Balancing recall and precision in stock market predictors using support vector machines

Marco Lippi, Lorenzo Menconi, Marco Gori

Dipartimento di Ingegneria dell'Informazione, Università degli Studi di Siena.
{lippi,menconi,marco}@dii.unisi.it

Abstract. Computational finance is one of the fields where machine learning and data mining have found in recent years a large application. Neverthless, there are still many open issues regarding the predictability of the stock market, and the possibility to build an automatic intelligent trader able to make forecasts on stock prices, and to develop a profitable trading strategy. In this paper, we propose an automatic trading strategy based on support vector machines, which employs recall-precision curves in order to allow a buying action for the trader only when the confidence of the prediction is high. We present an extensive experimental evaluation which compares our trader with several classic competitors.

1 Introduction

How far is the time when most of the trading volume in financial markets will come from intelligent and automated trading strategies rather than by human interaction? And how is actually complex to build a financial strategy that can operate autonomously in the market? Research now provides several tools to approach the problem. In particular, in this paper, we use the basic concepts underlying computational finance and algorithmic trading, with the purpose of predicting the behavior of the market and of developing an intelligent trading strategy, really capable of directly operating on the market, and completely independent of human intervention.

In recent years, there has been a tremendous growth in the field of computational finance [9], with the rise of a wide variety of different automatic intelligent strategies, based on data mining and machine learning techinques. This spread of interest has been mainly due to the growing availability of data coming from the World Wide Web: nowadays, huge data flows have become accessible every day from all the stock markets around the world, and computational intelligence techniques have been increasingly employed to process and analyze stock prices and trading patterns.

One of the main goals of computational finance is to predict the evolution of the stock market, given observations of its past behavior. An example is given by forecasting the trend of a stock price: given a price time series $X_t = \{x_1, \ldots, x_t\}$ the aim is to predict some property of the series at $t + \Delta$, and use such prediction to build an automatic trading strategy. When Δ is within the order of minutes or hours, we talk of intra-day predictions, which will be the main object of investigation of this paper. Short-term prediction methods are the ones which in the last years have received most of the attention [7], and many different algorithms

have been proposed, using several machine learning or statistical techniques, including artificial neural networks, support vector machines, volatility models and many others. Typical applications of such predictors include regression (e.g., predict the future price of a stock) or classification tasks (e.g., predict the trend of a stock in the forthcoming minutes/hours). Neverthless, most of the literature lacks an accurate experimental evaluation of the proposed algorithms: sometimes only the accuracy of the predictor is measured, but no concrete analysis of the trading strategy is performed, too often commissions are not taken into account, and finally only few works directly compare *intelligent* traders with trivial strategies such as *Buy and Hold*. For these reasons, it is still an open question to infer at what extent is the stock market predictable: such question boils down to facing what is called the *random walk hypothesis* for the stock market, which states that no better estimate than the current price can be given for the future price of a given stock [4].
In this paper, we present an automatic intelligent trading system which uses support vector machines to identify the stocks to buy, while employing a sophisticated technique based on recall-precision curves in order to filter predictions, maintaining a higher level of confidence. In the following sections of the paper, we first tackle some questions regarding the general implementation of an automatic trading agent, then we describe our model based on support vector machines, and finally give an extensive experimental evaluation of our trader, in comparison with other classic approaches.

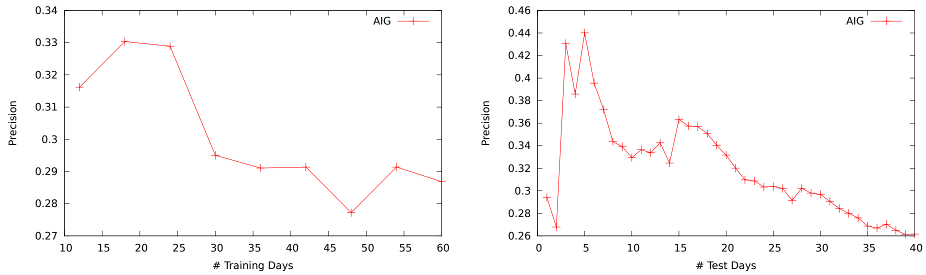
2 Limitations of automatic intelligent traders

Differently from many other contexts, in financial time series forecasting some characteristics of the domain should be taken into account when building a machine learning predictor and therefore also an intelligent trader. In this work, we use support vector machines (SVMs) to predict whether a certain stock will rise more than a certain amount within a short horizon (e.g., a few hours). The observed past time series is used to build the input features for the classifier. In our preliminary experiments we aimed at answering several questions, which are fundamental in order to create an accurate model.

1. Do larger training sets always improve SVM performance ?
2. Do performance degrade over days/weeks/months of prediction, if the predictive model is kept fixed ?
3. At what extent is the stock market really predictable ?

2.1 Training set dimension

Machine learning algorithms typically take advantage of large training sets to reduce their approximation error. Yet, in the case of time series forecasting, if the training data is too distant in the past from the current prediction time, the effect of a larger training set does not necessarily reflect in an improvement in the performance of the predictor. Figure 1(a) shows the precision of an SVM classifier – which employes the past time series as input features – as a function of the training set dimension, keeping the test set fixed. In this case, the binary classification task consists in predicting (positive class) whether the stock AIG



(a) (b)

Fig. 1. Left: learning curve as a function of training set dimension. Right: learning curve as a function of test set distribution drift.

(American International Group, NYSE) will gain at least 0.5% in the forthcoming 60 minutes: the test set consists of one whole week of data, with samples taken every 30 seconds, and the predictor is asked to make a prediction for each new sample. The time series made of the previous 60 samples is used as feature vector. It is clear from the graph that, after a certain period during which the precision of the predictor increases while growing the training set, then the performance starts degrading. It should also be noticed that we report the precision of the predictor (false positives ratio), which is always under 50 %: as it will be discussed in the following subsections, this somehow confirms the great difficulty of the task. The key idea of our SVM-based trader will be that of filtering the predictions, with the goal of maintaining a higher level of precision.

2.2 Test set horizon

Similarly to the previous case, in this second experiment we want to observe how the performance of a predictor behaves over a long test period, this time maintaining a fixed training set: this should give the idea of how often predictors should be updated in order to keep always a fresh and ready-to-use model. Figure 1 confirms that the performance degrades very quickly, suggesting that the models should be re-trained as often as possible. This phenomenon is well known in statistics and machine learning, and is called *distribution drift* or *covariate shift* [8]. When dealing with data coming from a distribution which changes over time, the performance of a predictor strongly depends on the temporal distance between training and test examples: for this reason, periodically re-training the models is a key strategy in order to get an accurate and updated forecasting system.

2.3 Stock market predictability

One of the main criticisms which are moved with regards to computational finance is that the stock market follows the random walk theory (RWT) and is actually not predictable [4]. The RWT comes as an interpretation of the efficient market hypothesis, and states that in an efficient market, on average, the information is instantly absorbed by the prices, and consequently the time-series of prices has no memory. According to our experiments, this is partially true. In fact, if an SVM is trained to predict the future price of a stock (either with a regression or a classification task) at any time, such predictor will not behave better than a random walk classifier, on average. On the other hand, this does not exclude the possibility that there might be rare predictable features and arbitrage opportunities, which could be exploited by a human or artificial trader. In particular, an artificial trader has the ability of processing continuously huge amounts of data, analyzing cross-correlations between different stocks, observing micro-trends across thousands of stocks in different markets, and can therefore take into consideration many trading strategies in parallel, which is unfeasible for humans. This is the fundamental reason for which computational finance has recently conquered a large slice of the trading market [1, 5].

In the following section we will present a machine learning predictor based on support vector machines, which will drive the trading strategy of an automatic agent, with the goal of making few transactions, but with high level of precision.

3 Trading strategy

Following the results obtained in our preliminary experiments, we designed an SVM-based trader which merges two fundamental ideas: (1) the models used at prediction time should be updated at least daily, because predictions quickly degrade if keeping the models fixed; (2) the trader should buy only if the predictor is somehow *confident*, which means that some parameters have to be designed in order to score and guide the decisions.

A different SVM is trained for each different stock in our basket. A binary classification task is structured as follows: a feature vector describes the past time series of the prices of a given stock at time t, and the goal is to predict whether the price will have a certain gain in the forthcoming future. More precisely, a positive example is given if the stock will have a percentage gain greater than K within Δ minutes, being K and Δ two tunable parameters. By training an SVM in these conditions, it is possible to build a recall-precision curve on a validation set. Recall (R) — sometimes called True Positive Rate or Sensitivity — is defined as the ratio between correctly identified positive examples and total number of positive examples: $R = \frac{TP}{TP+FN}$, while precision (P) — sometimes called Positive Predictive Value — is the ratio between correctly identified positive examples and total number of examples predicted as positives: $P = \frac{TP}{TP+FP}$. By using different SVM margins as the values for discriminating between positive and negative classes, we obtain different values for recall and precision, which can be plotted to draw a curve: we select a working point on this curve, such that precision is above a certain threshold (chosen as $\hat{P} = 90\%$) and the $F_{0.5}$ value is as maximum as possible[1]. The idea is to use the SVM margin corresponding to this working point on the RP curve as a triggering threshold τ_S for the positive class at prediction time: more precisely, if at time t the classifier outputs a margin $M > \tau_S$ for stock S, then the trader will buy S.

[1] In general, F_β combines precision and recall measures in order to obtain a single comprehensive score function: $F_\beta = (1+\beta^2)\frac{P \cdot R}{\beta^2 \cdot P + R}$. Typically, F_1 is employed, which corresponds to the harmonic mean between recall and precison. The choice of $\beta = 0.5$ gives a higher weight to the precision.

The feature vector is built as follows. Given the time series $X_t = \{x_{t-W\cdot A}, ..., x_t\}$ containing the last $W \cdot A$ samples, we defined the vector of W features $f_1, \ldots, f_W$, where feature f_j is computed as:

$$f_j = \frac{\hat{x}_j - \hat{x}_{j-1}}{\hat{x}_{j-1}} \tag{1}$$

where $\hat{x}_j$ are aggregations over A samples:

$$\hat{x}_j = \frac{x_{t-j\cdot A} + x_{t-j\cdot A+1} + \ldots + x_{t-j\cdot A+A}}{A} \tag{2}$$

Variations expressed in percentage are preferable features than plain prices, as they are independent from the absolute value of the stock. In our preliminary experiments for the design of the classifier, we also tried logarithmic returns as input features, but they headed to slightly worse performance.

Several parameters have to be tuned for each stock S during model selection: the percentage gain predicted as positive class K (in our experiments, chosen as $K = 0.1\%$ for all stocks), the training set dimension T, the prediction horizon Δ, the aggregation time for input features A, the input window W, the SVM parameter C which controls generalization. As a selling criterion, we employed a stoploss s, one of the most common tools used by investors to exit the market and limit losses. The idea is simple: when the price of a purchased stock drops below a predefined threshold, the stock is sold. This stoploss threshold can be easily made dynamic, if computed as a function of the highest price reached by the stock after its purchase:

$$lastHighestPrice(t) = \max(lastHighestPrice(t-1), price(t)) \tag{3}$$

$$stoplossThreshold(t) = lastHighestPrice(t) \cdot (1-s) \tag{4}$$

In our experiments, we use a stoploss value $s = 0.005$.

Finally, the trading strategy was designed so as to close all trading days without any shares in the portfolio: 15 minutes before the market closes, all the remaining stocks in the portfolio are automatically sold[2].

4 Experimental evalutation

We evaluate our SVM-based trading strategy on several test periods between 2010 and 2011. The data set consists in the stock price series of 86 assets of NYSE, from January 1st, 2010, to December 31th, 2011, with prices taken every 30 seconds. We use Interactive Brokers[3] as trading platform. The experiments were conducted over 12 different test periods of 6 months, starting from July 1st, 2010 up to November 31st, 2011, shifting the semester by one month for each test period (e.g., the first period runs from July 1st, 2010 to January 1st, 2011). In this way, we will observe the behavior of the traders in different market scenarios, which include both bull and bear periods.

[2] This typically corresponds to propitious trading operations, because as a matter of fact the stoploss threshold had not yet ordered a sell operation.

[3] http://www.interactivebrokers.com

4.1 Competitors

In order to have an extensive experimental evaluation, we compared the performance achieved by our SVM-based trader with several algorithms implementing different baseline trading strategies.

Random trader (RT) The comparison with a random trader allows to measure how much an intelligent strategy is better than a purely random and uninformed one. Several different implementations of a random trader can be employed: in our version, every day the trader randomly chooses a buy and a sell moment, in the range from 30 minutes after the opening to 30 minutes before closing, always performing the buying action first.

Random trader with stoploss (RST) This is a variant of RT, which uses a stoploss $s = 0.005$ to decide when to sell.

Daily open-close trader (DOCT) By using this strategy, every day a trader places a buy order 30 minutes after the opening and a sell order 30 minutes before closing. This is a simple strategy, that gives a measure of how much the market has grown in a day.

Buy and Hold trader (BHT) The Buy and Hold trading strategy offers a point of view which is similar to that of the daily open-close trader, but on a larger time horizon. The BHT buys at the beginning of the test period, holding the purchased shares until the last day, when he sells just before the closing.

Technical analysis trader (TAT) A smarter trading strategy uses the combination of four classic technical analysis tools [2]: the Bollinger's bands (BBs), the Money flow index (MFI), the On balance volume index (OBV), and the Stoploss policy (SL). The TAT buys a certain stock when the majority of its three indicators votes for a purchase. The stoploss indicator is used to decide when to sell.

4.2 Results

To evaluate the perfomance of each trading strategy, we consider the gain obtained in each semester, including commissions fees as applied by Interactive Brokers. The orders are assumed to be executed with no time lag. At each period a portfolio of 10,000 USD is initialized.

In Table 1 we report for each test period the returns of each trade strategy, averaged over all the examined stocks: the RT, RST, DOCT, BHT, TAT strategies were in fact independently run for each stock, and their performance averaged over the whole set of stocks, while our trader can decide which stock to buy. As expected, the BHT, which keeps the position open for a whole semester, has good returns in bull periods, and bad returns in bear periods: obviously, this strategy is also influenced by inter-day price gaps. Like all the trading strategies which maintain an amount of stocks for several months, its behavior is almost independent of commission fees, and has the advantages and disadvantages of all long investments. Anyhow, it should be remarked that the performance of BHT are extremely dependent on the stock which is chosen at the beginning of the strategy:

Semester	DOCT	BHT	RT	RST	TAT	SVMT
07/10 – 12/10	3.57 ± 13.37	29.94 ± 28.02	-4.34 ± 9.23	0.33 ± 9.42	1.52 ± 7.62	4.20
08/10 – 01/11	1.91 ± 13.80	20.40 ± 22.98	-3.26 ± 10.26	-2.57 ± 7.56	-0.12 ± 7.67	9.02
09/10 – 02/11	-0.83 ± 10.68	22.88 ± 21.94	-4.77 ± 8.75	-2.77 ± 8.27	-0.47 ± 7.90	6.19
10/10 – 03/11	-1.53 ± 10.82	13.61 ± 18.53	-5.88 ± 7.99	-4.14 ± 7.63	-0.11 ± 7.32	-2.30
11/10 – 04/11	-2.92 ± 10.49	12.06 ± 18.59	-4.65 ± 8.28	-4.17 ± 7.44	-0.15 ± 6.01	1.66
12/10 – 05/11	-5.27 ± 12.69	6.36 ± 19.69	-5.72 ± 7.95	-2.65 ± 7.43	-1.24 ± 6.55	0.03
01/11 – 06/11	-7.07 ± 13.83	-1.11 ± 20.48	-5.94 ± 8.37	-3.70 ± 6.60	-2.32 ± 6.66	-8.92
02/11 – 07/11	-8.16 ± 13.01	-5.30 ± 15.55	-6.60 ± 8.28	-4.96 ± 5.92	-3.13 ± 6.61	-4.47
03/11 – 08/11	-14.29 ± 14.12	-13.65 ± 17.70	-8.32 ± 10.35	-3.74 ± 6.56	-4.48 ± 8.01	-12.26
04/11 – 09/11	-18.59 ± 17.39	-23.46 ± 19.09	-8.61 ± 10.64	-6.21 ± 6.91	-6.59 ± 8.94	-9.26
05/11 – 10/11	-12.63 ± 19.29	-18.71 ± 16.70	-7.65 ± 12.21	-7.83 ± 6.69	-5.97 ± 8.79	0.23
06/11 – 11/11	-9.10 ± 19.49	-15.20 ± 18.19	-6.72 ± 12.67	-8.94 ± 7.39	-4.89 ± 9.48	-3.33

Table 1. Trading strategies comparison for 12 different test semesters between 2010 and 2011. We report the percentage gain of each trader, starting at each period with a portfolio of 10,000 USD. The SVMT has no standard deviation, as it decides on which stocks it operates every day, while for all the other competitors results are averaged on all the considered stocks: the very large standard deviations indicate that all the other traders have very heterogeneous behaviors on the different stocks.

this is confirmed by the very large standard deviations (results are averaged on the set of stocks) in the first column of Table 1, which indicate that such a strategy can bring very large gains as well as very large losses.
In the early (bull) semesters, there are considerable inter-day positive variations in price, while in the last few (bear) semesters the price changes are mostly intraday: for this reason, the DOCT earns much less then the BHT during bull periods, while having similar behavior during bear periods. The TAT, which uses BBs, MFI and OBV, performs generally better than a random trader and the DOCT, but is a very conservative trader and does not seem to offer real gain opportunities.
The SVMT, on the other hand, presents encouraging results, especially containing the losses during many bear periods. Further improvements can be applied to this trader, including a smarter analysis of model parameters, with the aim of buying stocks with high share values, so that commission fees have a lower impact on the trading performance. Similarly, also information about volumes and volatility should be included within the model. It should be remarked that, in our preliminary experiments, a plain SVM trader which does not use the recall-precision method for threshold selection headed to very poor results, with large losses in all semesters.

5 Future works

The goal of this work was to compare our SVM-based trading technique with common baselines and technical analysis strategies, which are typically employed in computational finance. Future directions of research will include a more complete experimental analysis, in order to compare the proposed approach also with other machine learning algorithms, such as neural networks, decision trees and classical time-series forecasting algorithms like SARIMA [10] or GARCH [3]. In order to improve the performance of our trading strategy, we are currently investigating techniques based on statistical relational learning, which might be used to model interdependencies between different time series: this kind of approach has been recently applied to other forecasting domains, like traffic forecasting [6]. Finally, the use of textual information obtained by several sources across the World Wide Web might greatly improve the opportunities of the automatic trader, for example to predict opening prices.

6 Conclusions

Computational finance has become nowadays a constantly evolving research area. As the amount of data becoming available with the world wide web has enormously grown in the last years, huge money streams are daily moved by computers all around the world. Machine learning and data mining have therefore found in finance a challenging and profitable domain.

In this paper we presented an automatic trading strategy based on support vector machines, which uses recall-precision curves in order to suggest a buying action only when the confidence level of the trader is high. We compared our trader on several test periods between 2010 and 2011, showing strongly encouraging results with respect to classic baseline traders, even based on technical analysis.

Many future directions of research can be developed: relational learning techniques can be used to capture relations among stocks; volumes and volatility information can be added to our model; text mining techniques can be applied in order to acquire knowledge from news, blogs and forums around the web, to guide the choices of the trader.

References

1. A. Chaboud, B. Chiquoine, E. Hjalmarsson, and C. Vega. Rise of the machines: algorithmic trading in the foreign exchange market. International Finance Discussion Papers 980, Board of Governors of the Federal Reserve System (U.S.), 2009.
2. R.D. Edwards, J. Magee, and W.H.C. Bassetti. *Technical analysis of stock trends*. John Magee Investment Series. St. Lucie Press, 2001.
3. R. Engle. GARCH 101: The Use of ARCH/GARCH Models in Applied Econometrics. *The Journal of Economic Perspectives*, 15(4):157–168, 2001.
4. E. F. Fama. Random Walks in Stock Market Prices. *Financial Analysts Journal*, 21:55–60, 1965.
5. A. A. Kirilenko, A. P. Kyle, M. Samadi, and T. Tuzun. The flash crash: The impact of high frequency trading on an electronic market. *Social Science Research Network Working Paper Series*, October 2010.
6. M. Lippi, M. Bertini, and P. Frasconi. Collective traffic forecasting. In *ECML/PKDD 2010*, pages 259–273, 2010.
7. L. Menconi, M. Gori, and M. Lippi. Computational models for short-term prediction of the stock market. *Intelligenza Artificiale*, 5(2):217–227, 2011.
8. J. Quionero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009.
9. R. U. Seydel. *Tools for Computational Finance*. Springer, 2009.
10. B. M. Williams and L. A. Hoel. Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results. *Journal of Transportation Engineering*, 129(6):664–672, 2003.