Stock Market Prediction

Radu Iacomin
Faculty of Automatic Control and Computers
University POLITEHNICA of Bucharest
Bucharest, Romania
Email: raduiacomin@yahoo.com

***Abstract*—In a financially volatile market, as the stock market, it is important to have a very precise prediction of a future trend. Because of the financial crisis and scoring profits, it is mandatory to have a secure prediction of the values of the stocks. Predicting a non-linear signal requires advanced algorithms of machine learning. The literature contains studies with different machine learning algorithms such as ANN (artificial neural networks) with different feature selection. The results of this study will show that the algorithm of classification SVM (Support Vector Machines) with the help of feature selection PCA (Principal component analysis) will have the success of making a profit.**

***Index Terms*—Nonlinear Systems; Computational Intelligence; Optimization.**

I. Introduction

Problem statement: Stock exchanges are financial institutions which allow transferability of different goods (monetary values, actions, precious metals) between stock broker components. With a turnover of trading around thousands of billions dollars, this gets people's eager attention of making a profit. Goods are traded on the market, following their subsequent value to determine if the transaction has generated profit or not. In general the value of a stock is given by its entry on the stock exchange and the volume of its transactions. The more a share is transacted, the more it is valuable, and conversely, if a share is put into transaction in a low volume, it is not so important for some traders and by default its value decreases. This anticipation of the market can generate profits or losses, depending on the power to predict future values. Therefore the problem becomes: for a given stock market history, determine the moment of buying/entry or selling/exit the good for generating profit. An aspect that has attracted researchers is predicting the values of the goods. A wave of algorithms in Artificial Intelligence, namely Machine Learning, tried to tackle the problem described above. Some of them are: AR models (Autoregressive), ARIMA (autoregressive integrated moving average), CBR (Case-Based Reasoning), ANN (Artificial Neural Networks) [4], GA (Genetic Algorithm), SVM (Support Vector Machines)[1], SVR(Support Vector Regression) [3], PCA (Principal component analysis). Due to the non-linear nature of the stock market signals, some methods have yet to give promising answers, others have not reacted as well on the stock market exchange. Sub-issues that have risen:

1) How to determine what features are optimal?
2) How is the problem defined as? (Classification and Regression)
3) How to connect algorithms to optimize the efficiency of prediction

The structure of the paper is as follows. In Section II I present a short description of the financial overview, Section III describes the algorithms and the tuned properties of support vector machines and principal component analysis used in this study. In Section IV, I report the data set and numerical results of the algorithm with an application on the stock market trade.

II. Economic perspective

From the economical point of view, the traders, brokering tools view the stock market as a volatile market. The best thing to do is to analyze everything and figure out the time to buy/sell the goods. Traders also make bad decisions. Because of the high number of traders, mathematical models have risen and have been helping their decision making are called technical indicators and they are used to generate decisions, trends, volatility bands, risk changes. The models are generated from the four prices of the stock market goods. Open, Close, High, Low represent the stages of the price depending on the sampling time: Opening price is the price at which a stock first trades upon the opening of an exchange on a given trading day. Closing price is the final price at which a stock is traded on a given day (sampling time). High and low prices represent the maximum and minimum a price can reach on a given day. Optimal decisions are given when a technical indicator makes an entry or exit decision that generates profit. Because of the high number of technical indicators, it is a plus to require determining which are better and minimizing the number of relevant features.

III. Computer science perspective

Nowadays, it is mandatory to find the best methods of resolving the main problem and sub-problems. This section is designed to explain the computer and mathematical algorithms that have been used in this study.

A. Classification

1) Linear case: Assuming that there are n labeled examples $(x_1, y_1), \dots, (x_n, y_n)$ with labels $y_i \in \{1, -1\}$, find the hyperplane with parameters (w, b) , $< w, x > +b = 0$ satisfying the following conditions:

a) As the scale of (w, b) is fixed, the plane must be in canonical position with regard to $\{x_1, \dots, x_n\}$, i.e.,

$$\min_{i \leq n} | < w, x_i > +b| = 1$$

978-1-4799-8481-7/15/$31.00 ©2015 IEEE

b) The parameters (w, b) defining the plane that separates the $+1$ labels from the -1 labels, i.e.,

$$y_i(< w, x_i > +b) \geq 0 \text{ for all } i \leq n$$

c) The maximum margin of the plane is $\rho = 1/\|w\|$, i.e., $\|w\|^2$ is minimum.

It is not mandatory to have a separating plane for the observed data. Assuming this time that the data are indeed linearly separable the general case will be solved later as shown in [1]. Conditions 1 and 2 can be combined into just one condition:

$$y_i(< w, x_i > +b) \geq 1 \text{ for all } i \leq n$$

Up to this point, the optimization problem below has to be solved

$$\text{minimize } \frac{1}{2}\|w\|^2$$

over all $w \in R^d$ and $b \in R$ subject to the constraint

$$y_i(< w, x_i > +b) - 1 \geq 0 \text{ for all } i \leq n$$

The simple quadratic programming problem can be solved with easily available algorithms of complexity $O(n^3)$ that are used to solve this problem. For instance the methods known as interior point algorithms that are adaptations of the Karmarkar's algorithm for linear programming will succeed. Unfortunately, if n and d have sizes of tens of thousands, even the leading QP methods will not attain success [2]. The beneficial distinctive of SVMs is that most of the data turn out to be insignificant. The relevant data are just the points that have the coordinates on the edge of the optimal classifier and the points represent a meager portion of n.

The Karush - Kuhn - Tucker Theory (KKT) problem that needs to be solved is an exceptional case of the familiar problem of minimizing the convex function $f(x)$ subject to n inequality constraints $g_j(x) \geq 0$ for $j = 1, 2, \ldots, n$ where the g_j functions are likewise convex. This problem will be called (CO). Observe that in this case $x = (w, b) \in R^{d+1}$ and the constraints are linear in the unknown x. There is no relation between the optimization variable x here and the coordinates x_i above.

The Karush-Kuhn-Tucker conditions give the characterization of the solution to the convex optimization problem (CO).

KKT-Conditions: $\bar{x}$ solves the (CO) problem if and only if there exists $\bar{\lambda} = (\bar{\lambda}_1, \ldots, \bar{\lambda}_m) \geq 0$ a vector of non-negative Lagrange multipliers that features $(\bar{x}, \bar{\lambda})$ the saddle point of the Lagrangian

$$L(x, \lambda) = f(x) - \sum_{j=1}^{n} \lambda_j g_j(x)$$

i.e., for all x and for all $\lambda \geq 0$ the following relations hold:

$$L(\bar{x}, \lambda) \leq L(\bar{x}, \bar{\lambda}) \leq L(x, \bar{\lambda})$$

$$\max_x \min_{\lambda \geq 0} L(x, \lambda) = L(\bar{x}, \bar{\lambda}) = \min_{\lambda \geq 0} \max_x L(x, \lambda)$$

2) The Dual: The $min(max) = max(min)$ characterization of the saddle point of the Lagrangian L brings a different approach to solve the (CO) problem. Alternately minimizing $f(x)$ subject to the $g_j(x) \geq 0$ constraints, the problem becomes maximize $W(\lambda)$, where

$$W(\lambda) = \min_x L(x, \lambda)$$

subject to the constraint $\lambda \geq 0$. This offers a substitute way of solving the equivalent saddle point of L.

B. The Support Vectors of the SVMs

Applying the KKT-conditions to the initial problem of finding the separating hyperplane with maximum margin. In this case the Lagrangian is,

$$L(w, b, \lambda) = \frac{1}{2}\sum_{i=1}^{d} w_i^2 - \sum_{j=1}^{n} \lambda_j \{y_j(< w, x_j > +b) - 1\}$$

and the optimal KKT-conditions are

$$\begin{aligned} \nabla_w L &= 0, \text{i.e., } w = \sum_{j=1}^{n} \lambda_j y_j x_j \\ \nabla_b L &= 0, \text{i.e., } \sum_{j=1}^{n} \lambda_j y_j = 0 \\ \lambda_j \{y_j(< w, x_j > +b) - 1\} &= 0, \text{ for all } j \leq n \end{aligned}$$

C. The Dual Problem of the SVMs

The dual problem of the SVMs is even more straightforward than the primal and its formulation leads to a wonderful non-linear generalization. Given a vector λ of Lagrange multipliers, the minimizer of $L(w, b, \lambda)$ with respect to (w, b) must prove the optimality conditions shown before, i.e., w is a linear combination of the x_j's with coefficients $\lambda_j y_j$ that should sum up to nil. Thus, the dual formulation is obtained by replacing these conditions into $L(w, b, \lambda)$

$$\text{maximize } W(\lambda)$$

where,

$$W(\lambda) = \sum_{j=1}^{n} \lambda_j - \frac{1}{2}\sum_{i,j} \lambda_i \lambda_j y_i y_j < x_i, x_j >$$

and the $\lambda_j \geq 0$ satisfying

$$\sum_{j=1}^{n} \lambda_j y_j = 0$$

Maximizing $W(\lambda)$ over $\lambda \geq 0$ subject to the linear constraint presented before, so this becomes the favored form to transmit into a QP algorithm. Once an optimal λ is achieved, the w is found as the linear combination of the x_j as shown above and the b is found by evoking that the plane has to be in canonical position so that

$$\min_{i \leq n} y_i(< w, x_i > +b) = 1 = y_j(< w, x_j > +b) \text{ for all } j \in J_0 \quad (1)$$

where J_0 ={j : x_j is a support vector}, and it turns into

$$b = y_j - < w, x_j >$$

Multiplying by λ_j and summing j the equation becomes

$$b = \frac{-\sum_{i,j} \lambda_i \lambda_j y_i y_j < x_i, x_j >}{\sum_j \lambda_j}$$

and it could be easily verified that the value concurs with the value of the Lagrange multiplier β correlated to the constraint $\sum_i \lambda_i y_i = 0$ (finding $\nabla_\lambda \mathcal{L} = 0$ for the Lagrangian correlated to the dual, i.e., $\mathcal{L} = W(\lambda) - \beta \sum_i \lambda_i y_i$). The contemporary QP solvers based on interior point algorithms are generally used to determine the optimal values of β and of λ.

D. Non-linear case

The artificial approach which was talked about in the previous point is related to the scalar product. It is considered a mapping of the product in another space, Euclidian H (possibly infinite dimensional space), where introducing a kernel instead of the scalar product would eliminate the need to exactly know the function. The most commonly used kernel is Radial Base Function (RBF).

$$K(x_i, x_j) = e^{-\frac{\|x_i - x_j{}^2\|}{2\sigma^2}} \tag{2}$$

Therefore, the problem becomes:

$$\begin{aligned} max\ L_D = \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \sum_{i=1}^{L} \alpha_i \alpha_j y_i y_j \boldsymbol{K}(x_i, x_j)\, a_i \\ \geq 0\, \forall i, \sum_{i=1}^{L} \alpha_i y_i = 0 \end{aligned} \tag{3}$$

$$\begin{aligned} max\ L_D = \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \sum_{i=1}^{L} \alpha_i \alpha_j y_i y_j e^{\gamma(\|x_i - x_j{}^2\|)} a_i \geq 0 \\ \forall i, \sum_{i=1}^{L} \alpha_i y_i = 0 \end{aligned} \tag{4}$$

Replacing

$$H_i j = y_i y_j e^{(\gamma(\|x_i - x_j\|^2)} \tag{5}$$

in the linear programming problem, the optimization problem becomes:

$$\begin{aligned} max\ \alpha\ > \sum_{i=1}^{L} \alpha_i - \frac{1}{2} \alpha^T \boldsymbol{H} \alpha \\ \text{with constraints } \alpha_i \geq 0\, \forall i, \\ \sum_{i=1}^{L} \alpha_i y_i = 0 \end{aligned} \tag{6}$$

E. Feature Selection Algorithms

Because of the high and different technical features of the stock markets, it is better to implement solutions of choosing the features that are proven correct for the stock analyzed and for the period of the year. Feature selection can be done hierarchical or intelligent as presented in [7]. In the algorithm section, the intelligent algorithms that were used in this study are described.

F. Principal component analysis

PCA is a method that projects a dataset to a new coordinate system by determining the eigenvectors and eigenvalues of a matrix. It involves a calculation of a covariance matrix of a dataset to minimize the redundancy and maximize the variance. Mathematically, PCA is defined as an orthogonal linear transformation and assumes that all basis vectors are an orthonormal matrix. PCA is concerned with finding the variances and coefficients of a dataset by finding the eigenvalues and eigenvectors. [6].

The PCA is computed by determining the eigenvectors and eigenvalues of the covariance matrix. The covariance matrix is used to measure how much the dimensions are varying from the mean with respect to each other. The covariance of two random variables (dimensions) is their tendency to vary together as:

$$cov(X, Y) = E[E[X] - X] \cdot E[E[Y] - Y] \tag{7}$$

where $E[X]$ and $E[Y]$ denote the expected value of X and Y respectively. For a sampled dataset, this can be explicitly written out.

The covariance matrix is a matrix A with elements $A_i, j = cov(i, j)$. It centers the data by subtracting the mean of each sample vector. In the covariance matrix, the exact value is not as important as its sign (i.e. positive or negative). If the value is positive, it indicates that both dimensions increase, meaning that as the value of dimension X increased, so did the dimension Y. If the value is negative, then as one dimension increases, the other decreases. In this case, the dimensions end up with opposite values. In the final case, where the covariance is zero, the two dimensions are independent of each other. With the covariance matrix, the eigenvectors and eigenvalues are calculated. A common method for finding eigenvectors and eigenvalues in non-square matrices is the singular value decomposition (SVD).

G. Validation criteria

The validation of the results is made by using the following methods:

1) Hit Ratio: Performance of the algorithm, correct observations over all observations

$$R = \frac{Card(Correct\_decisions)}{Total\_decisions} \tag{8}$$

2) Rate of Recognition: Performance rate to avoid over generating decisions

$$RoR = \frac{Card(Decisions\_generated)}{Card(Indicator\_decisions)} \quad (9)$$

3) Accuracy: Performance method to determine accurate decisions generated

$$Acc = \frac{Card(Correct\_decisions\_generated)}{Card(Indicator\_decisions)} \quad (10)$$

4) Relevant Rate of Recognition: Efficiency method to determine the correct and optimal decisions

$$RRR = \frac{Card(Optimal\_Dec\_Correctly\_Class)}{Card(Total\_Decisions)} \quad (11)$$

IV. EXPERIMENTAL STUDY

This section presents the data set, the algorithms used, the implementation issues and the proposed solutions. All of the algorithms and scripts were used from Matlab Toolboxes or built in Matlab.

A. Data set

The data set has been acquired from Bloomberg [9], a platform for trading stocks. Because of the high range of domains of trading the study was directed on a volatile and fast domain, the foreign exchange market, more precise the daily sample time of trading "forex". Because of the positioning of 3 important continents, there are 3 stock markets that function to complete all the time zones: New York, London and Tokyo. Most of them are rated in USD, but there are also rated with the local money. For this study there were chosen 16 forex stocks and their reverses, a total of 32 e.g. EUR-CHF is the Euro rated in Swiss Franc.

With these sets available from 1992 to nowadays, the more relevant timestamp was selected, the last period from 4 November 2008 to 7 January 2014, 6 years of daily signals. The main goal is to detect the direction and price of 8 January 2014 and to validate the algorithms for a real prediction.

B. SVM

In this part the technical inputs were fed from all of the indicators into one SVM to make a big hyper plane in R^n, where n is the number of technical indicators. Minimizing the Euclidian distance between the spaces and the support vectors is time-consuming and requires computational power, the issue being the dimensionality of the data set (n). After several trials, the Hit Ratio (8) reached the average of 53% for the used forex dataset (IV-A). Thus, to minimize the time and computation of the SVM, it will be important to reduce the space of the data set. There are algorithms in section III-E that discuss methods of selecting the important features of the data set. The study was performed using the feature selection with Genetic Algorithms and Principle Component Analysis that are shown below.

C. GASVM

Because of the poor result using the SVM algorithm without a proper selection of features, there had to be another trial by adding a new algorithm before feeding the classification predictor. This method can generate multiple solutions using techniques inspired by natural evolution, such as inheritance, mutation, selection, and crossover as described in [4], trying to reach the optimal solution, with some constraints that make the GA algorithm for selection more efficient. The following technique is used: the one-point cross-over, that randomly selects a crossover point within the chromosome where the chromosome is made from 10 genes, in this case each gene is a technical indicator, each one with "0" or "1", meaning that it is used or not to determine the decision of the stock, then two parent chromosomes are interchanged to produce two new offsprings and so on. The chromosomes are mutated with a probability of 0.01 per gene by randomly changing technical indicators from "0" to "1" and reverse. Finally, if the algorithm stops at the stop criteria (at least 80% Hit Ratio) or the maximum number of generations (250), it will generate the best solution it could find in these conditions. Using GA methods for feature selection can determine the key features of the stock, minimizing the space and features. The first 75% of the fold was used for the training and the last 25% of the fold for validation. The data set was divided in 2 folds, trained and validated each of them using the SVM output. If the decisions were close enough to the real decisions, then the parameters were saved for creating the prediction model. This last method is called calibration. After the calibration was finished, a model with the specific parameters (Box constraint and γ) was generated to predict future trends. The 10 technical indicators were:

Indicator	Description
Commodity Channel Index	Identifying new trends or announcing the instability variation
Accumulative Swing Index	Generating signals from the maximums and minimums of the price
Williams R	Identifying the current close price level depending on the last maximums
Swing Index	Generates the trend of the price in the short future
Polarized Fractal Efficiency	Explains the geometry of fractals in the form of an oscillator
Time series forecast	Illustrates statistic trends if the safety price of a period
Ravi	Uses 2 moving averages calculated in percentage to generate a decision
DMX	Uses a Jurik moving average on directional technical indicators of the price
Volatility Breakout	Generates the security and volatility of a stock
Schaff Trend Cycle	Generates trends by combining stochastically moving averages

The final 10 samples from the dataset were tested as the final validation. During the training period, the hit ratio (R) (8) and the Accuracy (10) , meaning the number of decisions generated correctly, were observed. The results were compared to the real decisions generated from the technical indicators. The optimization of SVM can be done varying C and σ and is useful for modeling more accurate on unknown data. This can be introduced in the calibration of the algorithm. The best way to do so is to generate the near future, and not to retrain the adaptation of the algorithm for new sets of data.

As mentioned above, the SVM can classify the entry and exit on a stock. SVM creates a space with 2 subspaces that are delimited by a hyper plane. The subspaces are Buy and Sell, meaning the actions that can be taken by traders. Thus, the SVM can predict the directional movement of the stocks. The results are as follows:

Training:

C	Sigma	R	Acc train	Accuracy
32768	0.2	24/32	0.70833	0.65079
32768	0.2	34/32	0.58824	0.5873

Validating:

C	Sigma	R	Acc train	Accuracy
32768	0.5	48/51	0.375	0.72
32768	18.5	48/51	0.64583	0.63
32768	3.5	42/19	0.38095	0.71

where C is the Boxcontraint of the SVM, Sigma is the tuning parameter of the RBF kernel explained in III-D, R (Hit Ratio) is explained in (1) and Accuracy explained in (3). The Acc train at Validating is the accuracy the model has generated from earlier datasets.

D. PCASVM

As presented in the algorithm section, the PCA algorithm of selecting the features is adapted from the same one in Matlab to generate not the most important components of a column, rather to generate the most important features of a stock market that can be considered measurements of a daily signal for a foreign exchange stock (column). The output of the PCA function generates a scoring for all the components of the data (principal and latent). Because the technical indicators are fed as vectors in a matrix of all of the inputs, the best indicator is generated by the best sums of the scoring vectors (columns). Thus having an ordering of the features that can be done by summing the principal component values, column-wise as the PCA does is achieved. The first 9 sums pass through the more efficient algorithm mcSVM as described in IV-D1, leaving the latent components not to train the classifications with different than principal components. During the feature selection, not for all of the 32 forex stocks the PCA generated the same features, and the most chosen are presented in the next table, with a short description of how the technical indicator is calculated.

The indicators selected by PCASVM are:

Indicator	Description
Moving Average	Exponential moving average
Relative strength index	Momentum oscillator that measures price variations and the speed of variations
Williams R	Identifying the current close price level depending on the last maximums
Bollinger Bands	Measures the price volatility and uses it for dynamic levels of support and resistance
Rate of Change	Illustrates the variation of the price for a given period
Stochastic Oscillator	Momentum oscillator that measures the distance between the close and the range
Ichimoku	Determines the tendency of buying or selling of the stock
Linear Regr. Slope	Determines the slope of linear regression of a period
Aroon Oscillator	Determines the level of overbought and oversold in the last period

1) mcSVM: The existing methods can roughly be divided between two different approaches: the single machine approach, which attempts to construct a multi-class SVM by solving a single optimization problem, and the divide and conquer approach, which decomposes the multiclass problem into several binary sub-problems, and builds a standard SVM for each. The most popular decomposing strategy is probably the one against all, which consists of building one SVM per class, trained to distinguish the samples in a single class from the samples in all remaining classes. According to the literature review, it seems impossible to conclude which multi-class SVM [5] is better for stock market decision recognition, but it was discovered that ”one against one” is a better approach for eliminating the false decisions. From the technical indicators, it is more efficient to eliminate the decisions of Buy and Sell at the same time, thus it is to have 3 binary classificators: one Buy against Hold for detecting Buy decisions, Sell against Hold for detecting Sell decisions and the last classifier that can work as a filter for Buy and Sell detecting the false decisions of the algorithm, Buy against Sell. To improve the accuracy, Buy against Non-Buy and Sell against Non-Sell are introduced to increase the space of other decisions and to eliminate the Hold decision, that is not that relevant.

2) Measuring : To have just one decision out of the 3 binary classifiers, a voting system was introduced, namely to choose the most dominant decision, having 2 votes out of 3 available. For example, the Buy decision can generate 1 from Buy vs NonBuy and 1 from Buy vs Sell and 1 from Sell vs NonSell, meaning it has reached 3 votes, another example Sell decision can generate 1 from Sell vs NonSell and 1 from buy Buy vs Sell and 0 from Buy vs NonBuy, meaning it has reached 2 votes. In the worst case it can receive 3 different votes the algorithm generates a Hold position.

For measuring the efficiency of the PCASVM algorithm, it was necessary to introduce the Relevant Rate of Recognition that is defined as explained in (11).

To ease the software and algorithm, the RBF parameters for the SVM in PCASVM are made with boxcontraint C=1 and σ =1.

E. Results

Below are shown the graphic results for the foreign exchange stock market EURO-CHF using the two algorithms described in this paper. For the GASVM algorithm (Fig. 1) the Rate of Recognition, ROR, that counts the overlapping buying or selling points, is 55%, meaning that only this amount of decisions are going to generate profit. In Fig. 2 is the implementation of the PCASVM algorithm, which has a slightly increased ROR of 68%.

Regarding the SVR presented in [3] and [8], and using the NMSE error for an accurate prediction, the SVR can help the traders with some information about the price and also about the trend. The only problem with the regression algorithms is that, in general, they tend to have a sample delay. In a market where time and accuracy are very important to be efficient, a small delay is not allowed as a viable error, so the SVR algorithm can be used just as a filter to eliminate the wrong buy and sell signals and having an approximate idea of how big is the difference from one day to another. This is helpful to detect the big fluctuations, to prevent the large losses and to increase the large winnings.

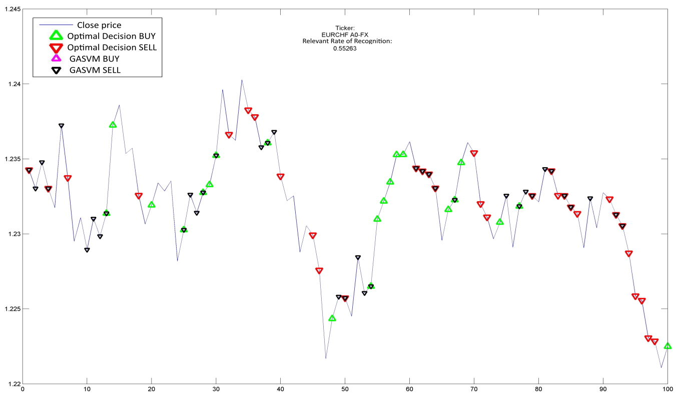

Fig. 1. Graphical result for one stock market using GASVM algorithm

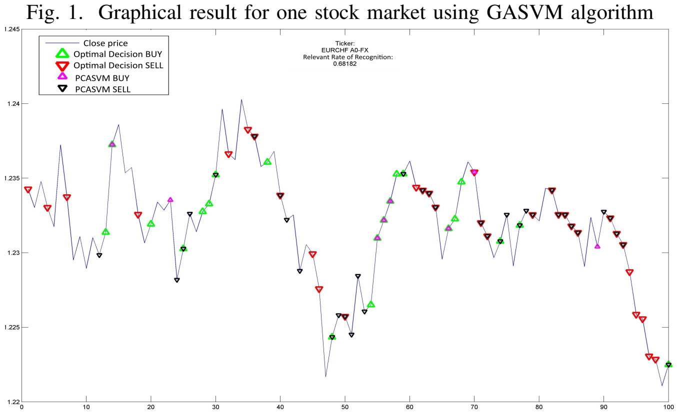

Fig. 2. Graphical result for one stock market using PCASVM algorithm

V. Conclusions

This study generated a new algorithm on predicting the stock markets. As it was shown in this paper, there are several methods on predicting signals through machine learning algorithms and numerical methods. PCASVM was implemented to both eliminate the false predictions and to determine what features are important. Comparing to the simple methods from SVM and evolving to GASVM and PCASVM, the solution to the main problem and sub-issues is more efficient and shows promising results for a real prediction using recent data sets.

Acknowledgment

The authors would like to thank prof. Bogdan Dumitrescu for the guidance on numerical methods and machine learning algorithms.

References

[1] Cortez C. and Vapnik, V., *Support-Vector Networks. Machine Learning*, 1994.
[2] Boyd S. and Vandenberghe L. , *Convex optimization*, Book, 2003.
[3] Yeh C, Huang C., Lee S. , *A multiple-kernel support vector regression approach for stock market price forecasting* , Expert Systems with Application. 2010.
[4] Yeh C, Huang C., Lee S. , *Foreign-exchange-rate forecasting with Artificial Neural Networks* , Book 2007.
[5] Duan K. , *Which Is the Best Multiclass SVM Method? An Empirical Study* , Expert Systems with Application. 2006.
[6] J Zhang, X Zhang, S Ha , *A novel approach using PCA and SVM for face detection.* , Natural Computation. 2008.
[7] Hazm K , *Analysis of Local Appearance-based Face Recognition: Effects of Feature Selection and Feature Normalization* , Computer Science Department, Universitt Karlsruhe 2006.
[8] Clark R., *Support Vector Regression* , Mathworks. 2013.
[9] Bloomberg L.P., *Bloomberg Data Sets* , www.blooomberg.com 2013.