponds to a correct prediction, $yF(x) > a > 0$ corresponds to a *confident* correct prediction, and the confidence increases monotonically with a. Friedman et al. [37], followed by Collins,

---

$F_0(x) \equiv 0$
for $t = 1 \ldots T$
  $w_i^t = e^{-y_i F_{t-1}(x_i)}$
  Get h_t from *weak learner*
  $\alpha_t = \frac{1}{2} \ln \left(\frac{\sum_{i:h_t(x_i)=1, y_i=1} w_i^t}{\sum_{i:h_t(x_i)=1, y_i=-1} w_i^t} \right)$
  $F_{t+1} = F_t + \alpha_t h_t$

Figure 1. The Adaboost algorithm [36]. y_i is the binary label to be predicted, x_i corresponds to the features of an instance i, w_i^t is the weight of instance i at time t, h_t and $F_t(x)$ are the prediction rule and the prediction score at time t respectively

Schapire, and Singer [19] suggested a modification of Adaboost, called Logitboost. Logitboost can be interpreted as an algorithm for step-wise logistic regression. Figure 2 describes Logitboost using notation similar to the one used in 1. Some succesful and popular way of using boosting is to combine it with a decision tree learning algorithms as the base learning algorithm [37]. We use boosting both to learn the decision rules constituting the tree and to combine these rules through a weighted majority vote. The form of the generated decision rules is called an *alternating decision tree* (ADT) [35]. In ADTs each node can be understood in isolation.

We explain the structure of ADTs using Figure 3. The problem domain is corporate performance prediction,

---

$F_0(x) \equiv 0$
for $t = 1 \ldots T$
  $w_i^t = \frac{1}{1+e^{y_i F_{t-1}(x_i)}}$
  Get h_t from *weak learner*
  $\alpha_t = \frac{1}{2} \ln \left(\frac{\sum_{i:h_t(x_i)=1, y_i=1} w_i^t}{\sum_{i:h_t(x_i)=1, y_i=-1} w_i^t} \right)$
  $F_{t+1} = F_t + \alpha_t h_t$

Figure 2. The Logitboost algorithm [37]. y_i is the binary label to be predicted, x_i corresponds to the features of an instance i, w_i^t is the weight of instance i at time t, h_t and $F_t(x)$ are the prediction rule and the prediction score at time t respectively

[3] Intuitively, a weak learner is an algorithm with a performance at least slightly better than random guessing

[4] Mapping x to $\{0, 1\}$ instead of $\{-1, +1\}$ increases the flexibility of the weak learner. Zero can be interpreted as "no prediction".

and the goal is to separate stocks with high and low values based on 17 different variables. The tree consists of alternating levels of ovals (*prediction nodes*) and rectangles (*splitter nodes*) (hence the word "alternating" in the name). The first number within the ovals defines contributions to the prediction score, and the second number (between parentheses) indicates the number of instances. In this example, positive contributions are evidence of high performance, while negative contributions are evidence of corporate financial problems. To evaluate the prediction for a particular company we start at the top oval (0.04) and follow the arrows down. We follow *all* of the dotted arrows that emanate from prediction nodes, but we follow *only one* of the solid-line arrows emanating from a splitter node, corresponding to the answer (yes or no) to the condition stated in rectangle. We sum the values in all the prediction nodes that we reach. This sum represents the prediction score $F(x)$ above, and its sign is the final, or strong, prediction. For example, suppose we had a company for which

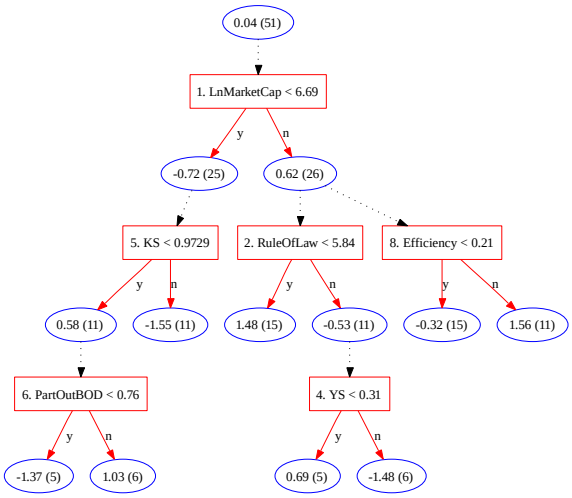


Figure 3. LAADR: representative ADT.

LnMarketCap=6, KS=0.86, RuleOfLaw=7.02, and PartOutBOD=0.76. In this case, the prediction nodes that we reach in the tree are 0.042, -0.7181, 0.583, and 1.027. Summing gives a score of 0.9339, i.e., a very confident indicator that the company has a high market value.

This example demonstrates how alternating decision trees combine the contributions of many indicators to generate a prediction. The ADT in the figure was generated by Adaboost from training data. In terms of Adaboost, each prediction node represents a weak prediction rule, and at every boosting iteration, a new splitter node together with its two prediction nodes is added to the model. The splitter node can be attached to any previous prediction node, not only leaf nodes. Each prediction node is associated with a weight α that contributes to the prediction score of every example reaching it. The weak hypothesis $h(x)$ is 1 for every example reaching the prediction node and 0 for all others. The number in front of the conditions in the splitter nodes of Figure 3 indicates the iteration number on which the node was added. In general, lower iteration numbers indicate that the decision rule is more important.

We decided to use boosting, specifically Logitboost, as our learning algorithm because of its feature selection capability, its error bound proofs [36], its interpretability, and its capacity to combine quantitative, and qualitative variables. Additionally, we used ADT implemented with Logitboost because of its capacity to generate classification rules that are smaller and easier to interpret than the rules generated by other boosting decision trees learning algorithms such as CART or C4.5. Additionally, ADT gives a measure of confidence or classification margin which is very useful for experts selection.

3 The trading system

The trading system is designed to trade stocks and relies on a layered structure consisting of a machine learning algorithm, an online learning utility, and a risk management overlay.[5] ADT, which is implemented with Logitboost, was chosen as the underlying algorithm. One of the strengths of our approach is that the algorithm is able to select the best combination of rules derived from well-known technical analysis indicators and is also able to select the best parameters of the technical indicators. Additionally, the online learning layer combines the output of several ADTs and suggests a short or long position. Finally, the risk management layer can validate the trading signal when it exceeds a specified non-zero threshold and limit the application of our trading strategy when it is not profitable (see Figure 4).

3.1 Layer 1: The machine learning algorithm and its implementation

The inputs to Logitboost, the learning algorithm, were a group of well-known technical indicators and investment signals introduced in the appendix: simple and exponential moving average, Bollinger bands, acceleration, momentum, rate of change, moving average convergence divergence, relative strength index, stochastic oscillators, Williams indicator, money flow index, on balance volume, accumulation/distribution line, Chaikin oscillator, negative and positive volume index, and price-volume trend. Additionally, we included the Sharpe ratio as a performance indicator and several measures of volatility such as GARCH, Chaikin, and Garman-Klass volatility. The instances were weighted by the Sharpe ratio. We also included ratios and trading rules suggested by the practice of technical analy-

[5] See Dempster and Leemans [24] for a previous trading system using machine learning algorithms and a layered structure.

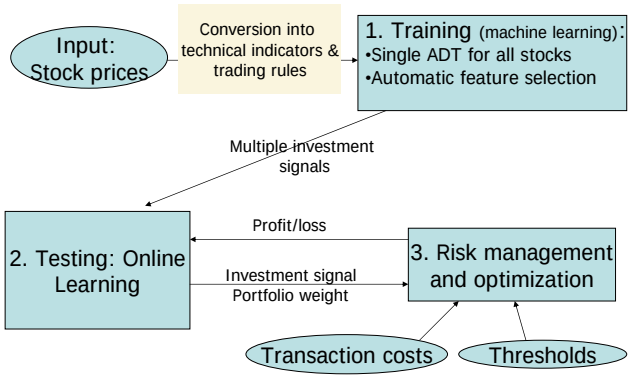


Figure 4. Trading system as a process flow. The inputs of the algorithm are price series that are transformed into technical indicators and trading rules. The first component trains several ADTs (experts) and each of them generates an investment signal. The second layer weights the investment signals of each ADT, and generates a weighted single investment signal for each stock using an online learning algorithm. The third layer filters the weak investment signals, and restricts not profitable trading strategies.

sis. We calculated these indicators using R and its financial engineering package called Rmetrics.[6]

Our goal was to predict the trend of beta excess returns ($BXRET$)[7] using the above investment signals. $y_t \in [-1, +1]$ is the binary label to be predicted where 1 represents the expectation of a positive beta excess return, and -1 otherwise.

A major problem with the use of technical indicators is their calibration or the adequate selection of their best parameters, as with the optimal number of days to calculate the moving averages used by stochastic indicators or the number of standard deviations used to calculate the Bollinger bands. There are many versions of optimizers such as the "brute force" approach where all the alternatives are tested one-at-a-time, and simulated annealing or genetic optimizers as suggested by Katz and McCormick [44]. In our case, we initially tried one-at-a-time optimization where each parameter was tested with several values while keeping the others constant. We found that this approach was very inefficient because it required too much computer power and time.

The solution that we implemented was the simultaneous recalculation of the technical indicators with several values of their parameters. Then the boosting algorithm would select the best combination of parameters and technical indicators. Our initial parameters were the parameters recommended in the literature. We tested 16 different variations of the initial parameters and did not find a major difference when we used only four different variations of the parameters. So, we included four different versions of most of the technical indicators in our training, validation, and testing set. In total, each instance of the training, validation, and testing set had 112 inputs. We were able to use this approach because of the boosting's feature selection capability (see the appendix for a presentation of the parameters used).

In our experiments we observed that a small change in the sample may lead to diferent ADTs changing the investment recommendation. In order to prevent this problem, our algorithm generates different ADTs and, in the next layer, selects the output of the best ADTs.

3.2 Layer 2: Online learning and expert weighting

We improved the performance of the boosting algorithm by adding online learning capacity with expert weighting. Our algorithm is derived from a previous algorithm proposed by Freund, Manssour and Schapire [34] that predicts with a exponentially weighted average of the training error of all hypotheses. The "exponential weights" formula comes from the weighted majority algorithm introduced by Littlestone and Warmuth [52] and further studied by Cesa-Bianchi et al. [12]. The "exponential weights" is a different formula to compute the posterior distribution as it would be proposed by Bayesian analysis. An interesting feature of this algorithm is that it abstains to predict on certain instances, so the predictions that it makes are very reliable. The algorithm that we present in the next subsection is an extension of this original algorithm applied to financial time series prediction, and therefore is an online learning model.[8] The machine learning layer introduced in the previous section and the online learning layer of our trading system have two different functions:

- The machine learning layer generates several "experts" or ADTs using historical stock price information. Each ADT has a set of rules that combines the technical indicators, and generates an investment signal for each stock.

- The online learning layer receives the new prices, combines the signals of the different ADTs generated by the machine learning layer, and reallocates the portfolio according to the recommendations of the "experts", and the input received from the risk management module (see section 3.3.1).

In the next section, we describe the algorithm used by the online learning layer.

[6] Information about R and Rmetrics can be found at <http://cran.r-project.org> and at <http://www.rmetrics.org> respectively.

[7] Beta excess return is the excess return of a specific asset less the average return of all assets in its beta portfolio for each trading date.

[8] Borodin and El-Yaniv [11, 28] propose a different approach to use online learning for trading and portfolio selection. They measure the performance of their trading algorithm in relation to a "statistical adversary", and an optimal offline algorithm.

3.2.1 The expert weighting algorithm

To simplify the presentation of our algorihtm, we introduce the case of one asset which can easily be extended to N assets. The final outcome sequence are the net weights of an asset ($W = W_1, W_2, .., W_t, ..., W_l$) where $W_t \in [-1, 1]$, t refers to a time step and l represents the last step of the sequence.

In this research, every expert is an ADT calculated with the training set which is a sequence of pairs as introduced in section 2.1. We refer to the sequence of experts by $\psi = \psi_1, \psi_2, ..., \psi_E$ where E is the number of experts.

The outcome of expert ψ_i at time t is the prediction score S_t^i. In order to calculate the experts' weight, we need to calculate the cumulative abnormal return of each expert i (ψ_i) as:

$$car_t^i \doteq \sum_{s=t_{i+1}}^{t} \text{sign}(S_s^i) \cdot \text{r}_s^i$$

where $t_1 = 0$, t_i is the time step at which ψ_i was added to the pool when $i > 1$, and r_s^i is the abnormal return for expert i at time s.

car_t^i is a sum of random variables that are close to $\mathcal{N}(\epsilon, 1)$ where ϵ is the slight advantage that the expert has over random guessing. Our goal is to give higher weight to experts that have higher ϵ. On the one hand, ϵ is masked by the noise, on the other hand, the noise increases only as $\sqrt{t}$ while the drift increases like $\epsilon\ t$.

The weight of the first expert is:

$$w_t^1 \doteq \exp\left(\frac{C \cdot car_t^1}{\sqrt{t}}\right)$$

where C is an exogenous parameter.

The weight of expert ψ_i at time t where $t > t_i$ is:

$$w_t^i \doteq I_i \cdot \text{ramp}(t - t_i) \cdot \exp\left(\frac{C \cdot car_t^i}{\sqrt{t - t_i}}\right)$$

where $I_i \doteq \frac{\sum_{j=1}^{i-1} w_{t_i}^j}{i-1}$ is the initial weight assigned to the new expert i (ψ_i) and is the average weight of the previous experts at the time that the new expert is added (t_i), $\text{ramp}(t - t_i) \doteq \min\left(\frac{t-t_i}{t_{t+1}-t_i}, 1\right)$ is a function that brings in the new expert gradually, and t_{i+1} is the time that the next expert is added.

The experts' weight (W_t) is the result of weighting the answers of all the experts:

$$W_t = L_t - S_t$$

where the fraction of experts that suggest a long or short position is $L_t = \frac{\sum_{i:S_t^i>0} w_t^i}{\sum_i w_t^i}$ or $S_t = 1 - L_t^i$, respectively. Therefore, $W_t \in [-1, +1]$ is also a trading suggestion to take either a long ($W_t > 0$), hold ($W_t = 0$) or a short position ($W_t < 0$).

The expert weighted cumulative abnormal return is:

$$CAR \doteq \sum_t (W_t \cdot r_t - (W_t - W_{t-1}) \cdot tc)$$

where tc are transaction costs.

We call this version of the algorithm as the "Base" version. Additionally, we include two variations of the algorithm where we modify the denominator of the weight functions as:

1. "Simple time adjustment" version where:

$$w_t^i \doteq I_i \cdot \text{ramp}(t - t_i) \cdot \exp\left(\frac{C \cdot car_t^i}{t - t_i}\right)$$
$$w_t^1 \doteq \exp\left(\frac{C \cdot car_t^1}{t}\right)$$

2. "No time adjustment" version where:

$$w_t^i \doteq I_i \cdot \text{ramp}(t - t_i) \cdot \exp\left(C \cdot car_t^i\right)$$
$$w_t^1 \doteq \exp\left(C \cdot car_t^1\right)$$

3.3 Layer 3: Risk management and optimization

An important aspect of the layered structure of the trading system is that the decision to trade is separated from the trade recommendation made by layers 1 and 2. While layer 1 and 2 suggest a preferred long or short position, layer 3 evaluates this position by considering market factors before taking an investment decision. Any trading system requires risk management rules, however they are difficult to include in the trading module itself, so it was easier to include them in an independent layer.

The risk management layer evaluates the strength of the trading signal W_t given by layer 2. As W_t is normalized between a completely short (-1) and completely long (1) position, the risk management system eliminates the trading signal that are not above (for long positions) or below (for short positions) a non-zero threshold γ_0. This parameter is fixed based on the information generated during the training and optimization stage.

A common complaint about automated trading systems is that they are not always profitable because market conditions change, and therefore what used to be adequate in certain period of time is not in another moment. In this respect, an indicator to evaluate the performance of an algorithm used by traders is the maximum drawdown ($D_{i,t}$) [25, 54]. This indicator defined for stock i over a period $t - t_0$ is:

$$D_{i,t} \doteq max(R_{i,t_x} - R_{i,t_y} | t_0 \leq t_x \leq t_y \leq t)$$

R_{i,t_x} and R_{i,t_y} are the accumulated return from time t_0 until time t_x and t_y respectively for stock i. We denote

the vector of the maximum drawdown for all stocks of our portfolio as D_t. The maximum drawdown is the largest potential loss in a certain period of time. If $D_{i,t} < \gamma_1$ for a certain period of time (thirty trading days), then the system holds its current position, and if $D_{i,t-1} == min(D_{t-1})$, the system liquidates the current stock position. If these conditions improve, the stock can be traded again. After the first thirty trading days, which is even part of the validation period, these rules are continuously applied. The rationality for the first risk management rule is very obvious: the trading system should not invest more in a strategy that has demonstrated to be unprofitable for a specific stock. Considering that the market conditions may change, the trading system only holds the position but does not liquidate it. If the situation is even worse, and if the maximum drawdown of a stock i is the maximum drawdown among all the selected stocks, the trading system liquidates the position. We think that with a large porfolio a relative indicator is useful because it incorporates market conditions. However, an additional rule, common among trading desks, that the system stops trading or liquidates its current position when its return is below a certain nominal amount could also coexist with the above rules.

We also used the Sharpe ratio (SR_t) and Sterling ratio (St_t) to evaluate the risk-adjusted return of our experts. The Sharpe ratio is annualized and is calculated as the mean of monthly returns divided by the standard deviation of the monthly returns:

$$SR_t \doteq \frac{\mu(R_t)}{\sigma(R_t)}$$

The Sharpe ratio could be very unstable for small return variances, and in a long period it cannot distinguish between shorter periods where there are large profits or losses. This is the reason that traders also incorporate the maximum drawdown in their risk analysis. The maximum drawdown helps to recognize those clusters of profits and losses that are undetected by the Sharpe ratio. In this respect, the Sterling ratio brings additional information that is not captured by the Sharp ratio when it includes the annualized return to maximum drawdown ratio:

$$St_t \doteq \frac{R_t}{D_t}$$

3.3.1 Optimization and training

We divided our time series between a training, validation, and test set. The training set was used to generate the first expert or ADT. The objective of the validation set is to choose the appropriate version of the expert weighting algorithm, and the optimal parameters C, γ_0, and γ_1 introduced in the previous section. Parameter C is an exogenous parameter that modifies the exponential weight of the experts; γ_0 is a non-zero threshold used to filter weak investment signals, and γ_1 is a threshold to detect large drawdowns in order to stop trading. In section 5, we present the result of all the versions of the model for comparative reasons. Based on the initial expert, optimal parameters, and test set, we conducted the experiments that we describe in the next section.

The complete algorithm that includes the three layers of the trading system is presented in Figure 5.

4 Experiments

Before trading started, we selected a random sample of 100 stocks of the S&P 500 index with daily data from January 2001 through December 2004. For every stock we had a complete series of close, open, high and low prices, volume, and $BXRET$. Companies that did not have the variable $BXRET$ were eliminated. We obtained these price series from CRSP. The time series were distributed in the following way:

- Training set included two years (500 trading days). This calculation required about 540 trading days because of about 40 lagged values that were discarded. This dataset includes data from October 2000 to December 2002. We aggregated in one set the observations of all the stocks. So, our training set had 50,000 observations. In previous tests, we tried the generation of individual ADTs for each stock, however this solution used a significant amount of computer power and time, and the results were still very similar. We also think that a single tree for all stocks is richer than individual trees for each stock because the former captures the diverse patterns of each company.

- Validation set is based on 100 trading days (10,000 observations). This dataset includes data from December 2002 to May 2003.

- Test set had the remaining observations (400 trading days or 40,000 observations). This dataset includes data from May 2003 to December 2005.

We run our algorithm using a moving window of two years for training followed by about two months out-of-sample testing (50 days). We conducted our tests on a daily rollover basis. Even though we trained a new expert (ADT) only every 50 trading days, we tested the existent experts every day with the additional price information. The algorithm presented in Figure 5 reweighted the participation of each expert according to individual performance and time transcurred since the initial training. Additionally, the risk management module would hold or liquidate positions when they were not profitable or became too risky.

The selection of 50 trading days between training periods was partially justified by the fact that using longer periods (100 days) or shorter periods (25 days) did not generate significant differences with the results obtained. However, the computational overhead was very important when the days between training were reduced. The training of every expert required about 45 minutes and the testing of

Input:
Set of price series (close, open, high, and low), volume and beta excess return ($BXRET$)
r is number of different values of parameters to calculate investment signals
N is number of stocks to be selected from market
d is number of days between experts' training
γ_0 and γ_1 are thresholds to filter experts' weight.
C is an exogenous parameter for expert weighting.

Train with machine learning algorithm:
1. Select a representative sample of N number of stocks from targeted market.

2. Calculate investment signals, and labels [a] with basic parameters for a selected group of stocks.

3. Recalculate investment signals with r variations of basic parameters, and include all of the investment signals as features in the training and test sets where the binary label is $y_t = sign(BXRET)$.

4. Integrate all the instances of the N stocks in a single training and single test set.

5. Train an initial expert (ψ_1) with Logitboost. Every d days train a new expert i ψ_i. Call the sequence of experts as $\psi = \psi_1, \psi_2, ..., \psi_E$ where E is the number of experts.

6. Every day recalculate test set and weight experts as in next steps.

Expert weighting algorithm (this part is simplified for one asset, even though can be extended to N assets):
6. Calculate the weight of the first expert at time t as $w_t^1 \doteq \exp\left(\frac{C \cdot car_t^1}{\sqrt{t}}\right)$ where:
   a. $car_t^i \doteq \sum_{s=t_{i+1}}^{t} \text{sign}(S_s^i) \cdot \text{r}_s^i$
   b. $t_1 = 0$, t_i is the time step at which ψ_i is calculated when $i > 1$
   c. r_s^i is the abnormal return for expert i at time s.
7. Calculate the weight of expert ψ_i at time $t > t_i$ as $w_t^i \doteq I_i \cdot \text{ramp}(t - t_i) \cdot \exp\left(\frac{C \cdot car_t^i}{\sqrt{t-t_i}}\right)$ where:
   a. $I_i \doteq \frac{\sum_{j=1}^{i-1} w_{t_i}^j}{i-1}$ is the initial weight assigned to i (ψ_i)
   b. $\text{ramp}(t - t_i) \doteq \min\left(\frac{t-t_i}{t_{t+1}-t_i}, 1\right)$
   c. t_{i+1} is the time that the next expert is added.

8. Calculate the experts' weight as $W_t = L_t - S_t$ where $L_t = \frac{\sum_{i:S_t^i>0} w_t^i}{\sum_i w_t^i}$ and $S_t = 1 - L_t^i$.

Risk management:
9. if $|W_t| < \gamma_0$, then $W_t = 0$

10. If for stock i, $D_{i,t-1} == min(D_{t-1})$ then $W_{i,t} = 0$, else if $D_{i,t-1} < \gamma_1$ then $W_{i,t} = W_{i,t-1}$ where:
   a. $D_{i,t} \doteq max(R_{i,t_x} - R_{i,t_y} | t_0 \leq t_x \leq t_y \leq t)$ is the maximum drawdown for stock i
   b. R_{i,t_x}, R_{i,t_y}, and $W_{i,t}$ are the accumulated return from time t_0 until time t_x and t_y, and experts' weights for stock i respectively
   c. D_t is the matrix of maximum drawdowns for selected stocks

Output:
Expert weighted cumulative abnormal return for stock i is:
$CAR_i \doteq \sum_t (W_{i,t} \cdot r_{i,t} - (W_{i,t} - W_{i,t-1}) \cdot tc_i)$ where tc_i are transaction costs for stock i.

Figure 5. Forecasting and trading algorithm.

[a] In this research we used all the investment signals described in the appendix.

every day lasted about two to three minutes with one expert and about eight to ten minutes with the eleven experts that we had in total.

We aggregated the daily results in 20 test sets (2,000 observations per set equivalent to 20 trading days) that we used to calculate the performance and risk indicators. The average of these results are presented in the next section. We ran the expert weighting algorithm using its three versions and the following values of the parameter C: 0, 0.05, 0.5, 1, 5, 10, 20, 30, 40, and 50. When C is 0 is equivalent to apply bagging to Logitboost (bagged boosting) or to average all the experts. The threshold to eliminate very weak expert weights (γ_0) is set to 0.20, and the threshold to restrict trading (γ_1) is set to 0.

We tested our results with transaction costs of \$0, \$0.001, \$0.002, and \$0.003 per stock. These values are realistic if we consider that ISLAND has the policy to pay a rebate of \$0.002 per stock to the trader whose order was in the order books, and charges a rebate of \$0.003 per stock to the trader that submitted the incoming order. Traders can use this policy in their favour and do not have to pay the full fee of \$0.003. On the contrary, they can capture the rebate using only limit orders as the market maker strategy suggests. Even more, large brokerage firms have much lower direct transaction costs and the initial investment to trade becomes a sink cost that does not have a major impact in individual trades.

We compare our results with a buy and hold (B&H) position, a constant rebalanced portfolio (CRP), and Cover's universal portfolio (CUP) algorithm. CRP and CUP are implemented with uniform distributions, and with 10 different portfolios for the case of CUP[9].

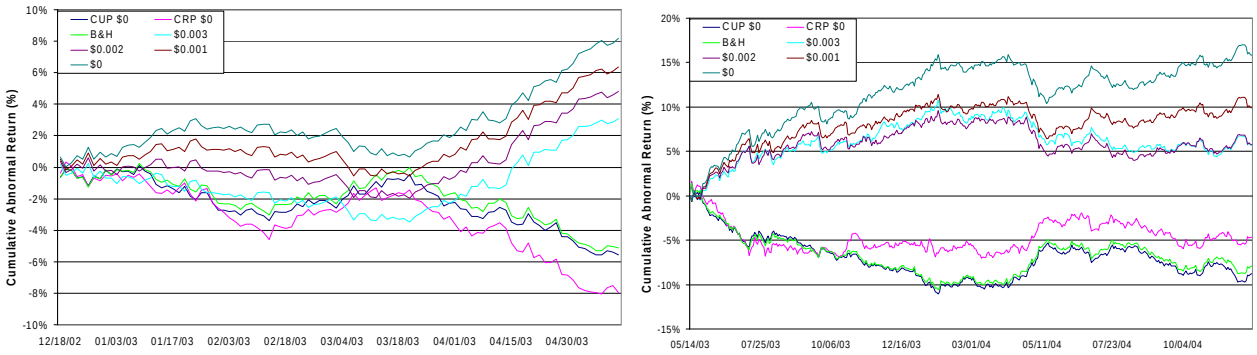
5 Results

Our results are very similar when the different values of C are used, in exception of cases when $C > 10$ where results often deteriorate. Hence, we present only the cases where $C = 1$ and $C = 0$ (bagged boosting).

Figure 6 shows the annualized in-sample and out-of-sample CAR for all stocks by transaction costs for the "Base" version and with $C = 1$. These figures indicate that all the alternatives performed better than CRP, CUP, or B&H. The validation or in-sample set was not profitable during the first two thirds of the trading period when transaction costs were \$0.002 and \$0.003. All of the alternatives of our trading algorithm, recovered in the last period with the introduction of a second expert, and also when the algorithm has accumulated some additional information. At the end of this period, all of the alternatives of our trading algorithm show positive CAR while B&H, CRP, and CUP have negative CAR (see Table 1).

Figure 6 also shows that all the out-of-sample alternatives of the trading algorithm have positive CAR while

[9] CUP and CRP are implemented following Cover [21] using the jTrade software developed in Java by Matthew Johnson, William Beaver, and German Creamer



(a) In-sample

(b) Out-of-sample

Figure 6. Cumulative abnormal return (CAR) by transaction costs with $C = 1$. CAR is calculated as weighted sum of profits of portfolio. CUP \$0 and CRP \$0 are Cover's universal portfolio algorithm and constant rebalanced portfolio respectively without transaction costs. B&H is buy and hold. \$0.003, \$0.002, \$0.001 and \$0 are transaction costs with coefficient C=1.

B&H, CRP, and CUP have negative CAR. CRP does better than B&H, while CUP does slightly worse than B&H. All our models show improvements until about half of the trading period (April 2004). Then, there is a decline and finally a smoother recovery. An explanation for these results is that the trading system was able to choose an adequate combination of experts that was efficient during a certain period of time. These experts became less efficient, and the incorporation of new experts and probably, new market conditions, led to the final improvement of the results. The above figures and Tables 1, and 2 indicate that an increase of \$0.001 in transaction costs when transaction costs are \$0 and \$0.001 represents about 3-6 percent points of differences in total and average CAR over the whole trading period. These results are even more evident with Sharpe ratios (see Table 3). The immediate explanation for these results is the high number of transactions required by the expert weighting algorithm. In this respect, one of the roles played by the risk management and optimization layer is to reduce the number of transactions to only those that seem to be profitable.

The Sterling ratio [10] in Table 4 shows sharper differences among the different versions of the model. While the Sharpe ratio is useful to compare the performance of each alternative over the complete trading period, the Sterling ratio, and specifically the maximum drawdown, seem to be more useful to capture high variations of risk-adjusted return.

After transaction costs, averaging all the experts or the bagged boosting algorithm is slightly more profitable than our "Base" algorithm in the out-of-sample set. Figure 7 shows that both algorithms have very similar patterns in the out-of-sample set, while they have identical performance in the in-sample set. The fact that bagged boosting is slightly better or about the same than our "Base" algorithm indicates that older experts or experts that were not profitable in certain moment, may become profitable in the future. As Creamer and Freud [22] have demonstrated in a previous application of bagging and boosting to risk analysis, bagging in top of boosting may reduce boosting variance improving risk adjusted indicators such as the Sharpe or the Sterling ratio.

6 Final comments and conclusions

The trading system introduced in this paper generated positive abnormal returns for a large group of stocks. This trading system was able to obtain these results based on a machine learning algorithm that makes the prediction, a weighting algorithm that combines the experts, and a risk management layer that selects only the strongest prediction and avoids trading when there is a history of negative performance. Every component of the trading system is important to obtain positive abnormal returns, and brings some functionality that is completed by the rest of the layers. We find that boosting requires powerful control mechanisms in order to reduce unnecessary and unprofitable trades that increase transaction costs. Hence, the contribution of new predictive algorithms by the computer science or machine learning literature to finance still needs to be incorporated under a formal framework of risk management.

As part of the optimization of the trading system, we propose a method to simultaneously calculate the same features with different parameters, leaving the final feature selection to boosting. Many trader systems become very inefficient because they try all the parameters or are forced to select in advance parameters that are not adequate after a trading period. Our experiments show that the boosting approach is able to improve the predictive capacity when indicators are combined and aggregated as a single pre-

[10] The in-sample Sterling ratio table is not included because an important part of its results are voided in order to calculate the maximum drawdown.

	In-sample				Out-of-sample			
Model	$0.003	$0.002	$0.001	$0	$0.003	$0.002	$0.001	$0
Avg. weight	8.05%	12.42%	16.62%	21.74%	3.86%	3.84%	6.35%	9.93%
Base	7.92%	12.46%	16.65%	21.73%	3.52%	3.53%	6.09%	9.64%
Simple	8.06%	12.43%	16.67%	21.74%	3.56%	3.44%	6.09%	9.96%
No adj.	7.93%	12.04%	16.77%	21.26%	3.61%	4.20%	6.19%	10.73%
CUP	-14.91%	-14.54%	-11.49%	-13.31%	-5.15%	-5.56%	-4.43%	-5.60%
CRP	-11.94%	-19.46%	-22.25%	-18.69%	-4.33%	-4.06%	-1.29%	-2.95%
B&H	-13.01%	-12.78%	-12.55%	-12.32%	-5.02%	-5.02%	-5.02%	-5.02%

Table 1. Annualized cumulative abnormal return (%) by transaction costs with $C = 1$. CAR is calculated as weighted sum of profits of portfolio. CUP is Cover's universal portfolio, CRP is constant rebalanced portfolio, and B&H is buy and hold.

	In-sample				Out-of-sample			
Model	$0.003	$0.002	$0.001	$0	$0.003	$0.002	$0.001	$0
Avg. weight	12.02%	16.03%	20.17%	26.63%	5.50%	5.47%	8.15%	12.74%
Base	11.82%	16.09%	20.20%	26.61%	5.16%	5.10%	7.88%	12.40%
Simple	12.03%	16.04%	20.23%	26.63%	5.19%	5.01%	7.84%	12.79%
No adj.	11.85%	15.49%	20.39%	25.93%	5.41%	5.99%	7.89%	13.49%
CUP	-12.55%	-12.43%	-9.34%	-11.07%	0.77%	0.06%	0.25%	-0.41%
CRP	-9.98%	-16.96%	-17.99%	-15.59%	0.51%	3.64%	9.55%	5.18%
B&H	-10.82%	-10.59%	-10.37%	-10.14%	-3.53%	-3.53%	-3.53%	-3.53%

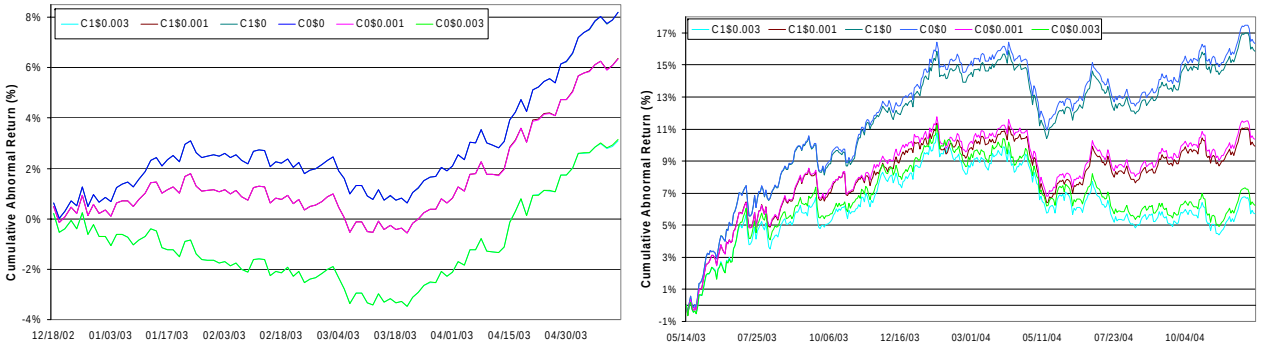
Table 2. Annualized average abnormal return (%) by transaction costs with $C = 1$. Annualized average is calculated over the monthly in-sample (December 2002 to May 2003) and out-of-sample (May 2003 to December 2005) sets.

	In-sample				Out-of-sample			
Model	$0.003	$0.002	$0.001	$0	$0.003	$0.002	$0.001	$0
Avg. weight	0.91	1.51	2.10	2.38	0.74	0.73	1.26	1.74
Base	0.91	1.51	2.10	2.38	0.67	0.68	1.20	1.69
Simple	0.91	1.51	2.10	2.38	0.68	0.66	1.22	1.74
No adj.	0.90	1.50	2.10	2.38	0.66	0.77	1.26	1.99
CUP	-2.27	-2.36	-1.80	-2.08	-0.20	-0.30	-0.26	-0.38
CRP	-1.95	-3.01	-2.61	-2.53	-0.21	0.18	0.56	0.29
B&H	-2.05	-2.01	-1.97	-1.93	-1.01	-1.01	-1.01	-1.01

Table 3. Annualized average Sharpe ratios by transaction costs with $C = 1$. Annualized average is calculated over the monthly in-sample (December 2002 to May 2003) and out-of-sample (May 2003 to December 2005) sets.

Model	$0.003	$0.002	$0.001	$0
Avg. weight	0.59	0.49	0.62	0.74
Base	0.54	0.45	0.60	0.71
Simple	0.54	0.44	0.59	0.74
No adj.	0.42	0.50	0.58	0.80
CUP	-0.23	-0.30	-0.21	-0.28
CRP	-0.19	-0.18	-0.04	-0.14
B & H	-0.24	-0.24	-0.25	-0.26

Table 4. Annualized average out-of-sample Sterling ratios by transaction costs with $C = 1$. Annualized average is calculated over the monthly out-of-sample (May 2003 to December 2005) sets.



(a) In-sample

(b) Out-of-sample

Figure 7. Cumulative abnormal return (CAR) by transaction costs with $C = 1$ and $C = 0$ (bagged boosting). CAR is calculated as weighted sum of profits of portfolio. C1$0.003-$0 or C00.003−0 refers to coefficient C=1 or C=0 (bagged boosting) and transactions costs $0.003, $0.002, $0.001 and $0.

dictor. Even more, the combination of indicators of different stocks demonstrated to be adequate in order to reduce the use of computational resources, and still maintain an adequate predictive capacity. Finally, the experiments show that a bagged boosting version of our algorithm may improve or obtain the same risk adjusted return than our "Base" algorithm. This case shows that the combination of all the experts generated by Logitboost may play an important role to define an optimal allocation of our portfolio.

Acknowledgements

GC thanks the Center for Computational Learning Systems at Columbia University for the computational support to conduct the experiments, to William Beaver and Matthew Johnson for their support in the joint development of the universal portfolio trading software, and to David E. Shaw, Oliver Watson, Anoop Prasad, David Waltz, Sal Stolfo, Vasant Dhar, and Tony Jebara for informal discussions about the trading algorithm. The opinions presented are the exclusive responsibility of the authors.

References

[1] S. Alexander, Price movements in speculative markets: trends or random walks, *Industrial Management Review*, *2*, 1961, 7–26.

[2] P. H. Algoet and T. M. Cover, Asymptotic optimality and asymptotic equipartition properties of log-optimum investment, *Annals of Probability*, *16*, 1988, 876–898.

[3] F. Allen and R. Karjalainen, Using genetic algorithms to find technical trading rules, *Journal of Financial Economics*, *51*(2), 1999, 245–271.

[4] J. Arifovic, The behavior of the exchange rate in the genetic algorithm and experimental economies, *Journal of Political Economy*, *104*(3), 1996, 510–541.

[5] W. B. Arthur, J. H. Holland, B. LeBaron, R. Palmer, and P. Talyer, Asset Pricing Under Endogenous Expectations in an Artificial Stock Market, in W. Arthur, S. Durlauf, and D. Lane, editors, *The Economy as an Evolving Complex System II* ("Reading, MA": "Addison-Wesley", 1997), 15–44.

[6] R. Bates, M. Dempster, and Y. Romahi, Evolutionary reinforcement learning in FX order book and order flow analysis, in *Proceedings of the IEEE International Conference on Computational Intelligence for Financial Engineering, Hong Kong, March 20-23, 2003* (Hong Kong: IEEE, 2003), 355–362.

[7] A. Blum and A. Kalai, Universal portfolios with and without transaction costs, *Machine Learning*, *35*(3), 1999, 193–205, special Issue for COLT '97.

[8] T. Bollerslev, Generalized autoregressive conditional heteroscedasticity, *Journal of Econometrics*, *31*, 1986, 307–327.

[9] J. A. Bollinger, *Bollinger on Bollinger Bands* (New York: McGraw-Hill, 2001).

[10] S. Bornholdt, Expectation bubbles in a spin model of markets: Intermittency from frustration across scales, *International Journal of Modern Physics C*, *12*, 2001, 667–674.

[11] A. Borodin and R. El-Yaniv, *Online computation and competitive analysis* (Cambridge: Cambridge University Press, 1998).

[12] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth, How to use expert advice, *Journal of the ACM*, *44*(3"), 1997, 427–485.

[13] M. Chaikin, How to find big winners using persistency of money flow, MP3 Audio, NA.

[14] N. Chan, B. LeBaron, A. Lo, and T. Poggio, Agent-based models of financial markets: A comparison with experimental markets, Technical Report 124, MIT Artificial Markets Project, 1999.

[15] T. Chan, *Artificial markets and intelligent agents*, Ph.D. thesis, Massachusetts Institute of Technology, 2001.

[16] T. S. Chande and S. Kroll, *The New Technical Trader: Boost your profit by plugging into the latest indicators* (New York: John Wiley & Sons, Inc., 1994).

[17] V. Cho, B. Wuthrich, and J. Zhang, Text processing for classification, *Journal of Computational Intelligence in Finance*, *7*, March/April 1999, 6–22.

[18] J. F. Clayburg, *Four Steps to Trading Success: Using everyday indicators to achieve extraordinary profits* (New York: John Wiley & Sons, Inc., 2001).

[19] M. Collins, R. E. Schapire, and Y. Singer, Logistic regression, adaboost and Bregman distances, *Machine Learning*, *48*(1-3), 2004, 253–285.

[20] T. Cover and E. Ordentlich, Universal portfolios with side information, *IEEE Transactions on Information Theory*, *42*(2), 1996, 348–363.

[21] T. M. Cover, Universal portfolios, *Mathematical Finance*, *1*(1), 1991, 1–29.

[22] G. Creamer and Y. Freund, Predicting performance and quantifying corporate governance risk for latin american adrs and banks, in *Proceedings of the Second IASTED International Conference Financial Engineering and Applications, Cambridge, MA, USA, November 8-10, 2004* (Acta Press, 2004), 91–101.

[23] K. Decker, K. Sycara, and D. Zeng, Designing a multi-agent portfolio management system, in *Proceedings of the AAAI Workshop on Internet Information Systems* (AAI Press, 1996).

[24] M. Dempster and V. Leemans, An automated FX trading system using adaptive reinforcement learning, *Expert Systems with Applications: Special issue on financial engineering*, *30*, 2006, 534–552.

[25] M. Dempster, T. W. Payne, Y. Romahi, and G. Thompson, Computational learning techniques for intraday FX trading using popular technical indicators, *IEEE Transactions on neural networks*, *12*, 2001, 744–754.

[26] J. F. Ehlers, *Rocket Science for Traders: Digital signal processing applications* (New York: John Wiley & Sons, Inc., 2001).

[27] N. Ehrentreich, The Santa Fe artificial stock market re-examined - suggested corrections, Computational Economics 45/02, Martin Luther Universitat, 2002.

[28] R. El-Yaniv, Competitive solutions for online financial problems, *ACM Computing Surveys*, *30*(1), 1998, 28–69.

[29] R. Engle, Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation, *Econometrica*, *50*, 1982, 987–1006.

[30] E. Fama, Efficient capital markets: a review of theory and empirical work, *Journal of Finance*, *25*, 1970, 383–417.

[31] E. Fama and M. Blume, Filter rules and stock market trading. security prices: a supplement, *Journal of Business*, *39*, 1970, 226–241.

[32] T. Fawcett and F. Provost, Activity monitoring: Noticing interesting changes in behavior, in *Proceedings of the Fifth ACM SIGKDD International conference on knowledge discovery and data mining (KDD-99), San Diego, CA, USA, August 15-18, 1999* (New York: ACM, 1999), 53–62.

[33] N. Fosback, *Stock Market Logic: A sophisticated approach to profits on Wall Street* (Chicago: Dearborn Trade Publishing, 1991).

[34] Y. Freund, Y. Mansour, and R. Schapire, Generalization bounds for averaged classifiers, *The Annals of Statistics*, *32*(4), 2004, 1698–1722.

[35] Y. Freund and L. Mason, The alternating decision tree learning algorithm, in *Machine Learning: Proceedings of the Sixteenth International Conference, Bled, Slovenia, June 27 - 30, 1999* (San Francisco: Morgan Kaufmann Publishers Inc., 1999), 124–133.

[36] Y. Freund and R. E. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, *Journal of Computer and System Sciences*, *55*(1), 1997, 119–139.

[37] J. Friedman, T. Hastie, and R. Tibshirani, Additive logistic regression: A statistical view of boosting, *The Annals of Statistics*, *38*(2), 2000, 337–374.

[38] M. Garman and M. Klass, On the estimation of security price volatilities from historical data, *Journal of Business*, *53*, 1980, 67–78.

[39] D. K. Gode and S. Sunder, Allocative efficiency of markets with zero intelligence traders: Market as a partial substitute for individual rationality, *Journal of Political Economy*, *101*(1), 1993, 119–137.

[40] J. Granville, *Granville's New Key to Stock Market Profits* (Upper Saddle River: Prentice Hall, 2000).

[41] S. Grossman and J. Stiglitz, On the impossibility of informationally efficient markets, *American Economic Review*, *70*, 1980, 393–408.

[42] E. Ising, Beitrag zur theorie des ferromagnetismus, *Z Physik*, *31*, 1925, 253–258.

[43] A. Kalai and S. Vempala, Efficient algorithms for universal portfolios, *Journal of Machine Learning Research*, *3*, 2003, 423–440.

[44] J. Katz and D. McCormick, *The Encyclopedia of Trading Strategies* (New York: McGraw-Hill, 2000).

[45] J. L. Kelly, A new interpretation of information rate, *Bell System Technical Journal*, *35*, 1956, 917–926.

[46] V. Lavrenko, M. Schmill, D. Lawrie, P. Oglivie, D. Jensen, and J. Allan, Language models for financial news recommendation, in *Proceedings of the Ninth International Conference on Information and Knowledge Management, McLean, VA, November 6-11, 2000* (New York: ACM, 2000), 389–396.

[47] B. LeBaron, An evolutionary bootstrap method for selecting dynamic trading strategies, in *Refenes, A.-P. N., Burgess, A. and Moody, J. eds., Decision Technologies for Computational Finance, Proceedings of the Fifth International Conference Computational Finance* (New York: Springer-Verlag, 1998), 141–160.

[48] B. LeBaron, Agent based computational finance: Suggested readings and early research, *Journal of Economic Dynamics and Control*, *24*, 2000, 679–702.

[49] B. LeBaron, Empirical regularities from interacting long and short memory investors in an agent-based financial market, *IEEE Transactions on Evolutionary Computation*, *5*, 2001, 442–455.

[50] B. LeBaron, W. B. Arthur, and R. Palmer, The time series properties of an artificial stock market, *Journal of Economic Dynamics and Control*, *21*, 1998, 1487–1516.

[51] M. Lettau, Explaining the facts with adaptive agents: The case of mutual funds flows, *Journal of Economic Dynamics and Control*, *21*, 1997, 1117–1148.

[52] N. Littlestone and M. Warmuth, The weighted majority algorithm, *Information and Computation*, *108*, 1994, 212–261.

[53] A. Lo, H. Mamaysky, and J. Wang, Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation, *Journal of Finance*, *4*, 2000, 1705–1765.

[54] J. Moody and M. Saffell, Learning to trade via direct reinforcement, *IEEE Transactions on Neural Networks*, *12*, 2001, 875–889.

[55] M. Pring, *Technical Analysis Explained*, 4 edition (New York: McGraw-Hill, 2002).

[56] B. R. Routledge, Genetic algorithm learning to choose and use information, *Macroeconomic dynamics*, *5*, 2001, 303–325.

[57] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, Boosting the margin: A new explanation for the effectiveness of voting methods, *The Annals of Statistics*, *26*(5), 1998, 1651–1686.

[58] Y.-W. Seo, J. Giampapa, and K. Sycara, Financial news analysis for intelligent portfolio management, Technical Report CMU-RI-TR-04-04, Robotics Institute,Carnegie Mellon University, 2004.

[59] C. J. Sherry, *The New Science of Technical Analysis* (Chicago: Probus publishing, 1994).

[60] T. Stridsman, *Trading Systems and Money Management* (New York: McGraw-Hill, 2003).

[61] J. D. Thomas, *News and Trading Rules*, Ph.D. thesis, Carnegie Mellon University, 2003.

[62] N. Towers and A. N. Burgess, Implementing trading strategies for forecasting models, in *Y. S. Abu-Mostafa, B. LeBaron, A. W. Lo and A. S. Weigend (eds.) Computational finance. Proceedings of the Sixth International Conference on Computational Finance, New York, USA, January 6-9, 1999* (Cambridge, MA: The MIT Press, 2000), 313–325.

[63] R. Trippi and J. Lee, editors, *Foundations of Investment System Using Artificial Intelligence and Web* (New York: McGraw-Hill, 2000).

[64] R. Trippi and E. Turban, editors, *Investment management: Decision support and expert systems* (New York: Van Nostrand Reinhold, 1990).

[65] T. Tsay, *Analysis of Financial Time Series* (New York: Wiley, 2002).

[66] V. Vovk and C. Watkins, Universal portfolio selection, in *Proceedings of the 11th Annual Conference on Computational Learning Theory (COLT-98)* (New York: ACM Press, 1998), 12–23.

[67] B. Wuthrich, D. Permunetilleke, S. Leung, V. Cho, J. Zhang, and W. Lam, Daily prediction of major stock indices from textual www data, in *Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, New York, August 27-31, 1998* (New York: AAAI Press, 1998), 364–368.

[68] E. Zivot and J. Wang, *Modeling Financial Time Series with S-Plus* (New York: Springer, 2003).

Appendix. Investment signals for automated trading system

Technical indicators are statistics of the market that quantify market trends. Most technical indicators have been developed by professional traders using trial and error. It is common practice to use rules based on technical indicators to choose the timing of buy and sell orders. These rules are called buy and sell "signals". In this work we use a combination of market indicators and trading signals. We

define these indicators in this appendix and provide the basic intuition that motivates them. Throughout this section we assume a single fixed stock.

We start with some basic mathematical notation. We index the trading days by $t = 1, 2, \ldots$. We denote by $P_t^{\mathrm{o}}, P_t^{\mathrm{c}}, P_t^{\mathrm{uc}}, P_t^{\mathrm{h}}$, and P_t^{l}, the open, adjusted close, unadjusted close[11], high, and low price of the tth trading day. We eliminate the lower index when we wish to refer to the whole sequence, i.e. P^{c} refers to the whole sequence $P_1^{\mathrm{c}}, P_2^{\mathrm{c}}, \ldots$.

Many of the technical indicators incorporate time averages of prices or of other indicators. We use two types of time averages, the simple moving average and the exponentially weighted moving average.[12] Let $\mathbf{X}$ denote a time sequence $X_1, X_2, \ldots$. The **simple moving average** is defined as

$$\mathbf{SMA}_t(\mathbf{X}, n) = \frac{1}{n} \sum_{s=0}^{n-1} X_{t-s} \ ,$$

and the **exponentially weighted moving average** is defined as

$$\mathbf{EMA}_t(\mathbf{X}, n) = \lambda \sum_{s=0}^{\infty} (1-\lambda)^s X_{t-s}; \quad \lambda = \frac{2}{n+1} \ .$$

A useful property of $\mathbf{EMA}_t(\mathbf{X}, n)$ is that it can be calculated using a simple update rule:

$$\mathbf{EMA}_t(\mathbf{X}, n) = \lambda X_t + (1-\lambda)\mathbf{EMA}_{t-1}(\mathbf{X}, n) \ .$$

We name as "rule" and follow by an identification number the most common rules associated with each indicator. Most of the buy and sell signals are generated when the value of an indicator crosses some threshold or the value of another indicator. The input to our learning system includes both signals and indicators. We use *normalized* indicators, by which we mean indicators whose value does not change if all the prices in the sequence are multiplied by a constant factor. This is important when working with adjusted stock prices.

Additionally, we recalculate a selected group of indicators and their rules with three different values of the main parameters that are close to the industry practice. So, our learning system should be able to select the optimal combination of indicators and parameters. Additionally, we include ratios of the indicators which generally are calculated as the indicator divided by its moving average. Most of these ratios are part of the trading rules. However, we include the ratios by themselves so that our learning system finds its own rules. Finally, there are indicators that do not have a specific trading rule such as the volatility and return indicators. We include several measures of volatility, so that our model is able to discover its own rules of risk management. These volatility measures are mostly based on GARCH models that we discuss next.

GARCH

ARCH and GARCH models have been widely used in finance literature to forecast volatility and assets' return, especially since the volatility of assets return seems to be serially correlated. Engle [29] introduced the Autoregressive Conditionally Heteroskedastic (ARCH) model as a way to simulate the serial correlation of volatility. We follow Tsay [65] in describing the ARCH/GARCH models.

In essence, in ARCH models the price changes by a normal distribution with constant mean and time-varying variance. We denote by r_t log-return on day t:

$$r_t = \log(p_{t+1}/p_t)$$

where p_t is the price (at a specific time, usually the close price) on day t. At this moment, we are not considering transaction costs.

The ARCH(m) model defines a stochastic process for generating the sequence of log-returns $r_1, r_2, \ldots$. The process is defined over the *mean adjusted returns* which are defined as $a_t \doteq r_t - \mu$ where μ is the fixed *mean return*. The source of randomness is a sequence of "noise" random variables $\epsilon_1, \epsilon_2, \ldots$ which are chosen independently at random according to the distribution $N(0, 1)$. For $t = 1, 2, \ldots$ we compute the variance $\sigma_t^2(n)$ according to the formula

$$\sigma_t^2(n) = \alpha_0 + \alpha_1 a_{t-1}^2 + \ldots + \alpha_m a_{t-m}^2 \ .$$

where $\alpha_0 > 0$ and $\alpha_i \geq 0$.

Given σ_t we set $a_t = \sigma_t \epsilon_t$. To initialize the process we set $\sigma_0^2(n) = 0$.

Bollerslev [8] extended the ARCH model and proposed the Generalized Autoregressive Conditionally Heteroskedastic (GARCH) model to simulate volatility without having to calculate a large number of coefficients for polynomials of high-order. The GARCH models assume that r_t can be simulated with an autoregressive moving-average (ARMA) model. [13]

The $GARCH(m, s)$ model adds a distributed lag structure to simulate the conditional variance:

$$\sigma_t^2(n) = \alpha_0 + \sum_{i=1}^{m} \alpha_i a_{t-i}^2 + \sum_{j=1}^{s} \beta_j \sigma_{t-j}^2(n)$$

assuming that $\alpha_0 > 0$, $\alpha_i \geq 0$, $\beta_j \geq 0$, and $\sum_{i=1}^{s} \alpha_i +$

[11] Unadjusted close prices are the actual published prices at the end of the trading day. The adjusted stock price removes the effect of stock splits and dividend payments. Our goal is to predict P_t^{c}, the *adjusted* close price.

[12] We follow Zivot and Wang [68] in describing the technical analysis indicators. Additional useful references about technical analysis and trading are [44, 55, 16, 59, 60, 26, 18].

[13] An ARMA(p,q) model to calculate r_t has the following form:

$$\hat{r}_t = \phi_0 + \sum_{i=1}^{p} \phi_i r_{t-i} + a_t - \sum_{i=1}^{q} \theta_i a_{t-i}^2$$

$\sum_{i=1}^{s} \beta_i < 1$. This last condition assures that the unconditional variance of a_t is not infinite or a_t is stationary, and its conditional variance changes over time ($\sigma_t^2(n)$). We calculate the one step ahead forecast of the log return $\hat{r}_{t+1}$, the volatility $\hat{\sigma}_t^2$, and the Sharpe ratio ($\hat{r}_{t+}/\hat{\sigma}_t^2$) using the $GARCH(1,1)$ model. in the calculation of VAR using GARCH.).

In the following table we describe the technical indicators. The parameters of each indicator are in parentheses. Most of the parameters used refer to the length of the period (n) selected to calculate the indicator. In case of exponential moving average, the parameter used is λ which also depends of n. We have assigned parameters which are typically used in the industry for each indicator.

Variable	Description	Calculation detail
Price indicators:		
$EMA_t^c(\lambda)$	Exponential moving average of a time series $P^{\mathbf{c}}$.	$\mathbf{EMA}_t(P^{\mathbf{c}}, \lambda)$ $where\, \lambda = 0.9,\ 0.84,\ and\, 0.78$
$rule1_t - rule4_t$	Exponential moving average to price ($P_t^{\mathbf{c}}$, P_t^{med}, P_t^{typ}, and P_t^{wc}).	$\frac{EMA_t^c(\lambda)}{P^{\mathbf{c}}}$, $\frac{EMA_t^{med}(\lambda)}{P^{\text{med}}}$, $\frac{EMA_t^{typ}(\lambda)}{P^{\text{typ}}}$, and $\frac{EMA_t^{wc}(\lambda)}{P^{\text{wc}}}$ where n= 0.9, 0.84, and 0.78
$SMA_t^c(n)$	Simple moving average of the last n observations of a time series $P^{\mathbf{c}}$.	$\mathbf{SMA}_t(P^{\mathbf{c}}, n)$ $where$ n= $10,\ 16\ and\ 22$
$rule5_t$	Simple moving average to $P_t^{\mathbf{c}}$	$\frac{SMA_t^c(n)}{P^{\mathbf{c}}}$ where n=10, 16 and 22
Bollinger bands:	Using the moving average or the median band ($Boll_t^m(n)$) as the reference point, the upper and lower Bollinger [9] bands ($Boll_t^u(n)$ and $Boll_t^d(n)$ respectively) are calculated in function of s standard deviations. When price crosses above (below) the upper (lower) Bollinger band, it is a sign that the market is overbought (oversold). Technical analysts typically calculate Bollinger bands using 20 days for the moving average and 2 standard deviations.	$Boll_t^m(n) = SMA_t^c(n)$
$Boll_t^u(n)$	Upper Bollinger band	$Boll_t^m(n) + s\sigma_t^2(n)$ where s=2, n=20, 26 and 32
$Boll_t^d(n)$	Lower Bollinger band	$Boll_t^m(n) - s\sigma_t^2(n)$ where s=2, n=20, 26 and 32
$PBoll_t^u(n)$	Price to upper Bollinger band	$\frac{P_t^{\mathbf{c}}}{Boll_t^u(n)}$
$PBoll_t^d(n)$	Price to lower Bollinger band	$\frac{P_t^{\mathbf{c}}}{Boll_t^d(n)}$
$rule6_t$	Bollinger trading rule	$\begin{cases} Buy & \text{if } P_{t-1}^{\mathbf{c}} \geq Boll_t^d(n) \text{ and } P_t^{\mathbf{c}} < Boll_t^u(n) \\ Sell & \text{if } P_{t-1}^{\mathbf{c}} \leq Boll_t^d(n) \text{ and } P_t^{\mathbf{c}} > Boll_t^u(n) \\ Hold & \text{Otherwise} \end{cases}$
Momentum and oscillation indicators:		
$MOM_t(n)$	Momentum: price ($P_t^{\mathbf{c}}$) change in the last n periods. When it crosses above (below) zero, it indicates that trend is up (down). The default value of n is 12.	$P_t^{\mathbf{c}} - P_{t-n}^{\mathbf{c}}$ $where\, n = 12,\ 18,\ and\, 24$
$MomEMA_t(n, \lambda)$	Momentum to $\mathbf{EMA}_t(MOM_t(n), \lambda)$	$\frac{MOM_t(n)}{\mathbf{EMA}_t(MOM_t(n)\lambda)}$ where n=12, 18, and 24 and $\lambda = 0.75$
$rule7_t$	Momentum trading rule	$\begin{cases} Buy & \text{if } MOM_{t-1}(n) \leq \mathbf{EMA}_t(MOM_t(n), \lambda) \\ & \text{and } MOM_t(n) > \mathbf{EMA}_t(MOM_t(n), \lambda) \\ Sell & \text{if } MOM_{t-1}(n) \geq \mathbf{EMA}_t(MOM_t(n), \lambda) \\ & \text{and } MOM_t(n) < \mathbf{EMA}_t(MOM_t(n), \lambda) \\ Hold & \text{Otherwise} \end{cases}$
$ACCEL_t(n)$	Acceleration: difference of price change. The default value of n is 12.	$MOM_t(n) - MOM_{t-1}(n)$ $where\, n = 12,\ 18,\ and\, 24$
$rule8_t$	Acceleration trading rule	$\begin{cases} Buy & \text{if } ACCEL_{t-1}(n) + 1 \leq 0 \\ & \text{and } ACCEL_t(n) + 1 > 0 \\ Sell & \text{if } ACCEL_{t-1}(n) + 1 \geq 0 \\ & \text{and } ACCEL_t(n) + 1 < 0 \\ Hold & \text{Otherwise} \end{cases}$
$ROC_t(n)$	Rate of change: rate of change of $P_t^{\mathbf{c}}$. Technical analysts recommend using 10 periods to calculate this indicator.	$\frac{P_t^{\mathbf{c}} - P_{t-n}^{\mathbf{c}}}{P_{t-n}^{\mathbf{c}}} \cdot 100$ $where\, n = 10,\ 16,\ and\, 22$
$rule9_t$	ROC trading rule	$\begin{cases} Buy & \text{if } ROC_{t-1}(n) \leq 0 \text{ and } ROC_t(n) > 0 \\ Sell & \text{if } ROC_{t-1}(n) \geq 0 \text{ and } ROC_t(n) < 0 \\ Hold & \text{Otherwise} \end{cases}$

$MACD_t(s,f)$	Moving average convergence divergence: difference between two moving averages of slow and fast periods (s,f). $MACD_t(s,f)$ is regularly calculated using 26 (s) and 12 (f) periods.	$\mathbf{EMA}_t(P^{\mathrm{c}},s) - \mathbf{EMA}_t(P^{\mathrm{c}},f)$ $where\ f = 12,\ and\ s = 18,\ 24,\ and\ 30$
$MACDS_t(s,f,n)$	MACD signal line: moving average of $MACD_t(s,f)$ of past n periods. A buy (sell) signal is generated when the $MACD_t(s,f)$ crosses above (below) the signal line or a threshold.	$\mathbf{EMA}_t(MACD_t(s,f),n)$ $where\ f = 12,\ n = 9,\ ands = 18,\ 24,\ and\ 30$
$rule10_t$	MACD trading rule	$\begin{cases} Buy & \text{if } MACD_{t-1}(s,f) \le MACDS_t(s,f,n) \\ & \text{and } MACD_t(s,f) > MACDS_t(s,f,n) \\ Sell & \text{if } MACD_{t-1}(s,f) \ge MACDS_t(s,f,n) \\ & \text{and } MACD_t(s,f) < MACDS_t(s,f,n) \\ Hold & \text{Otherwise} \end{cases}$
$MACDR_t(s,f,n)$	$MACD_t(s,f)$ to $MACDS_t(s,f,n)$	$\frac{MACD_t(s,f)}{MACDS_t(s,f,n)}$
$RSI_t(n)$	Relative strength index: compares the days that stock prices finish up against those periods that stock prices finish down. Technical analysts calculate this indicator using 9, 14 or 25 periods. A buy signal is when $RSI_t(n)$ crosses below a lower band of 30 (oversold), and a sell signal when $RSI_t(n)$ crosses above an upper band of 70 (overbought)	$100 - \frac{100}{1 + \frac{\mathbf{SMA}_t(\mathbf{P^{up}_n}, n_1)}{\mathbf{SMA}_t(\mathbf{P^{dn}_n}, n_1)}}$ $where\ n_1 = 8,\ 14,\ and\ 20$ and n is the length of the time series $P^{\mathrm{up}}_t = \begin{cases} P^{\mathrm{c}}_t & \text{if } P^{\mathrm{c}}_t > P^{\mathrm{c}}_{t-1} \\ \text{empty} & Otherwise \end{cases}$ $P^{\mathrm{dn}}_t = \begin{cases} P^{\mathrm{c}}_t & \text{if } P^{\mathrm{c}}_t < P^{\mathrm{c}}_{t-1} \\ \text{empty} & Otherwise \end{cases}$ $\mathbf{P^{up}_n} = (P^{\mathrm{up}}_{t-n},\ P^{\mathrm{up}}_{t-n+1},\ P^{\mathrm{up}}_{t-n+2},\ \ldots,\ P^{\mathrm{up}}_t)$ $\mathbf{P^{dn}_n} = (P^{\mathrm{dn}}_{t-n},\ P^{\mathrm{dn}}_{t-n+1},\ P^{\mathrm{dn}}_{t-n+2},\ \ldots,\ P^{\mathrm{dn}}_t)$
$rule11_t$	RSI trading rule	$\begin{cases} Buy & \text{if } RSI_{t-1}(n) \ge 30 \text{ and } RSI_t(n) < 70 \\ Sell & \text{if } RSI_{t-1}(n) \le 30 \text{ and } RSI_t(n) > 70 \\ Hold & \text{Otherwise} \end{cases}$
Stochastic oscillator:	Compares close price to a price range in a given period to establish if market is moving to higher or lower levels or is just in the middle. The oscillator indicators are:	
$FAST\%K_t(n)$	Percent measure of the last close price in relation to the highest high and lowest low of the last n periods (true range).	$\frac{P^{\mathrm{uc}}_t - min(\mathbf{P^l_n})}{max(\mathbf{P^h_n}) - min(\mathbf{P^l_n})}$ $where\ n = 12,\ 18,\ and\ 24$
	Vector with low prices of last n periods	$\mathbf{P^l_n} = (P^{\mathrm{l}}_{t-n},\ P^{\mathrm{l}}_{t-n+1},\ P^{\mathrm{l}}_{t-n+2},\ \ldots,\ P^{\mathrm{l}}_t)$
	Vector with high prices of last n periods	$\mathbf{P^h_n} = (P^{\mathrm{h}}_{t-n},\ P^{\mathrm{h}}_{t-n+1},\ P^{\mathrm{h}}_{t-n+2},\ \ldots,\ P^{\mathrm{h}}_t)$
$FAST\%D_t(n)$	Moving average of $FAST\%K_t(n)$.	$\mathbf{SMA}_t(FAST\%K_t(n),3)$
$SLOW\%K_t(n)$	Identically calculated to $FAST\%D_t(n)$ using a 3-period moving average of $FAST\%K_t(n)$.	$\mathbf{SMA}_t(FAST\%K_t(n),3)$
$SLOW\%D_t(n)$	Moving average of $SLOW\%K_t(n)$. Typically a period of 3 is used.	$\mathbf{SMA}_t(SLOW\%K_t(n),3)$
$rule12_t$	Fast stochastic trading rule	$\begin{cases} Buy & \text{if } FAST\%K_{t-1}(n) \le FAST\%D_t(n) \\ & \text{and } FAST\%K_t(n) > FAST\%D_t(n) \\ Sell & \text{if } FAST\%K_{t-1}(n) \ge FAST\%D_t(n) \\ & \text{and } FAST\%K_t(n) < FAST\%D_t(n) \\ Hold & \text{Otherwise} \end{cases}$
$rule13_t$	Slow stochastic trading rule	$\begin{cases} Buy & \text{if } SLOW\%K_{t-1}(n) \le SLOW\%D_t(3) \\ & \text{and } SLOW\%K_t(n) > SLOW\%D_t(3) \\ Sell & \text{if } SLOW\%K_{t-1}(n) \ge SLOW\%D_t(3) \\ & \text{and } SLOW\%K_t(n) < SLOW\%D_t(3) \\ Hold & \text{Otherwise} \end{cases}$
$slowKslowD_t(n)$	$SLOW\%K_t(n)$ to $SLOW\%D_t(3)$	$\frac{SLOW\%K_t(n)}{SLOW\%D_t(3)}$
$fastKfastD_t(n)$	$FAST\%K_t(n)$ to $FAST\%D_t(n)$	$\frac{FAST\%K_t(n)}{FAST\%D_t(n)}$

$WILL_t(n)$	Williams indicator: the calculation is similar to the stochastic oscillator with a scale from 0 to -100. It tries to capture moments when the market is overbought (0 - -20) or oversold (-80 - -100).	$\frac{max(\mathbf{P^h_n})-P^{\mathrm{uc}}_t}{max(\mathbf{P^h_n})-min(\mathbf{P^l_n})}(-100)$ $where\ n = 14$
$rule14_t$	Williams trading rule	$\begin{cases} Buy & \text{if } WILL_{t-1}(n) \geq -20 \text{ and } WILL_t(n) < -80 \\ Sell & \text{if } WILL_{t-1}(n) \leq -20 \text{ and } WILL_t(n) > -80 \\ Hold & \text{Otherwise} \end{cases}$
$MFI_t(n)$	Money flow index: measures the strength of money flow (MF_t) in and out of a stock. At difference of the $RSI_t(n)$ which is calculated using stock prices, $MFI_t(n)$ is calculated using volume.	$100 - \frac{100}{1+\frac{PMF_t(n)}{NMF_t(n)}}$ where: $n = 14$ $MF_t = P^{\mathrm{typ}}_t \cdot VOL_t$ $PMF_t(n) = \mathbf{SMA}_t(MF_t, n)$ when $MF_t > 0$ $NMF_t(n) = \mathbf{SMA}_t(MF_t, n)$ when $MF_t < 0$ VOL_t is volume of day t $PMF_t(n)$ is positive money flow $NMF_t(n)$ is negative money flow
$rule15_t$	Money flow index trading rule. When $MFI_t(n)$ crosses above (below) 70 (30), this is a sign that the market is overbought (oversold).	$\begin{cases} Buy & \text{if } MFI_{t-1}(n) \geq 30 \text{ and } MFI_t(n) < 70 \\ Sell & \text{if } MFI_{t-1}(n) \leq 30 \text{ and } MFI_t(n) > 70 \\ Hold & \text{Otherwise} \end{cases}$
Volatility and return indicators: [14]		
$CHV_t(n, n_1)$	Chaikin volatility: evaluates the widening of the range between high and low prices. This indicator also calculates the rate of change of the moving average of the difference between high and low prices. Chaikin [13] suggests using 10 periods (n_1) to calculate this indicator. Chaikin also considers that a very fast increase (decrease) of the Chaikin volatility is a signal that the bottom (top) of the market is near.	$\mathbf{EMA}_t(P^{\mathrm{h}} - P^{\mathrm{l}}, n)/\mathbf{EMA}_{t-n_1}(P^{\mathrm{h}} - P^{\mathrm{l}}, n) - 1$ $wheren = n_1 = 10$
GK_t	Garman-Klass volatility: this is an extreme-value indicator proposed by Garman and Klass [38] that takes into account intraday price range to calculate the variation of the stock price. According to Garman and Klass variance is minimized when $\alpha = 0.12$. f is the fraction of the day that trading is closed. We use a value suggested by Rmetrics (0.19) which is about 4.5 hours.	$\alpha\frac{(P^{\mathrm{o}}_t - P^{\mathrm{uc}}_{t-1})}{f} + (1 - \alpha)\frac{\hat{\sigma^2_t}(n)}{1-f}$ $where\ f < 1$ $u = (P^{\mathrm{h}}_t - P^{\mathrm{o}}_t)$ $d = (P^{\mathrm{l}}_t - P^{\mathrm{o}}_t)$ $\hat{\sigma^2_t}(n) = 0.511(u-d)^2 - 0.019P^{\mathrm{c}}_t \cdot (u+d) - 2ud - 0.383(P^{\mathrm{uc}}_t)^2$
$\hat{r}_{t+1}, \sigma^2_t$	Next period return and volatility calculated using the $GARCH(1,1)$ model.	
Sharpe ratio	Risk adjusted return.	$\hat{r}_t/\sigma^2_t$
Volume indicators:		
OBV_t	On balance volume: this indicator was developed by Granville [40] to evaluate the impact of positive and negative volume flows. OBV_t adds the volume when the close price has increased and substracts it when the close price has decreased. A sign of market reversal is when OBV_t diverges with the price movement.	if $P^{\mathrm{c}}_t > P^{\mathrm{c}}_{t-1}$ $\quad OBV_t = OBV_{t-1} + VOL_t$ if $P^{\mathrm{c}}_t < P^{\mathrm{c}}_{t-1}$ $\quad OBV_t = OBV_{t-1} - VOL_t$

ADL_t	Accumulation/distribution line: this indicator was developed by Chaikin [13] to evaluate the effect of accumulative flow of money of a particular security. ADL_t is calculated using the close location value (CLV_t). This indicator compares the unadjusted close price with the range of prices for the same period without comparing with the previous period as the OBV_t does. ADL_t range is from -1 to +1, and zero is the central point. A positive value indicates buying pressure and a negative value indicates selling pressure. If there is an important positive (negative) divergence between the accumulation distribution line and the price,we have a bullish (bearish) signal.	$\sum_{t=1}^{n} CLV_t \cdot VOL_t$ $where\, CLV_t = \frac{2P_t^{\mathrm{uc}} - P_t^{\mathrm{l}} - P_t^{\mathrm{h}}}{P_t^{\mathrm{h}} - P_t^{\mathrm{l}}}$ and n refers to the length of the time series.
CHO_t	Chaikin oscillator: this indicator was also developed by Marc Chaikin. The Chaikin oscillator is the $MACD$ of the ADL. This oscillator is the difference between a short and a long $\mathbf{EMA}_t(ADL, n)$ of the ADL. The interpretation of this indicator is similar to the $MACD$.	$\mathbf{EMA}_t(ADL, n_1) - \mathbf{EMA}_t(ADL, n_2)$ $where\, n_1 = 3,\ n_2 = 10$
$rule16_t$	Chaikin volatility trading rule	sign (CHO_t)
NVI_t and PVI_t	Negative and positive volume index: these indicators were introduced by Fosback [33] as signals of bull markets. NVI_t (PVI_t) concentrates on days when volume decreases (increases). The rationality is that "informed" investors take positions on days when volume decreases, while the "uninformed" investors take position on days when the volume increases. NVI_t (PVI_t) is calculated as the cumulative sum of $ROC_t(n)$ when volume decreases (increases). Fosback maintains that there is 95% probability that a bull market is going to develop when NVI_t crosses above its one year moving average, and 67% probability of a bear market when PVI_t crosses below its one year moving average.	if $VOL_t < VOL_{t-1}$ $\quad NVI_t = NVI_{t-1} + ROC_t(n)NVI_{t-1}$ $PVI_t = PVI_{t-1}$ if $VOL_t \geq VOL_{t-1}$ $\quad PVI_t = PVI_{t-1} + ROC_t(n)PVI_{t-1}$ $NVI_t = NVI_{t-1}$ $where$ n=1
$rule17_t$	Negative volume index trading rule.	$\begin{cases} Buy & \text{if } NVI_{t-1} \leq \mathbf{SMA}_t(NVI, l) \\ & \text{and } NVI_t > \mathbf{SMA}_t(NVI, l) \\ Hold & \text{Otherwise} \end{cases}$
$rule18_t$	Positive volume index trading rule.	$\begin{cases} Buy & \text{if } PVI_{t-1} \leq \mathbf{SMA}_t(PVI, l) \\ & \text{and } PVI_t > \mathbf{SMA}_t(PVI, l) \\ Sell & \text{if } PVI_{t-1} \geq \mathbf{SMA}_t(PVI, l) \\ & \text{and } PVI_t < \mathbf{SMA}_t(PVI, l) \\ Hold & \text{Otherwise} \end{cases}$ where $l = 10$
$nviSMA_t(l)$	NVI_t to $\mathbf{SMA}_t(NVI, l)$	$\frac{NVI_t}{\mathbf{SMA}_t(NVI, l)}$, where $l = 10$
$pviSMA_t(l)$	PVI_t to $\mathbf{SMA}_t(PVI, l)$	$\frac{PVI_t}{\mathbf{SMA}_t(PVI, l)}$, where $l = 10$
$PV_t(n)$	Price-volume trend: this indicator is similar to OBV_t. It calculates a cumulative total of volume where the portion of volume added/substracted is given by the increase or decrease of close prices in relation to the previous period.	$\sum_{t=1}^{n} VOL_t \cdot ROC_t(n_1)$ $where\, n_1 = 1$ and n is the length of the time series.

[14] GARCH is also another indicator of volatility.