Neuro-Evolutionary Modeling of the Milan Stock Exchange for Intraday Trading

Antonia Azzini[1], Mauro Dragoni[2], and Andrea G. B. Tettamanzi[1]

[1] Università degli Studi di Milano
Dipartimento di Tecnologie dell'Informazione
via Bramante, 65 - 26013 Crema (CR) Italy
{antonia.azzini,andrea.tettamanzi}@unimi.it
[2] Fondazione Bruno Kessler (FBK-IRST)
Via Sommarive 18, Povo (Trento), Italy
dragoni@fbk.eu

Abstract. We investigate the correlations among the intraday prices of the major stocks of the Milan Stock Exchange by means of a neuro-evolutionary modeling method. In particular, the method used to approach such problem is to apply a very powerful natural computing analysis tool, namely evolutionary neural networks, based on the joint evolution of the topology and the connection weights together with a novel similarity-based crossover, to the analysis of a financial intraday time series expressing the stock quote variations of the FTSE MIB components. We show that it is possible to obtain extremely accurate models of the variations of the price of one stock based on the price variations of the other components of the stock list, which may be used for statistical arbitrage. The approach has been also validated by implementing a simple trading system that performs decisions by considering the results computed by the algorithm.

Keywords: Evolutionary Algorithms, Neural Networks, Intraday Trading, Statistical Arbitrage

1 Introduction

Many successful applications of natural computing techniques to the modeling and prediction of financial time series have been reported in the literature [5–7].

To our knowledge, the vast majority of such approaches consider only the time series of daily closing prices only, or at most the opening and closing prices of each security, without considering what happens during the real-time market, i.e., the intraday prices.

However, recently, different authors [4, 9, 10] explored the direction of analyzing the behavior of the intraday stock prices in order to discover correlations between the behaviors of different stock prices.

In particular, Bi's work [4], which takes into account the Chinese stock market, highlights that it is possible to infer serial correlations between components,

while the work presented in [10] discusses the application of data mining techniques to the detection of stock price manipulations by incorporating the analysis of intraday trade prices, in addition to closing prices, for the investigation of intraday variations.

Another interesting direction early explored is related to the study of the impact of financial news on the intraday prices. In this sense, Mittermayer [14] describes a system implemented to predict stock price trends based on the publication of press releases. Its system mainly consists of three components: a crawler of relevant information from press releases, a press release categorizer, and a reasoner that is able to derive appropriate trading strategies. The results that he obtained is that, with an effective categorization of press releases and with an adequate trading strategy, additional information, useful to forecast stock price trends, can be easily provided. Two alternatives have been discussed also in [8] and [11]. In the former, the author focuses on the use of rough set theory for transforming the unstructured information into structured data; while in the latter, the author applies four different machine learning techniques in order to detect patterns in the textual data that could explain increased risk exposure of some stocks with respect to financial press releases.

The idea proposed in this work follows the first direction. We claim that the analysis of intraday variations could lead to the discovery of patterns that permit to improve the accuracy of the predictions of stock variations. In particular, in this work, we perform a preliminary investigation focusing on the components of the FTSE MIB index of the Milan Stock Exchange. We use an evolutionary algorithm to jointly evolve the structure and connection weights of feed-forward multilayer neural networks that predict the intraday price variations of one component of the index based on the price variations of the other components recorded at the same time. The resulting neural networks may be regarded as non-linear factor models of one security with respect to the others.

The rest of the paper is organized as follows: Section 2 presents the problem and the dataset, while a brief description of the evolutionary approach considered in this work is reported in Section 3. The experiments carried out by applying the evolutionary algorithm are presented in Section 4, together with a discussion of the results obtained. Finally, Section 5 provides some concluding remarks.

2 Problem Description

The object of our investigation may be formulated as a modeling problem, whereby we are seeking for a non-linear factor model that expresses the returns of one security as a function of the returns of a set of other securities at the same instant [12].

In particular, we use feed-forward multilayer neural networks to represent such non-linear factor models and we exploit a well-tested neuro-evolutionary algorithm [3] to optimize both the structure (number of hidden layers, number of neurons in each hidden layer) and the connection weights of the neural networks.

Factor models are used primarily for statistical arbitrage. A statistical arbitrageur builds a hedged portfolio consisting of one or more long positions and one or more short positions in various correlated instruments. When the price of one of the instruments diverges from the value predicted by the model, the arbitrageur puts on the arbitrage, by going long that instrument and short the others, if the price is lower than predicted, or short that instrument and long the others, if the price is higher. If the model is correct, the price will tend to revert to the value predicted by the model, and the arbitrageur will profit.

The simplest case of statistical arbitrage is pair trading, whereby the financial instruments considered for constructing the hedged portfolio are just two. The non-linear factor models that we obtain with our approach may be used for the more general case where the opportunity set is constituted by all the components of the index. However, the same models might also be used for pair trading of each individual stock against the index future, which is generally regarded as an accurate proxy of the index itself, at least at the time scales involved in intraday trading.

3 The Neuro Genetic Algorithm

The neuro-evolutionary algorithm that we use is based on the evolution of a population of individuals, each representing a feed-forward multilayer neural network, also known as a multilayer perceptron (MLP), through the joint optimization of their structures and weights, here briefly summarized; a more complete and detailed description can be found in the literature [3]. In this work, the algorithm uses the Scaled Conjugate Gradient method (SCG) [13] instead of the more traditional error back-propagation (BP) algorithm to decode a *genotype* into a *phenotype* NN, in order to speed up the convergence of such a conventional training algorithm. Accordingly, it is the genotype which undergoes the genetic operators and which reproduces itself, whereas the phenotype is used *only* for calculating the genotype's fitness. The rationale for this choice is that the alternative of applying SCG to the genotype as a kind of 'intelligent' mutation operator, would boost exploitation while impairing exploration, thus making the algorithm too prone to getting trapped in local optima.

The population is initialized with different hidden layer sizes and different numbers of neurons for each individual according to two exponential distributions, in order to maintain diversity among all of them in the new population. Such dimensions are not bounded in advance, even though the fitness function may penalize large networks. The number of neurons in each hidden layer is constrained to be greater than or equal to the number of network outputs, in order to avoid hourglass structures, whose performance tends to be poor. Indeed, a layer with fewer neurons than the outputs destroys information which later cannot be recovered.

3.1 Evolutionary Process

The initial population is randomly created and the genetic operators are then applied to each network until the termination conditions are not satisfied.

At each generation, the first half of the population corresponds to the best $\lfloor n/2 \rfloor$ individuals selected by truncation from a population of size n, while the second half of the population is replaced by the offsprings generated through the crossover operator. Crossover is then applied to two individuals selected from the best half of the population (parents), with a probability parameter p_{cross}, defined by the user together with all the other genetic parameters, and maintained unchanged during the entire evolutionary process.

It is worth noting that the p_{cross} parameter refers to a 'desired' crossover probability, set at the beginning of the evolutionary process. However, the 'actual' probability during a run will usually be lower, because the application of the crossover operator is subject to the condition of similarity between the parents.

Elitism allows the survival of the best individual unchanged into the next generation and the solutions to get better over time. Then, the algorithm mutates the weights and the topology of the offsprings, trains the resulting network, calculates fitness over the validation set, and finally saves the best individual and statistics about the entire evolutionary process.

The application of the genetic operators to each network is described by the following pseudo-code:

1. Select from the population (of size n) $\lfloor n/2 \rfloor$ individuals by truncation and create a new population of size n with copies of the selected individuals.
2. For all individuals in the population:
   (a) Randomly choose two individuals as possible parents.
   (b) Check their local similarity and apply crossover according to the crossover probability.
   (c) Mutate the weights and the topology of the offspring according to the mutation probabilities.
   (d) Train the resulting network using the training set.
   (e) Calculate the fitness f over the validation set.
   (f) Save the individual with lowest f as the best-so-far individual if the f of the previously saved best-so-far individual is higher (worse).
3. Save statistics.

The `SimBa` crossover starts by looking for a 'local similarity' between two individuals selected from the population. If such a condition is satisfied the layers involved in the crossover operator are defined. The contribution of each neuron of the layer selected for the crossover is computed, and the neurons of each layer are reordered according to their contribution. Then, each neuron of the layer in the first selected individual is associated with the most 'similar' neuron of the layer in the other individual, and the neurons of the layer of the second individual are re-ranked by considering the associations with the neurons of the first one. Finally a cut-point is randomly selected and the neurons above the cut-point are swapped by generating the offspring of the selected individuals.

Weights mutation perturbs the weights of the neurons before performing any structural mutation and applying SCG to train the network. All the weights and the corresponding biases are updated by using variance matrices and evolutionary strategies applied to the synapses of each NN, in order to allow a control parameter, like mutation variance, to self-adapt rather than changing their values by some deterministic algorithms. Finally, the topology mutation is implemented with four types of mutation by considering neurons and layer addition and elimination. The addition and the elimination of a layer and the insertion of a neuron are applied with three independent probabilities, indicated as p^{+}_{layer}, p^{-}_{layer} and p^{+}_{neuron}, while the elimination of a neuron is carried out only if the contribution of that neuron is negligible with respect to the overall network output.

For each generation of the population, all the information of the best individual is saved.

As previously considered [1, 2], the evolutionary process adopts the convention that a lower fitness means a better NN, mapping the objective function into an error minimization problem. Therefore, the fitness used for evaluating each individual in the population is proportional to the mean square error (mse) and to the computational cost of the considered network. This latter term induces a selective pressure favoring individuals with reduced-dimension topologies.

The fitness function is calculated, after the training and the evaluation processes, by the Equation 1

$$f = \lambda kc + (1 - \lambda) * mse, \tag{1}$$

where λ corresponds to the desired tradeoff between network cost and accuracy, and it has been set experimentally to 0.2 to place more emphasis on accuracy, since the NN cost increase is also checked by the entire evolutionary algorithm. k is a scaling constant set experimentally to 10^{-6}, and c models the computational cost of a neural network, proportional to the number of hidden neurons and synapses of the neural network.

Following the commonly accepted practice of machine learning, the problem data is partitioned into training, validation and test sets, used, respectively for network training, to stop learning avoiding overfitting, and to test the generalization capabilities of a network. The fitness is calculated over the validation set.

4 Experiments and Results

In this section, we present the intraday dataset that has been created for performing our experiments and the discussion about the results that we have obtained.

4.1 Dataset and Experiment Set-Up

We created the dataset starting from raw data representing the intraday stock quotes of the FTSE MIB components in the period beginning on August the

1st, 2011 and ending on November the 20th, 2011, observed every 5 minutes. For each observation, we have computed the price variation (technically, a log-return) between the quote of the observation at instant t and the quote of the previous observation, at instant $t-1$,

$$r(t) = \log \frac{x(t)}{x(t-1)}. \tag{2}$$

We discarded the observations for which t was the first observation of each day (i.e., the opening price). Therefore, knowing that the trading hours of the Milan Stock Exchange are from 9:00 am to 5.30 pm, we did not consider for our dataset any of the observations available whose time label is 9:00 am. The rationale behind this choice is that while markets are closed, perturbing events (economics, politics, etc.) that may strongly alter the opening quote of a stock may occur and, in that case, the variation between the first observation of the current day and the last observation of the day before could be orders of magnitude larger than the other stock quotes variations. As a matter of fact, that particular price variation is anyway of little or no interest to a day trader, who will typically refrain from maintaining open positions overnight, because of the risks doing that would involve.

Starting from the data thus obtained, we constructed a distinct dataset for each component stock of the list. In the case of FTSE MIB, we have 40 stocks, resulting in the construction of 40 datasets. In each dataset D_i, where i is the stock for which we want to build a model, for each instant t, we used all the log-returns $r_k(t)$, with $k \neq i$ as inputs, and we considered the log-return $r_i(t)$ as the desired output of the model. In other words, we are seeking for a model M_i such that, for all times t,

$$r_i(t) \sim M(r_1(t), \ldots, r_{i-1}(t), r_{i+1}(t), \ldots r_{40}(t)). \tag{3}$$

In compliance with the usual practice in machine learning, each dataset has been split into three subsets:

- training set: used for training the neural networks, it takes into account the stock quotes in the period between August 23rd, 2010 and October 31st, 2010;,
- validation set: used for selecting the best individuals returned by the neuro-evolutionary algorithm, it takes into account the stock quotes in the period between August 1st, 2010 and August 13th, 2010;
- test set: used for the final evaluation of the performance of the models obtained, it takes into account the stock quotes in the period between November 1st, 2010 and November 20th, 2010.

The experiments have been carried out by setting the parameters of the algorithm to the values obtained from a first round of experiments aimed at identifying the best parameter setting. These parameter values are reported in Table 1. For each modeling task, i.e., for each component of the FTSE MIB index, 40 runs were performed, with 40 generations and 60 individuals for each run.

The number of maximum epochs used to train the neural network represented by each individual was 250.

Symbol	Meaning	Value
n	Population size	60
p^{+}_{layer}	Probability of inserting a hidden layer	0.05
p^{-}_{layer}	Probability of deleting a hidden layer	0.05
p^{+}_{neuron}	Probability of inserting a neuron in a hidden layer	0.05
p_{cross}	'Desired' probability of applying crossover	0.7
δ	Crossover similarity cut-off value	0.9
N_{in}	Number of network inputs	39
N_{out}	Number of network outputs	1
α	Cost of a neuron	2
β	Cost of a synapsis	4
λ	Desired tradeoff between network cost and accuracy	0.2

Table 1. Parameters of the Algorithm.

4.2 Results and Discussion

In this subsection we report the results obtained by applying the neuro/evolutionary algorithm to the 40 FTSE MIB intraday datasets. Table 2 shows the average, over the 40 models returned by independent runs of the algorithm, of the mean square error for each of the 40 component of the FTSE MIB index, along with the standard deviation of the mean square errors. The average MSE are very small: in practice, the price variations are predicted by the model with an error between 2% and 3% of the log-return.

In addition, Table 2 reports, for each stock, the 5-minute volatility (computed as the square of the 5-minute log-returns over the entire period of observation) is reported, to allow for a comparison of the relative volatilites of the securities considered—of course, one would expect the most volatile stocks to be harder to model, and *vice versa*.

From the results, we can infer some patterns in the performance of the models discovered by the algorithm. The first aspect is that on stocks of the financial sector (for instance, BP.MI, BPE.MI, FSA.MI, ISP.MI, and UCG.MI) the approach obtains higher error values than on stocks related to the energy sector (ENEL.MI, ENI.MI, and SRG.MI). Related to the former set of stocks, the results suggest that the task of finding correlations between financial stocks and the other components of the index is harder; a possible reason is almost certainty related with the speculative nature that, especially in the last months, characterizes these stocks categories. This is confirmed by the observation that the stocks of the financial sector are, on average, more volatile than stocks of

the other sectors (4.6×10^{-5} vs. 1.2×10^{-5}). On the contrary, cyclic stocks like the ones of the energy sector, appear to be easier to model.

The second aspect is related to the set of manufacturing stocks (like F.MI, FI.MI, FNC.MI, and LUX.MI). By observing their MSE values, we can notice that the model performances vary widely. Our interpretation of these different behaviors is related to the very different contingent situations the companies are facing. Take, for instance, FIAT, who was facing financial tensions and a crisis of the automotive market during the period under investigation.

Anyway, Figure 1 shows that there is a clear correlation between the volatility of a security and the average MSE obtained by the neuro-evolutionary algorithm, as expected.

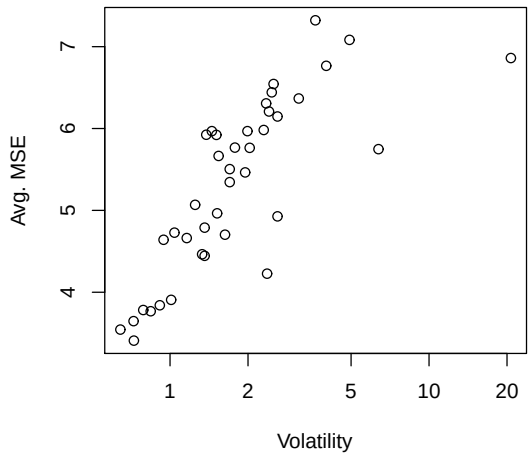

Fig. 1. Plot of the average MSE of evolved models against the 5-minute volatility of the security modeled.

We have also evaluated our approach by implementing a simple trading system based on the results generated by the algorithm. By analyzing the predictions for each observation, the system is able to perform the following decisions:

- if the prediction is higher with respect to the current price, and the system does not hold the current security, the system buys the security;
- if the prediction is higher with respect to the current price, and the system already holds the current security, the system does nothing;
- if the prediction is lower with respect to the current price, and the system does not hold the current security, the system sells short the security with

Table 2. Summary of results obtained on the FTSE MIB intraday dataset.

Ticker	Stock Name	5-min Volatility ($\times 10^{-5}$)	Avg. MSE ($\times 10^{-4}$)	St. Dev ($\times 10^{-5}$)
A2A.MI	A2A	2.37	4.227	4.3
AGL.MI	AUTOGRILL	0.944	4.641	4.9
ATL.MI	ATLANTIA	0.788	3.783	5.1
AZM.MI	AZIMUT HOLDING	2.47	6.442	2.2
BMPS.MI	BANCA MPS	6.38	5.748	1.9
BP.MI	BANCO POPOLARE	3.14	6.367	6.4
BPE.MI	BCA POP. EMILIA R.	2.35	6.307	4.5
BZU.MI	BUZZI UNICEM	1.70	5.504	6.3
CPR.MI	CAMPARI	0.725	3.409	0.7
DIA.MI	DIASORIN	1.04	4.728	9.4
EGPW.MI	ENEL GREEN POWER	1.33	4.464	8.8
ENEL.MI	ENEL	0.913	3.841	6.6
ENI.MI	ENI	0.643	3.544	0.9
EXO.MI	EXOR	1.99	5.968	5.3
F.MI	FIAT	2.60	6.147	8.1
FI.MI	FIAT INDUSTRIAL	2.41	6.208	1.0
FNC.MI	FINMECCANICA	1.51	5.922	7.6
FSA.MI	FONDIARIA-SAI	3.64	7.322	4.7
G.MI	GENERALI	1.25	5.068	4.2
IPG.MI	IMPREGILO	2.03	5.764	3.1
ISP.MI	INTESA SANPAOLO	4.01	6.766	6.1
LTO.MI	LOTTOMATICA	1.54	5.665	2.6
LUX.MI	LUXOTTICA GROUP	0.723	3.647	0.3
MB.MI	MEDIOBANCA	1.52	4.964	7.9
MED.MI	MEDIOLANUM	2.51	6.544	4.4
MS.MI	MEDIASET	1.36	4.445	9.5
PC.MI	PIRELLI & C.	1.78	5.767	1.1
PLT.MI	PARMALAT	1.63	4.703	6.8
PMI.MI	BCA POP. MILANO	20.7	6.861	4.9
PRY.MI	PRYSMIAN	1.38	5.924	2.3
SPM.MI	SAIPEM	1.16	4.662	9.4
SRG.MI	SNAM RETE GAS	0.842	3.768	0.6
STM.MI	STMICROELECTRONICS	1.70	5.345	6.5
STS.MI	ANSALDO STS	1.95	5.464	7.1
TEN.MI	TENARIS	1.36	4.789	8.0
TIT.MI	TELECOM ITALIA	2.60	4.927	7.4
TOD.MI	TOD'S	1.45	5.968	9.3
TRN.MI	TERNA	1.01	3.907	0.5
UBI.MI	UBI BANCA	2.30	5.982	4.7
UCG.MI	UNICREDIT	4.93	7.083	5.5

the price that is the average between the current price and the predicted one,
- if the prediction is lower with respect to the current price, and the system already holds the current security, the system sells the security,
- if the prediction is the same of the current price, the system does nothing.

For each security, if the system decides to sell short, at the end of the day, the system closes all eventually opened positions by buying stocks at the closure price of the related security. We supposed that the system opens each position by investing 100,000€ and that the commission for each transaction is 19€.

Table 3 shows the performance obtained by the system on each security took into account.

We can observe that the approach obtained an average return of +4.02%, which would correspond to a staggering +471% on an annual basis, net of trading commissions. Such results owe in part to the extreme simplicity of the simulation, which does not take the price impact of trades into account. As a matter of fact, for some of the less liquid securities of the FTSE-MIB, trades worth 100,000€ should be expected to significantly change the price of the traded security in the unfavorable direction. However, estimating the extent of such change is exceedingly complex and was intentionally left out of the scope of this work. Therefore, the actual performance of the trading system described would certainly be less impressive, although probably still largely positive.

5 Conclusions

In this work we presented an approach whose aim is to discover correlations between stocks in the intraday market. We performed experiments by considering the components of the FTSE MIB index of the Milan Stock Exchange and our results suggest that it is possible to obtain a high prediction accuracy for a time scale of a few minutes. In general, we found that the accuracy of the evolved models is higher the less volatile is the stock under investigation, although even the accuracies of the models for the most volatile stocks are very good. Future work in this direction will aim at improving the approach by introducing a features selection module in order to refine input data by excluding those features that introduce noise into the learning process. Moreover, further experiments will be performed on several European and non-European stock markets in order to verify the existence of correlations both on the same or on different indexes.

References

1. Azzini, A., Dragoni, M., Tettamanzi, A.: A novel similarity-based crossover for artificial neural network evolution. In: Parallel Problem Solving from Nature PPSN XI, Lecture Notes in Computer Science. vol. 6238, pp. 344–353. Springer (2010)
2. Azzini, A., Tettamanzi, A.: Evolving neural networks for static single-position automated trading. Journal of Artificial Evolution and Applications 2008(Article ID 184286), 1–17 (2008)

Table 3. Summary of results obtained on the FTSE MIB intraday dataset.

Ticker	Stock Name	Return (%)	Sharpe Ratio	Annualized Return (%)
A2A.MI	A2A	+7.15	1.20	+ 2526.61
AGL.MI	AUTOGRILL	+4.55	0.52	+ 125.25
ATL.MI	ATLANTIA	+11.75	1.91	+ 659.48
AZM.MI	AZIMUT HOLDING	−1.2	- 1.45	- 19.77
BMPS.MI	BANCA MPS	+0.55	- 0.76	+ 10.53
BP.MI	BANCO POPOLARE	−2.35	- 0.68	- 35.20
BPE.MI	BCA POP. EMILIA R.	+2.65	0.14	+ 61.17
BZU.MI	BUZZI UNICEM	+4.45	0.39	+ 121.34
CPR.MI	CAMPARI	+16.05	20.07	+ 1412.77
DIA.MI	DIASORIN	+11.2	0.98	+ 594.08
EGPW.MI	ENEL GREEN POWER	+9.85	0.89	+ 455.30
ENEL.MI	ENEL	+20.1	2.74	+ 2729.17
ENI.MI	ENI	+17.3	17.00	+ 1739.45
EXO.MI	EXOR	+3.55	0.29	+ 89.01
F.MI	FIAT	+2.2	0.03	+ 48.75
FI.MI	FIAT INDUSTRIAL	+4.65	2.65	+ 129.21
FNC.MI	FINMECCANICA	+8.4	0.84	+ 335.79
FSA.MI	FONDIARIA-SAI	−13.75	- 3.35	- 93.27
G.MI	GENERALI	−2.95	- 1.17	- 42.10
IPG.MI	IMPREGILO	+1.05	- 0.31	+ 21.01
ISP.MI	INTESA SANPAOLO	−4.3	- 1.03	- 55.16
LTO.MI	LOTTOMATICA	+2.0	0.00	+ 43.53
LUX.MI	LUXOTTICA GROUP	+23.75	72.50	+ 4785.96
MB.MI	MEDIOBANCA	+6.15	0.53	+ 197.19
MED.MI	MEDIOLANUM	−5.95	- 1.80	- 67.35
MS.MI	MEDIASET	+4.2	0.23	+ 111.87
PC.MI	PIRELLI & C.	+7.3	4.82	+ 261.78
PLT.MI	PARMALAT	+8.65	0.98	+ 354.50
PMI.MI	BCA POP. MILANO	−6.75	- 1.78	- 72.06
PRY.MI	PRYSMIAN	−0.9	- 1.26	- 15.21
SPM.MI	SAIPEM	+3.2	0.13	+ 77.68
SRG.MI	SNAM RETE GAS	+15.55	22.58	+ 1298.14
STM.MI	STMICROELECTRONICS	+6.4	0.68	+ 210.23
STS.MI	ANSALDO STS	+3.6	0.23	+ 90.68
TEN.MI	TENARIS	+4.05	0.26	+ 106.38
TIT.MI	TELECOM ITALIA	+1.1	- 0.12	+ 22.09
TOD.MI	TOD'S	+3.35	0.15	+ 82.45
TRN.MI	TERNA	+12.15	20.30	+ 710.65
UBI.MI	UBI BANCA	−7.45	- 2.01	- 75.65
UCG.MI	UNICREDIT	−20.65	- 4.11	- 98.53
	AVERAGE	**+ 4.02**		**+ 470.94**

3. Azzini, A., Tettamanzi, A.: A new genetic approach for neural network design. In: Engineering Evolutionary Intelligent Systems. Studies in Computational Intelligence. vol. 82. Springer (2008)
4. Bi, T., Zhang, B., Xu, R.: Dynamics of intraday serial correlation in china's stock market. Communications in Statistics - Simulation and Computation 40(10), 1637–1650 (2011)
5. Brabazon, A., O'Neill, M. (eds.): Natural Computing in Computational Finance, Studies in Computational Intelligence, Volume 1. Springer-Verlag (2008)
6. Brabazon, A., O'Neill, M. (eds.): Natural Computing in Computational Finance, Studies in Computational Intelligence, Volume 2. Springer-Verlag (2009)
7. Brabazon, A., O'Neill, M., Maringer, D. (eds.): Natural Computing in Computational Finance, Studies in Computational Intelligence, Volume 3. Springer-Verlag (2010)
8. Cheng, S.H.: Forecasting the change of intraday stock price by using text mining news of stock. In: ICMLC. pp. 2605–2609. IEEE (2010)
9. Chicco, D., Resta, M.: An intraday trading model based on artificial immune systems. In: Apolloni, B., Bassis, S., Esposito, A., Morabito, F. (eds.) WIRN. Frontiers in Artificial Intelligence and Applications, vol. 226, pp. 62–68. IOS Press (2010)
10. Diaz, D., Theodoulidis, B., Sampaio, P.: Analysis of stock market manipulations using knowledge discovery techniques applied to intraday trade prices. Expert Syst. Appl. 38(10), 12757–12771 (2011)
11. Groth, S., Muntermann, J.: An intraday market risk management approach based on textual analysis. Decision Support Systems 50(4), 680–691 (2011)
12. Harris, L.: Trading and exchanges: market microstructure for practitioners. Oxford University Press (2003)
13. Hestenes, M., Stiefel, E.: Methods of conjugate gradients for solving linear systems. Journal of Research of the National Bureau of Standards 49(6) (1952)
14. Mittermayer, M.A.: Forecasting intraday stock price trends with text mining techniques. In: HICSS (2004)