Expert Systems with Applications 42 (2015) 7046–7056

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Evaluating multiple classifiers for stock price direction prediction

Michel Ballings [a,*], Dirk Van den Poel [b], Nathalie Hespeels [b], Ruben Gryp [b]

[a] The University of Tennessee, Department of Business Analytics and Statistics, 249 Stokely Management Center, 37996 Knoxville, TN, USA
[b] Ghent University, Department of Marketing, Tweekerkenstraat 2, 9000 Ghent, Belgium

ARTICLE INFO

Article history:
Available online 14 May 2015

Keywords:
Ensemble methods
Single classifiers
Benchmark
Stock price direction prediction

ABSTRACT

Stock price direction prediction is an important issue in the financial world. Even small improvements in predictive performance can be very profitable. The purpose of this paper is to benchmark ensemble methods (Random Forest, AdaBoost and Kernel Factory) against single classifier models (Neural Networks, Logistic Regression, Support Vector Machines and K-Nearest Neighbor). We gathered data from 5767 publicly listed European companies and used the area under the receiver operating characteristic curve (AUC) as a performance measure. Our predictions are one year ahead. The results indicate that Random Forest is the top algorithm followed by Support Vector Machines, Kernel Factory, AdaBoost, Neural Networks, K-Nearest Neighbors and Logistic Regression. This study contributes to literature in that it is, to the best of our knowledge, the first to make such an extensive benchmark. The results clearly suggest that novel studies in the domain of stock price direction prediction should include ensembles in their sets of algorithms. Our extensive literature review evidently indicates that this is currently not the case.

1. Introduction

Predicting stock prices is an important objective in the financial world (Al-Hmouz, Pedrycz, & Balamash, 2015; Barak & Modarres, 2015; Booth, Gerding, McGroarty, 2014), since a reasonably accurate prediction has the possibility to yield high financial benefits and hedge against market risks (Kumar & Thenmozhi, 2006). A great point of discussion in literature is whether stock price behavior is predictable or not. For a long time investors accepted the Efficient Market Hypothesis (EMH) (Malkiel & Fama, 1970). This hypothesis states that no abnormal returns can be obtained by studying the evolution of stock prices' past behavior (Tsinaslanidis & Kugiumtzis, 2014; Yeh & Hsu, 2014). In the beginning of the 21st century however, some economists indicated that future stock prices are at least partially predictable (Malkiel, 2003). Therefore a lot of prediction algorithms have been explored and showed that stock price behavior can indeed be predicted (Huang, Yang, & Chuang, 2008; Ou & Wang, 2009). However predictable, it remains hard to forecast the stock price movement mainly because the financial market is a complex, evolutionary, and non-linear dynamical system which interacts with political events, general economic conditions and traders' expectations (Huang, Nakamori, & Wang, 2005).

Different techniques have already been explored for stock price direction prediction. One of the best performing algorithms in the financial world appears to be Support Vector Machines (SVM) (Huang et al., 2005; Kim, 2003; Lee, 2009). Other well-known techniques are Neural Networks (Kim & Chun, 1998), Decision Trees (Wu, Lin, & Lin, 2006), Logistic Regression (Brownstone, 1996), Discriminant Analysis (Ou & Wang, 2009) and K-Nearest Neighbors (Subha & Nambi, 2012). However, most studies ignore ensemble methods in their benchmarks. To the best of our knowledge Kumar and Thenmozhi (2006), Rodriguez and Rodriguez (2004), Lunga and Marwala (2006) and Patel, Shah, Thakkar, and Kotecha (2015) are the only four studies in the domain of stock price direction prediction that use an ensemble method. This is an important gap in literature because ensemble methods have been proven to be top performers in many other areas such as customer churn behavior (Ballings & Van den Poel, 2012), social media analytics (Ballings & Van den Poel, 2015) and unsupervised word sense disambiguation (WSD) (Brody, Navigli & Lampata, 2006).

In our study we will therefore include several ensemble methods such as Random Forest (RF) (Breiman, 2001), AdaBoost (AB) (Freund & Shapire, 1995) and Kernel Factory (KF) (Balings & Van den Poel, 2013) in our benchmark. While others conduct discrete analyses to predict exact stock prices, we focus on classification models (Leung, Daouk & Chan, 2000). Literature shows that forecasting the direction is enough to execute profitable trading strategies (Cheung, Chinn, & Pascual, 2005; Pesaran & Timmerman, 1995). Hence, we predict the direction of stock prices instead of absolute stock prices. The main contribution of this study is an

* Corresponding author.
E-mail addresses: Michel.Ballings@utk.edu (M. Ballings), Dirk.VandenPoel@UGent.be (D. Van den Poel), Nathalie.Hespeels@UGent.be (N. Hespeels), Ruben.Gryp@UGent.be (R. Gryp).

http://dx.doi.org/10.1016/j.eswa.2015.05.013
0957-4174/

extensive benchmark comparing the performance of ensemble methods (RF, AB and KF) and single classifier models (Neural Networks (NN), Logistic Regression (LR), SVM, K-Nearest Neighbors (KNN)) in predicting the stock price direction. We hypothesize that, given their superiority in other domains, ensemble methods will outperform the single classifier methods.

The remainder of this paper is structured as follows. In Section 2 we will review the literature on which algorithms have been used for stock price direction prediction. Section 3 details our methodology for benchmarking the ensemble methods against other algorithms. Section 4 discusses the results. Section 5 concludes this study and Section 6 describes limitations and avenues for future research.

2. Literature review

The use of prediction algorithms is in contradiction with one of the basic rules in finance, the Efficient Market Hypothesis (EMH) (Malkiel & Fama, 1970). This hypothesis states that if one can get an advantage from analyzing past returns, the entire financial market will notice this advantage and as a consequence the price of the share will be corrected. This means that no abnormal returns can be obtained by examining past prices and returns of stocks. Although the EMH is generally accepted, it was initially based on traditional linear statistical algorithms (Malkiel & Fama, 1970). Many researchers have already rejected the hypothesis by using algorithms that can model more complex dynamics of the financial system (Lo, Mamaysky, & Wang, 2000; Malkiel, 2003). Since methods handling the complex and non-linear financial market are yielding positive results, researchers still try to invent better techniques.

There are three major methodologies to predict the stock price behavior: (1) technical analysis, (2) time series forecasting and (3) machine learning and data mining (Hellström & Holmströmm, 1998). The first category uses charts and plots as a principal tool. Analysts use these plots to make a buy or sell decision. The second category aims at predicting future stock prices by analyzing past returns on stock prices. Common methods are the autoregressive method (AR), the moving average model (MA), the autoregressive-moving average model (ARMA) and the threshold autoregressive model (TAR). The third category, data mining, is "the science of extracting useful information from large data sets or databases" (Hand, Manilla & Smyth, 2001). The popularity of data mining in the financial world has been growing since the main problem with predicting stock price direction is the huge amount of data. The data sets are too big to handle with non data mining methods such that they obscure the underlying meaning and one cannot obtain useful information from it (Fayyad, Shapiro & Smyth, 1996; Widom 1995).

Several algorithms have been used in stock price direction prediction literature. Simpler techniques such as the single decision tree, discriminant analysis, and Naïve Bayes have been replaced by better performing algorithms such as Random Forest, Logistic Regression and Neural Networks. General-purpose solvers such as Genetic Algorithms (Kuo, Chen, & Hwang 2001) have also been used but generally perform worse and are computationally more expensive. The majority of stock price direction prediction literature has focused on Logistic Regression, Neural Networks, K-Nearest Neighbors and Support Vector Machines. Ensemble methods such as Radom Forest, (Stochastic) AdaBoost and Kernel Factory are still very unexplored in the domain of stock price direction prediction.

In Table 1 we provide an overview of those algorithms used for predicting stock price direction in literature (we excluded single Decision Trees, Naïve Bayes, Discriminant Analysis and Genetic Algorithms because they have been superseded by newer and better methods discussed above). LR stands for Logistic Regression, NN stands for Neural Networks, KN stands for K-nearest neighbors, SVM stands for Support Vector Machines, RF stands for Random Forest, AB stands for AdaBoost and KF stands for Kernel Factory. It is clear from Table 1 that our study is the first to include all seven algorithms in one benchmark. This is important if we want to find, globally, the best algorithm. Using suboptimal algorithms may hinder scientific progress in that important patterns in the data might be missed.

In our study we will benchmark ensemble methods against single classifier models. The ensemble methods mentioned above all use a set of individually trained classifiers as base classifiers. We believe that the ensemble methods will outperform the individual classification models because they have proven to be very successful in several other domains such as face recognition (Tan, Chen, Zhou, & Zhang, 2005), gene selection (Diaz-Uriarte & de Andres, 2006), protein structural class prediction (Ballings & Van den Poel, 2015) and credit scoring (Paleologo, Elisseeff, & Antonini, 2010). In stock price direction prediction literature both Support Vector Machines (SVM) and Random Forest (RF) have proven to be top performers (Kumar & Thenmozhi, 2006; Patel et al., 2015). However, there is no consensus on which algorithm is best with SVM outperforming RF in Kumar and Thenmozhi (2006) and vice versa in Patel et al. (2015). AdaBoost has also been shown to perform well, albeit not as well as Random Forest (Rodriguez & Rodriguez 2004). In an effort to help provide clarity in which algorithm is best, this study will benchmark SVM, AB, RF and four other algorithms.

Table 1
Algorithms for stock price direction prediction used in literature.

	Prediction method						
	LR	NN	KN	SVM	AB	RF	KF
Schöneburg (1990)		x					
Bessembinder and Chan (1995)	x						
Brownstone (1996)	x	x					
Saad, Prokhorov, and Wunsch (1996)		x					
Kim and Chun (1998)		x					
Saad, Prokhorov, and Wunsch (1998)		x					
Kim and Han (2000)		x					
Kuo et al. (2001)		x					
Kim (2003)		x		x			
Kim and Lee (2004)		x					
Rodriguez and Rodriguez (2004)	x	x			x	x	
Huang et al. (2005)		x		x			
Kumar and Thenmozhi (2006)	x	x		x		x	
Lunga and Marwala (2006)					x		
Wu et al. (2006)							
Wang and Chan (2007)	x						
Huang et al. (2008)	x	x	x	x			
Senol and Ozturan (2008)	x	x					
Lai, Fan, and Chang (2009)							
Lee (2009)		x		x			
Ou and Wang (2009)	x	x	x	x			
Kara, Boyacioglu and Baykan (2011)		x		x			
Wei and Cheng (2011)		x					
Subha and Nambi (2012)	x		x				
Lin, Guo, and Hu (2013)				x			
De Oliveira, Nobre, and Zárate (2013)		x					
Chen, Chen, Fan, and Huang (2013)		x					
Rechenthin et al. (2013)			x	x	x		
Ji, Che, and Zong (2014)		x					
Bisoi and Dash (2014)		x					
Zikowski (2015)				x			
Hafezi, Shahrabi, and Hadavandi (2015)		x					
Patel et al. (2015)		x		x		x	
This study	x	x	x	x	x	x	x

3. Methodology

3.1. Data and variables

For this study we gathered yearly data from 5767 publicly listed European companies. The data cover a wide range of industries (see Table 2). Data were selected from the Amadeus Database from "bureau van Dijk". Amadeus is a database of companies' financial information across Europe.

To make our results generalizable we computed the most important predictors in extant literature. Kumar and Thenmozhi (2006) stated that earnings yield, cash flow yield, book-to-market ratio, and size are the major predictors of stock returns. Kim and Chun (1998) on the other hand use the stock price index and turnover by volume and price/earnings ratio. Wu et al. (2006) make use of money supply and inflation rate.

Besides the above stated variables in literature, we also introduced other important financial indicators that improve the predictive value of our model. These include liquidity indicators (current ratio, collection period of receivables), solvency indicators (solvency ratio, gearing ratio) and profitability indicators (ROE, ROCE, ROA). In addition information of the balance sheet (total assets, debtors, capital, long term debt) and the profit and loss statement (sales, depreciation, taxation, interest paid) is added. General economy features (public debt, GDP, unemployment, trade balance) are also included. The table in Appendix A provides a full list of all the variables in this study along with descriptions and formulas.

The goal of this study is to predict one year ahead whether the stock price will go up by a predetermined amount. In total, for year 2009 we collected 81 variables (including the market price of 2009). For year 2010 we only extracted the market price of 2010 from the database. If the stock price of 2010 went up by 25% relative to the 2009 price we coded the stock price direction as going up, otherwise we coded the direction as down. In Appendix B we provide the results of the sensitivity analysis for two other thresholds: 15% and 35%. The results largely coincide.

Using a threshold of 15%, 25% and 35% resulted in respectively 40.6%, 32.4% and 25.6% of cases being classified as a positive. Because some analytical techniques are more sensitive to class imbalance than others we over-sampled the positive class to obtain a perfectly balanced sample. We chose over-sampling of the positive class instead of under-sampling of the negative class because the former ensures that no valuable information is discarded. Other more complicated sampling methods have been shown not to outperform over-sampling (Burez & Van den Poel 2009). Benchmarking algorithms on a balanced sample promotes fairness.

Table 2
Number of companies per industry.

Industry	Companies
Agriculture, forestry & fishing	3.3%
Mining and quarrying	3.0%
Manufacturing	26.4%
Electricity, gas, steam and air conditioning supply	2.3%
Water supply, sewerage, waste management & remediation activities	0.9%
Construction	6.2%
Wholesale & resale trade, repair of motor vehicles and motor cycles	9.3%
Transportation and storage	3.7%
Accommodation & food service activities	2.8%
Information & communication	6.5%
Financial & insurance activities	13.1%
Real estate activities	4.3%
Professional, scientific and technical activities	13.6%
Administrative & support service activities	2.6%
Public administration and defense, compulsory social security	0.1%
Education	0.2%
Human health and social work activities	0.5%
Arts, entertainment & recreation	0.7%
Other service activities	0.6%
TOTAL	5767

3.2. Analytical techniques

In this section we will describe the seven techniques used in this study: Logistic Regression (LR), Neural Networks (NN), K-nearest neighbors (KN), Support Vector Machines (SVM), Random Forest (RF), AdaBoost (AB) and Kernel Factory (KF). The first four are single classifiers and the last three are ensemble methods. All analyses are performed using *R* (R Core Team, 2013) version 3.1.3.

3.2.1. Single classifiers

3.2.1.1. Logistic Regression. We use a regularized, also called penalized, approach to Logistic Regression in order to avoid overfitting. The technique sacrifices a little bias to reduce the variance of the predicted values and hence improves overall prediction performance (Tibshirani, 1996). The lasso technique stands for least absolute shrinkage and selection operator (Tibshirani, 1996) and imposes a bound on the sum of the absolute values of the coefficients shrinking coefficients towards zero (Guisan, Edwards, & Hastie, 2002). This is done by adding a penalty term to the negative binomial log-likelihood in the objective function (Friedman et al., 2010):

$$\underset{(\beta_0,\beta)\in\mathbb{R}^{p+1}}{\operatorname{argmin}} -\left[\frac{1}{N}\sum_{i=1}^{N} y_i\cdot(\beta_0+x_i^T\beta)-\log\left(1+e^{(\beta_0+x_i^T\beta)}\right)\right]+\lambda\sum_{j=1}^{p}|\beta_j|$$

with N the number of instances and p the number of predictors. The equation clearly shows that in order to minimize the objective function coefficients are required to shrink. The parameter λ determines the amount of shrinkage. Higher values of λ will result in smaller coefficients. We cross-validated the shrinkage parameter by optimizing the AUC on a holdout set of 50% of the training set. Once we determined the optimal value of λ we re-estimated the Logistic Regression model on the full training set to ensure a fair comparison with the other algorithms. We used the *glmnet* *R*-package by Friedman et al. (2010, 2013). We set the α parameter to 1 to obtain the lasso method and we let the function compute the sequence of λ by setting *nlambda* to 100 (the default).

3.2.1.2. Neural Networks. We used a feed-forward artificial neural network optimized by BFGS. This method is more efficient, reliable and convenient than backpropagation. We use one layer of hidden neurons. This is generally sufficient for classifying most data sets (Dreiseitl & Ohno-Machado, 2002). We use the logistic activation function and re-scale the numerical predictors to $[-1,1]$ by subtracting the midrange, defined as (max + min)/2, from each column and dividing by the range/2 of each column. The binary variables are coded as $\{-1,1\}$. Scaling of the data is necessary to avoid local optima. The underlying reason of why scaling avoids local optima can be traced back to the initialization phase. The classification boundary (i.e., hyperplane) is defined by the points where the input is zero (Sarle, 1997). Similar to the coefficient and intercept in linear regression, the weights in a Neural Network determine the orientation of the hyperplane, and the bias determines the distance of the hyperplane from the origin. If the bias terms are small, then the initial classification boundaries will pass fairly through the origin. If the data are not centered at the origin, many of the hyperplanes will not cut the data and hence separate the two

classes (Sarle, 1997). Those that do pass through the data will only provide a limited range of directions. This will increase the likelihood of obtaining local optima. Hence it is important to scale the inputs to include the origin such as $[-1,1]$.

Scaling is also required to overcome numerical problems and obtain training efficiency. Scaling avoids the limits of floating-point representations. When we have a very large input, weights will be very small and small changes cannot be represented. In addition the logistic activation function saturates for large inputs making the algorithm adjust its weight slowly during training. Scaling therefore often accelerates training. To avoid saturation the weights are desired to be small in the random initialization. The magnitude of the weights depends on the scale of the inputs. Therefore scaling also removes the scale dependence of the initial weights.

We used the *nnet R*- package (Ripley, 2013; Venables & Ripley, 2002). The network weights at the start of the iterative procedure are chosen at random (Ripley, 1996, pg. 154). As recommended by Spackman (1991) and Ripley (1996, pg. 149) the *entropy* parameter is set to use the maximum conditional likelihood. The *rang* parameter, controlling the range of the initial random weights parameter was left at the default of 0.5. To avoid overfitting we used weight decay (Dreiseitl & Ohno-Machado, 2002) and therefore the maximum number of weights (*MaxNWts*) and the number of iterations (*maxit*) were set to very large values (5000) in order not to run into a situation of early stopping. Finally, we determined the weight decay factor and the number of nodes in the hidden layer by performing a grid search (Dreiseitl & Ohno-Machado, 2002). We tried all combinations of decay = {0.001, 0.01, 0.1} (Ripley, 1996, pg. 163), and size = [1, 2,...,20] (Ripley, 1996, pg. 170) and selected the optimal combination.

3.2.1.3. K-Nearest Neighbor. We used the k-d tree (Bentley, 1975) implementation of the K-Nearest Neighbors algorithm. An important parameter to determine is the *K*. This represents the number of closest neighbors that will be taken into account when scoring the unknown sample. The unknown sample will receive a score that corresponds to the proportion of positives of the *K* samples. We cross-validated the *K*-parameter by trying all values of $K = \{1,2,3,\ldots,150\}$. We used the *FNN R*-package (Beygelzimer et al. 2013).

3.2.1.4. Support Vector Machines. We used the most commonly used kernels: the linear, polynomial, radial basis (RBF), and sigmoid kernel (Ballings & Van den Poel, 2013a). The RBF kernel requires a width parameter λ of the Gaussian function (Ben-Hur & Weston, 2010). The penalty parameter *C*, also called the cost or soft margin constant, specifies the trade-off between the size of the margin and hyperplane violations. The higher the *C*, the more it costs to have examples in the soft margin and hence the smaller the margin. The polynomial kernel requires a choice of degree *d*. The linear kernel function only requires setting *C*. To determine the optimal parameter values we follow Hsu et al. (2010) by performing a grid search on $C = [2^{-5}, 2^{-4}, \ldots, 2^{15}]$, $\lambda = [2^{-15}, 2^{-13}, \ldots, 2^{3}]$ and $d = \{2, 3\}$ to identify the best combination. Support Vector Machines are implemented through Meyer et al.'s (2012) *e1071 R*-package using the *svm* function.

3.2.2. Ensemble methods

Ensembles methods solve problems that are (1) statistical, (2) computational, and (3) representational in nature (Dietterich, 2000) by averaging models. Many different classifiers can be learned from specific combinations of data, representations, objective functions and optimization methods. The set of all possible classifiers that might be learned is called the hypothesis space $\mathcal{H}$. An algorithm searches $\mathcal{H}$ to identify the best possible hypothesis $h \epsilon \mathcal{H}$. Because in reality one only has limited data it is well possible that the algorithm finds different hypotheses of equal overall accuracy. If however, in some region of the input domain, some hypotheses outperform the others, averaging reduces the risk of selecting the wrong hypothesis in a specific region. This is what Dietterich (2000) calls the statistical problem.

The second problem that ensembles can alleviate is what Dietterich (2000) calls the computational problem. Algorithms employ optimization methods to perform local searches. These searches may get stuck in local optima. By averaging different hypotheses, obtained by different search initializations, the true hypothesis f may be better approximated than the individual hypotheses.

Finally if $f \notin \mathcal{H}$, averaging can expand the space of representable functions. For example, a linear algorithm cannot learn a nonlinear boundary, but a combination of linear algorithms can. This is why ensemble methods are said to use a divide and conquer approach; each individual learns a simpler and smaller part of the problem. This is what Dietterich (2000) calls a representational problem.

An algorithm that suffers from the statistical or computational problem has problems of high variance, while an algorithm that suffers from the representational problem has high bias. Hence, ensembles are attractive because they may reduce both bias and variance (Zhou, 2012, p67).

3.2.2.1. Random Forest. Random Forest builds an ensemble of trees to improve upon the limited robustness and suboptimal performance of decision trees (Dudoit et al.,2002). Individual trees are built on a bootstrap sample using a version of Binary Recursive Partitioning (BRP) (Breiman, Friedman, Stone, & Olshen 1984). The BRP algorithm initializes by making a random subset of candidate variables and evaluating all possible splits of all candidate variables (Breiman, 2001). The best split is subsequently used to create a binary partitioning. This process of selecting a subset of variables, evaluating all splits and creating a binary partitioning is repeated recursively within each subsequent partition and stops when the partition size equals 1. There are only two parameters in Random Forest: the number of trees in the ensemble and the number of variables to try at each split. As recommended by Breiman (2001) we use a large number of trees (500) and set the number of variables to the square root of the total number of predictors. We employed the *randomForest R*- package by Liaw and Wiener (2002).

3.2.2.2. AdaBoost. Deterministic AdaBoost works by sequentially applying a learning algorithm to reweighted versions of the training data (Freund & Schapire, 1996). In each iteration the instances that were misclassified in the previous iteration are assigned more weight. The final model is a linear combination of the models obtained in the different iterations (Friedman, Hastie, & Tibshirani, 2000). We use one of the most recent boosting variants: stochastic boosting (Friedman, 2002). It improves on the original algorithms by incorporating randomness as an integral part of the procedure (Friedman, 2001). Specifically, stochastic AdaBoost draws bootstrap samples from the training sample, with the probability of an instance being selected proportional to the current weight (Friedman, 2002). Two important parameters are the number of iterations and the number of terminal nodes in the team members. As recommended by Friedman (2001) we set the maximum number of nodes to 8 by setting the maximum depth of the trees to 3. In addition we set the number of iterations to 500. We used the *ada R*- package by Culp, Johnson, and Michailidis (2012).

3.2.2.3. Kernel Factory. Kernel Factory randomly divides the training data into a number of mutually exclusive partitions defined

by a row and column parameter. Each partition forms a separate input space and is subsequently transformed by a kernel into a kernel matrix *K*. Next each *K is* employed to train a Random Forest. These Random Forest models are then deployed on a validation set. The scores are then used by a Genetic Algorithm to optimize the weights for subsequent combination through a weighted average. Ballings and Van den Poel (2013a) recommend the burn method for Kernel Factory. This method automatically tries the polynomial, linear and radial basis kernel function on the first partition and selects the best kernel function for use on all other partitions. Furthermore, we use the recommended values of 1 and *int(log10(N)),* with N the number of instances, for respectively the number of column partitions and row partitions (Ballings & Van den Poel, 2013b). Kernel Factory is implemented through the *kernelFactory R-* package (Ballings & Van den Poel, 2013b).

3.3. Model evaluation criteria

The area under the receiver operating characteristic curve (AUC) is seen as an adequate and accurate classifier performance measure (Provost, Fawcett, & Kohavi, 1997). AUC can take a value from 0.5 to 1. A value of 0.5 means that the predictions are not better than random, while a value of 1 means that the predictions are perfect. The advantage of AUC over other performance measures (such as Percent Correctly Classified (PCC)) is that AUC includes all cut-off values (Ballings & Van den Poel, 2013). AUC is defined as follows:

$$AUC = \int_0^1 \frac{TP}{(TP+FN)} d\frac{FP}{(FP+TN)} = \int_0^1 \frac{TP}{P} d\frac{FP}{N}$$

with TP: True Positives, TN: True Negatives, P: Positives (event), N: Negatives (non-event). All AUCs reported in this study are medians of five times twofold cross validation.

In order to determine whether there are any important differences across the output domain we also report the five times twofold cross- validated receiver operating characteristic (ROC) curves (see Section 3.4. for more information about cross- validation). We use threshold averaging. This method of averaging chooses a uniformly distributed set of thresholds among the sorted set of all thresholds of the ROC curves in the sample. For each of these thresholds it generates the ROC points for each of the ROC curves and takes the mean true positive and false positive rates for subsequent plotting.

3.4. Cross-validation

Cross-validation is widely used for model selection because of its simplicity and universality (Arlot & Celisse, 2010). We follow Demšar's (2006) recommendation and use five times twofold cross-validation (5x2f cv). This method of cross-validation splits the data randomly into two equal parts. One part is used to train the algorithm and the other part to evaluate the algorithm, and vice versa. This process is then repeated five times. This results in 10 performance values.

To determine whether the differences between the algorithms in terms of AUCs are significant we use the Friedman test (Friedman, 1937; Friedman, 1940). The Friedman test is a non-parametric equivalent of the repeated-measures ANOVA (Demšar, 2006). It ranks the algorithms for each cross-validation fold separately, with the top algorithm receiving the rank of 1, the second best receiving the rank of 2,. . ., and the worst performing algorithm receiving rank equal to the number of algorithms (in our case 7). Average ranks are assigned in case of ties. The Friedman statistic is defined as:

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right]$$

where N is the number of folds (10 in our case), k is the number of algorithms (7 in our case), and R_j is the average rank of the jth of k algorithms. The average rank is defined as $Rj = \frac{1}{N}\sum_i^N r_i^j$ where r_i^j is the rank of the jth of k algorithms on the ith of N folds.

When the null hypothesis (all algorithms are equivalent) of the Friedman test is rejected we perform the Nemenyi post hoc test to determine which algorithms are significantly different (Nemenyi, 1963). Two classifiers are significantly different if the corresponding average ranks differ by as least the critical difference

$$CD = q_\alpha\sqrt{\frac{k(k+1)}{6N}} = 2.949\sqrt{\frac{7(7+1)}{6.10}} = 2.849005 \quad (1)$$

where the critical values q_α are based on the Studentized range statistic divided by $\sqrt{2}$, N is the number of folds and k is the number

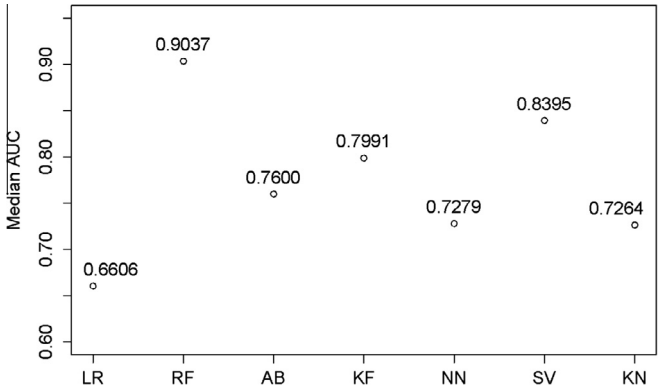

Fig. 1. 5x2fcv median AUC per algorithm.

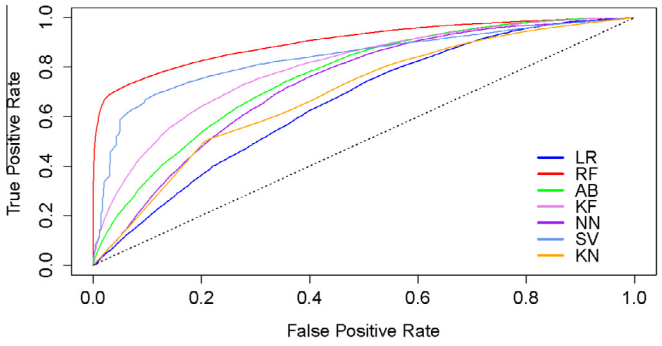

Fig. 2. 5x2fcv averaged ROC curve per algorithm.

Table 3
Mean ranking of the folds for AUC.

Prediction method						
LR	NN	KN	SVM	RF	AB	KF
6.8	5.4	5.8	2.0	1.0	4.0	3.0

Table 4
The interquartile range of the ten folds for AUC.

Prediction method						
LR	NN	KN	SVM	RF	AB	KF
0.0082	0.0187	0.0489	0.0116	0.0061	0.0134	0.0076

Table 5
Profit implications.

	2009	2010	2011	2012	2013	2014	
Price Barco stock	15.17	28.49	48.27	38.76	55.48	56.53	
Profits							
Prediction rate	2009	2010	2011	2012	2013	2014	
70%		9.324	13.846	−2.853	11.704	0.735	
71%		9.457	14.044	−2.758	11.871	0.746	
		0.133	0.198	0.095	0.167	0.0111	0.604
Discounted profits							
Discount rate	2009	2010	2011	2012	2013	2014	
5%		0.133	0.188	0.086	0.144	0.009	0.561

of algorithms to be compared (Demšar, 2006). If the difference between the mean rankings of 2 algorithms is bigger than the critical difference (CD) of 2.849005 (this is the critical difference for a p-value of 0.05, 7 classifiers and 10 folds) then the performance of these algorithms differs significantly.

4. Results and discussion

Fig. 1 presents the median AUC of the five times twofold cross-validation. LR stands for Logistic Regression, NN stands for Neural Networks, KN stands for K-nearest neighbors, SVM stands for Support Vector Machines, RF stands for Random Forest, AB stands for AdaBoost and KF stands for Kernel Factory. AUC values are all above 0.5 stating that the evaluated prediction methods are better than random. From Fig. 1 it is clear that Random Forest is the top predictor followed at a distance by SVM, KF, AB, NN, KN and LR. Our results are in accordance with Patel et al. (2015), who also found that RF outperforms SVM. In contrast Kumar and Thenmozhi (2006) found that SVM outperforms RF. However the difference in Kumar and Thenmozhi (2006) was very small (1.04%-points in terms of Hit ratio) whereas in this study the difference is very large (0.0642 in terms of AUC). Fig. 2 confirms the generalizability of the results. Across the entire output domain, RF is the best algorithm.

The Friedman test indicates that the differences in AUC are significant with $\chi^2(6) = 57.9429, p < 0.001$. Because the null hypothesis of the Friedman rank sum test is rejected, we perform the Nemenyi post hoc test to compare all algorithms to each other. To determine which specific algorithms are significantly different we first calculate the average rankings of the AUCs in Table 3 and then compare which differences are greater than the critical difference of 2.849005. First of all RF significantly outperforms all the other classifiers except SVM. Furthermore AB is not significantly better than any other classifier. Finally KF only significantly outperforms LR (See Table 4).

Finally, to determine the stability of the results we calculate the interquartile range (IQR) in Table 3 by taking the difference of the 75th and 25th percentile. The bigger this difference, the bigger the deviation is and the lower the stability. The dominance of RF is again confirmed because it has the smallest IQR of all the algorithms.

5. Conclusions

This study set out to benchmark the performance of ensemble methods (Random Forest, Adaboost and Kernel Factory) against single classifier models (Neural Networks, Logistic Regression, Support Vector Machines and K-Nearest Neighbors) in predicting stock price direction. To the best of our knowledge this is the first study to include this set of algorithms in a stock price direction prediction application. Being able to predict the stock price direction more accurately is beneficial to companies specialized in predicting financial quotes, investment banks and all people dealing with stocks. While ensembles have proven to be top performers in various other domains, only a small number of benchmarks in the domain of this study include ensembles. We found that Random Forest is the top performer, followed by Support Vector Machines, Kernel Factory and AdaBoost. In other words, the three ensembles that we included in our benchmark are ranked as the top four algorithms. The results clearly suggest that novel studies in the domain of stock price direction prediction should include ensembles in their sets of algorithms, which, as Table 1 indicates, is evidently not the case.

Hence we recommend Random Forest to predict stock price direction. It has the highest median AUC and its performance is significantly better than all but one other classifiers. Furthermore, other ensemble methods appear to be good alternatives for single classifier models because they are ranked very high. In Table 5, we underscore the importance of improving stock price direction prediction.

As illustration we used the stock quotes of Barco over the last 5 years and took a discount rate of 5% into account. Consider an investor making a hold or sell decision each year. The decision is valid for one year. When the investor uses a model with an accuracy of 70% to predict the stock price direction he has 70% chance to estimate the correct direction and thus only 70% chance on the profits incurred on the stock. When the investor is able to increase the prediction rate with only 1%-point, there are already financial benefits attached. On the stock of Barco over 5 years the investor retains €0.561 per stock. This equals a 1,429% increase in profit.

6. Limitations and future research

As we based our research on data from the Amadeus database, our data are limited to information from European companies. Further research might incorporate worldwide data. According to Kumar and Thenmozhi (2006), it might be useful to benchmark algorithms per continent. This because some techniques tend to perform better in some parts of the world then others, for example in BRIC countries.

Our focus was on predicting the direction of the stock price. Another focus might be to try to predict the exact stock prices (Kaboudan, 2000; Pai & Lin, 2005) using ensemble methods. In addition Wang (2002) attempts to predict the stock price immediately at any given time, while Oh and Kim (2002) and Qian and Rasheed (2007) focus on predicting stock market indices. Finally Gençay and Qi (2001) used Neural Networks to predict the pricing of derivative securities.

In this study we predict one-year ahead stock price directions. Future research might also investigate intra-daily (Rechenthin & Street, 2013), daily (Rechenthin, Street, & Srinivasan, 2013), weekly or monthly predictions. We focus on long-term investment decisions since we only predict one year ahead. That is why our results are not generalizable to High Frequency Trading (HFT). This way of

trading uses computer-aided algorithms that analyze the market and/or an asset. Because trading volumes are huge and positions are held only for a very short period of time (only 1 minute in the paper of Manahov et al. (2014)) high frequency traders are able to make profit with only a small change in stock prices. Long-term investors, however, apply a buy and hold strategy. For this strategy small changes in quotes are less important since their volumes are much smaller and trades are repeated less often. The difference between these two strategies can be subject to future research.

Appendix A.

Predictors used in this study

Variable type	Variable name	Description/definition	Formula
General info	Number of employees	Indication of the firm size	
	Added value	Amount added to the value of a product or service	
	Working capital	Indicator of the firm's efficiency and short-term financial health	Current assets-current liabilities
	Stock price 2009	The closing stock price of 2009	
Assets	Fixed assets	Long-term tangible piece of property a firm owns.	
	Other fixed assets	Sum of fixed assets not detailed by individual class on balance sheet	
	Current assets	Assets that are expected to be converted within one year	
	Net current assets	How much capital being generated or used by day-to day activities	Current assets – current liabilities
	Other current assets	Current assets not including cash, securities, receivables, inventory,..	
	Creditors	Claims against customers (settled within 1 year)	
	Cash	Fair value of cash and cash equivalents	
	Total assets	Resources a firm owns to create future economic benefits	
Liabilities	Shareholders funds	Balance of share capital, all profits retained and reserves	
	Other shareholders funds		
	Total shareholders funds	Balance sheet value of the shareholders' interest	Total assets – total liabilities
	Capital	Financial resources available for use	
	Market capitalization	Market value the outstanding shares	
	Stock	Equity stake of the owners	
	Current liabilities	Trade payables and revenues to be earned in next years	
	Other current liabilities	Claims against the company held by others than suppliers	
	Provision	Present liability of an entity to another entity	
	Debtors	Suppliers' claims against the company	
	Non-current liabilities	Long-term financial obligations that are not due within present year	
	Other non-current liabilities		
	Long-Term debt	Debt that does not need to be repaid in one year	
	Loans	Debt provided by one entity to another entity at an interest rate	
Income	Depreciation & amortization	Non-cash expense of a tangible/intangible investment over time	
Statement	Operating revenue	Revenue generated from operating activities	

Appendix A. (*continued*)

Predictors used in this study			
Variable type	Variable name	Description/definition	Formula
	Sales	Revenues generated through the use of operating assets	
	EBIT	Earnings Before Interests and Taxes, operating profit and loss	
	Costs of employees	Amount paid for all employee wages and benefits	
	Average costs of employees	Average amount paid for all employee wages and benefits	
	EBITDA	Earnings Before Interests, Taxes, Depreciation and Amortization	
	Financial revenue	Revenue from financial activities	
	Financial expenses	Expenses from financial activities	
	Interest paid	Interest accrued on financial liabilities	
	Financial profit and loss	Profit or loss resulting from financing activities	Financial revenue-financial expenses
	Cash flow	Shows the amount of cash generated and used in a given period	
	Profit and loss before tax	Profitability measure looking at profit before corporate income tax	
	Taxation	Amount a company expects to pay for taxes	
	Profit and loss after tax	Profitability of a company after accounting for all costs	
	Net income	Profit and loss for the period	
Ratio's	Profit margin	Percentage of selling price that turned into profit	Profit/sales
	Net assets turnover	Efficiency with which net assets are used	Sales/net assets
	EBIT margin	Indication of operating performance	EBIT/sales
	EBITDA margin	Measurement of a company's operating profitability	EBITDA/sales
	Mrkt cap/shareholders funds	Cf. infra	Market cap/shareholders funds
	Inventory turnover	Indication of how well a company converts stock into revenues	Cost of goods sold/ average inventory
	Current ratio	Ability to pay short-term obligations	Current assets/current liabilities
	Solvency ratio	Ability to meet debt and other obligations	Total debt/total assets
	Liquidity ratio	Ability to repay short-term creditors out of total cash	Liquid assets/short-term liabilities
	Shareholders liquidity ratio		Shareholders funds/non current liabilities
	Collection period	Amount of time it takes to receive payments owed	(days∗accounts receivable)/credit sales
	Cash flow yield	Return evaluation ratio	Cash flow per share/Stock price 2009
	ROE (using PL before tax)	Return On Equity: Indicator of a firm's performance	PL before tax/shareholders' equity
	ROCE(using PL before tax)	Return On Capital Employed: measures the firm's profitability	PL before tax/ capital employed
	ROA (using PL before tax)	Return On Assets: profit for each euro of assets invested	PL before tax/total assets
	ROE (using net income)	Cf. Infra	Net income/shareholders' equity
	ROCE (using net income)	Cf. infra	Net income/capital employed
	ROA (using net income)	Cf. infra	Net income/total assets
	ROI	Return On Investment: Measures the efficiency	PL for the period/shareholders funds
	Costs of empl./operating rev.	Cf. infra	Costs of employees/operating revenue
	Price-earnings ratio	Valuation ratio	Stock price per share/Earnings per share
	Price cashflow ratio	Valuation ratio	Stock price per share/Operating cash flow per share

(*continued on next page*)

Appendix A. (*continued*)

Predictors used in this study			
Variable type	Variable name	Description/definition	Formula
	Price book value ratio	Valuation ratio	Stock price/(total assets-intangible assets and liabilities)
	Credit period	The period of time a firm grants credit to its customers	(accounts payable∗number of working days)/credit purchases
	Interest coverage	Indication of how easily a company can pay interest on outstanding debt	(Net profit + interest and tax expense)/interest expense
	Gearing	Level of debt related to equity capital (in %); measure of financial leverage	(Long term debt + short term debt)/ shareholders' equity
	Earnings per share	Indicator of profitability	(Net income – dividends on preferred stock)/average outstanding shares
	Earnings yield	Percentage of each dollar invested in the stock that was earned by the company	EPS for the most recent 12 month period/ current stock price
	Book value per share	Level of safety of each individual stock after all debts are paid	(Total shareholder equity-preferred equity)/total shares outstanding
	Cash flow per share	Indicator of financial strength	(Operating cash flow-preferred dividends)/Common shares outstanding
Economic info	International reserves	Reserve funds that can be passed between the central banks.	
	Public debt	Debt owed by a central government	
	Budget balance	Overall difference between government revenues and spending	Govern. revenues – govern. spending
	Trade balance	Difference between the monetary value of exports and imports	Exports - imports
	Unemployment	Number of people without work and actively seeking work	Number of unemployed people/all people in the labor force
	GDP real growth rate	Indicator of economic health; rate of change that a nation's gross domestic product (GDP) experiences from one year to another.	(Real GDP year X – real GDP year (X-1))/ Real GDP year (X-1)
	Inflation	General increase in prices and fall in purchasing value of money	((Consumer price index value year X – consumer price index value year (X-1)) / Consumer price index value year (X-1)) ∗ 100

Appendix B.

The results for the two alternative thresholds largely coincide. For a threshold of 15 percent the 5x2fcv median AUCs are (from high to low): RF = 0.8667, SV = 0.8006, KF = 0.7688, AB = 0.7404, NN = 0.7088, LR = 0.6576, KN = 0.6494. The top five algorithms receive the same ranking as for the 25 percent threshold. LR switches from the last position to the one but last position.

For a threshold of 35 percent the 5x2fcv median AUCs are (from high to low): RF = 0.9287, SV = 0.8958, KF = 0.8203, KN = 0.7608, AB = 0.7578, NN = 0.7394, LR = 0.6468. The top three algorithms receive the same ranking as in the 25 percent case. In the tail of the ranking KN gains two positions, and AB and NN lose one position. LR keeps its original position.

In conclusion, the sensitivity analysis clearly indicates that the results are not sensitive to the chosen threshold. The results and conclusions are very similar across thresholds. For all test thresholds the top three performers are RF, SVM and KF. There are only minor changes in rakings in the bottom performers.

References

Al-Hmouz, R., Pedrycz, W., & Balamash, A. (2015). Description and prediction of time series: A general framework of granular computing. *Expert Systems with Applications, 42*, 4830–4839. http://dx.doi.org/10.1016/j.eswa.2015.01.060.

Arlot, S., & Celisse, A. (2010). A survey of cross-validation procedures for model selection. *Statistics Surveys, 4*, 40–79. http://dx.doi.org/10.1214/09-SS054.

Ballings, M., & Van den Poel, D. (2012). Customer event history for churn prediction: How long is long enough? *Expert Systems with Applications, 39*, 13517–13522.

Ballings, M., & Van den Poel, D. (2013b). R package kernelFactory: An ensemble of kernel machines. Available at: http://cran.r-project.org/web/packages/kernelFactory/index.html.

Ballings, M., & Van den Poel, D. (2013a). Kernel factory: An ensemble of kernel machines. *Expert Systems with Applications, 40*(8), 2904–2913.

Ballings, M., & Van den Poel, D. (2015). CRM in social media: Predicting increases in facebook usage frequency. *European Journal of Operational Research, 244*, 248–260. http://dx.doi.org/10.1016/j.ejor.2015.01.001.

Barak, S., & Modarres, M. (2015). Developing an approach to evaluate stocks by forecasting effective features with data mining methods. *Expert Systems with Applications., 42*, 1325–1339. http://dx.doi.org/10.1016/j.eswa.2014.09.026.

Ben-Hur, A., & Weston, J. (2010). *A user's guide to support vector machines*. *Methods in Molecular Biology*. Department of Computer Science Colorado State University (pp. 223–239). Department of Computer Science Colorado State University.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM, 18*(9), 509–517.

Bessembinder, H., & Chan, K. (1995). The profitability of technical trading rules in the asian stock markets. *Pacific-Basin Finance Journal, 3*(2–3), 257–284. http://dx.doi.org/10.1016/0927-538X(95)00002-3.

Beygelzimer, A., Kakadet, S., Langford, J., Arya, S., Mount, D., & Shengqiao, L. (2013). R package: FNN, fast nearest neighbor search algorithms and applications. Available at: http://cran.r-project.org/web/packages/FNN/index.html.

Bisoi, R., & Dash, P. K. (2014). A hybrid evolutionary dynamic neural network for stock market trend analysis and prediction using unscented Kalman filter. *Applied Soft Computing, 19*, 41–56. http://dx.doi.org/10.1016/j.asoc.2014.01.039.

Booth, A., Gerding, E., & McGroarty, F. (2014). Automated trading with performance weighted random forests and seasonality. *Expert Systems with Applications, 41*, 3651–3661. http://dx.doi.org/10.1016/j.eswa.2013.12.009.

Breiman, Leo. (2001). Random Forests. *Machine Learning, 45*(1), 5–32. http://dx.doi.org/10.1023/A:1010933404324.

Breiman, L., Friedman, J., Stone, C. J., & Olshen, R. A. (1984). *Classification and regression trees. Wadsworth Statistics/Probability* (1st ed.,). New York, N.Y: Chapman and Hall/CRC.

Brody, S., Navigli, R., & Lapata, M. (2006). Ensemble methods for unsupervised WSD. In *Proceedings of the 21st international conference on computational linguistics and the 44th annual meeting of the association for computational linguistics* (pp. 97–104). Berlin: Springer. http://dx.doi.org/10.3115/1220175.1220188.

Brownstone, D. (1996). Using percentage accuracy to measure neural network predictions in stock market movements. *Neurocomputing, 10*(3), 237–250. http://dx.doi.org/10.1016/0925-2312(95)00052-6.

Burez, J., & Van den Poel, D. (2009). Handling class imbalance in customer churn prediction. *Expert Systems with Applications, 36*, 4626–4636. http://dx.doi.org/10.1016/j.eswa.2008.05.027.

Chen, M.-Y., Chen, D.-R., Fan, M.-H., & Huang, T.-Y. (2013). International transmission of stock market movements: An adaptive neuro-fuzzy inference system for analysis of TAIEX forecasting. *Neural Computing and Applications, 23*, S369–S378. http://dx.doi.org/10.1007/s00521-013-1461-4.

Cheung, Y.-W., Chinn, M. D., & Pascual, A. G. (2005). Empirical exchange rate models of the nineties: Are any fit to survive? *Journal of International Money and Finance, 24*(7), 1150–1175. http://dx.doi.org/10.1016/j.jimonfin.2005.08.002.

Culp, M., Johnson, K., & Michailidis, G. (2012). R package ada: An R package for stochastic boosting. Available at: http://cran.r-project.org/web/packages/ada/index.html.

De Oliveira, F. A., Nobre, C. N., & Zárate, L. E. (2013). Applying artificial neural networks to prediction of stock price and improvement of the directional prediction index – Case study of PETR4, Petrobras, Brazil. *Expert Systems with Applications, 40*, 7596–7606. http://dx.doi.org/10.1016/j.eswa.2013.06.071.

Demšar, Janez. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research, 7*, 1–30.

Díaz-Uriarte, R., & de Andres, S. A. (2006). Gene selection and classification of microarray data using random forest. *BMC Bioinformatics 2006, 7*, 3.

Dietterich, T.G. (2000). Ensemble methods in machine learning. In: Kittler, J., Roli, F. (Eds.), Multiple classifier systems (pp. 1–15).

Dreiseitl, S., & Ohno-Machado, L. (2002). Logistic regression and artificial neural network classification models: A methodology review. *Journal of Biomedical Informatics, 35*(5–6), 352–359.

Dudoit, S., Fridlyand, J., & Speed, T. P. (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association, 97*, 77–87.

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). The KDD process for extracting useful knowledge from volumes of data. *Communications ACM, 39*(11), 27–34. http://dx.doi.org/10.1145/240455.240464.

Freund, Y., & Schapire, R. (1996). Experiments with a new boosting algorithm. In *machine learning. proceedings of the thirteenth international conference (ICML '96)* (pp. 148–156). Bari, Italy.

Freund, Y., & Schapire, Robert E. (1995). A Desicion-theoretic generalization of on-line learning and an application to boosting. In P. Vitányi (Ed.), *Computational Learning Theory. Lecture Notes in Computer Science* (Vol. 904, pp. 23–37). Berlin Heidelberg: Springer.

Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association, 32*, 675–701. http://dx.doi.org/10.2307/2279372.

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics, 11*, 86–92. http://dx.doi.org/10.2307/2235971.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics, 29*(5), 1189–1232.

Friedman, J. H. (2002a). Stochastic gradient boosting. *Computational Statistics & Data Analysis, 38*, 367–378.

Friedman, Jerome H. (2002b). Stochastic gradient boosting. *Computational Statistics & Data Analysis, 38*(4), 367–378. http://dx.doi.org/10.1016/S0167-9473(01)00065-2.

Friedman, J., Hastie, T., & Tibshirani, R. (2013). R package glmnet: Lasso and elastic-net regularized generalized linear models. Available at: http://cran.r-project.org/web/packages/glmnet/index.html.

Friedman, J., Hastie, T., & Tibshirani, R. (2000). Additive logistic regression: A statistical view of boosting. *Annals of Statistics, 28*, 337–374. http://dx.doi.org/10.1214/aos/1016218223.

Friedman, J., Hastie, T., & Tibshirani, R. (2010a). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software, 33*, 1–22.

Friedman, J., Hastie, T., & Tibshirani, R. (2010b). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software, 33*(1), 1–22.

Gençay, R., & Qi, Min. (2001). Pricing and hedging derivative securities with neural networks: Bayesian regularization, early stopping, and bagging. *IEEE Transactions on Neural Networks, 12*(4), 726–734. http://dx.doi.org/10.1109/72.935086.

Guisan, A., Edwards, T. C., & Hastie, T. (2002). Generalized linear and generalized additive models in studies of species distributions: Setting the scene. *Ecological Modelling, 157*(2–3), 89–100.

Hafezi, R., Shahrabi, J., & Hadavandi, E. (2015). A bat-neural network multi-agent system (BNNMAS) for stock price prediction: Case study of DAX stock price. *Applied Soft Computing, 29*, 196–210. http://dx.doi.org/10.1016/j.asoc.2014.12.028.

Hand, D. J., Mannila, Heikki, & Smyth, Padhraic (2001). *Principles of data mining*. MIT Press.

Hellström, T., Holmströmm, K. (1998). Predictable Patterns in Stock Returns. *Technical Report Series IMa-TOM-1997-09* (August 9, 1998).

Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2010). *A practical guide to support vector classification (Technical Report)*. Department of Computer Science and Information Engineering, National Taiwan University, Taipei 106, Taiwan.

Huang, W., Nakamori, Y., & Wang, S.-Y. (2005). Forecasting stock market movement direction with support vector machine. *Computers & Operations Research, 32*(10), 2513–2522. http://dx.doi.org/10.1016/j.cor.2004.03.016.

Huang, C.-J., Yang, D.-X., & Chuang, Y.-T. (2008). Application of wrapper approach and composite classifier to the stock trend prediction. *Expert Systems with Applications, 34*(4), 2870–2878. http://dx.doi.org/10.1016/j.eswa.2007.05.035.

Ji, T., Che, W., & Zong, N. (2014). Stock market forecast based on RBF neural network. In Z. Wen & T. Li (Eds.), *Practical Applications of Intelligent Systems, Iske 2013* (pp. 955–962). Berlin, Berlin: Springer-Verlag.

Kaboudan, M. A. (2000). Genetic programming prediction of stock prices. *Computational Economics, 16*(3), 207–236. http://dx.doi.org/10.1023/A:1008768404046.

Kara, Y., Boyacioglu, M. A., & Baykan, Ö. K. (2011). Predicting direction of stock price index movement using artificial neural networks and support vector machines: The sample of the Istanbul stock exchange. *Expert Systems with Applications, 38*(5), 5311–5319. http://dx.doi.org/10.1016/j.eswa.2010.10.027.

Kim, K.-j. (2003). Financial time series forecasting using support vector machines. *Neurocomputing, 55*(1–2), 307–319. http://dx.doi.org/10.1016/S0925-2312(03)00372-2.

Kim, S. H., & Chun, S. H. (1998). Graded forecasting using an array of bipolar predictions: Application of probabilistic neural networks to a stock market index. *International Journal of Forecasting, 14*(3), 323–337. http://dx.doi.org/10.1016/S0169-2070(98)00003-X.

Kim, K.-j., & Han, I. (2000). Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index. *Expert Systems with Applications, 19*(2), 125–132. http://dx.doi.org/10.1016/S0957-4174(00)00027-0 (August 2000).

Kim, K.-j., & Lee, W. B. (2004). Stock market prediction using artificial neural networks with optimal feature transformation. *Neural Computing & Applications, 13*(3), 255–260. http://dx.doi.org/10.1007/s00521-004-0428-x.

Kumar, M., & Thenmozhi, M. (2006). *Forecasting Stock index movement: A comparison of support vector machines and random forest*. SSRN Scholarly Paper. Rochester, NY: Social Science Research Network, January 24, 2006. <http://papers.ssrn.com/abstract=876544>.

Kuo, R. J., Chen, C. H., & Hwang, Y. C. (2001). An Intelligent stock trading decision support system through integration of genetic algorithm based fuzzy neural network and artificial neural network. *Fuzzy Sets and Systems, 118*(1), 21–45. http://dx.doi.org/10.1016/S0165-0114(98)00399-6.

Lai, R. K., Fan, W.-H. H., & Chang, P.-C. (2009). Evolving and clustering fuzzy decision tree for financial time series data forecasting. *Expert Systems with Applications Part, 36*(2), 3761–3773. http://dx.doi.org/10.1016/j.eswa.2008.02.025.

Lee, M.-C. (2009). Using support vector machine with a hybrid feature selection method to the stock trend prediction. *Expert Systems with Applications, 36*(8), 10896–10904. http://dx.doi.org/10.1016/j.eswa.2009.02.038.

Leung, Mark. T., Daouk, Hazem., & Chen, An.-Sing. (2000). Forecasting stock indices: A comparison of classification and level estimation models. *International Journal of Forecasting, 16*(2), 173–190. http://dx.doi.org/10.1016/S0169-2070(99)00048-5.

Liaw, A., & Wiener, M. (2002). Classification and regression by random Forest. *R News, 2*(3), 18–22.

Lin, Y., Guo, H., & Hu, J., (2013). An SVM-based Approach for Stock Market Trend Prediction. Neural Networks (IJCNN), *The 2013 international joint conference* (August 2013).

Lo, Andrew W., Mamaysky, H., & Wang, J. (2000). Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation. *The Journal of Finance, 55*(4), 1705–1770. http://dx.doi.org/10.1111/0022-1082.00265.

Lunga, D., & Marwala, T. (2006). Online forecasting of stock market movement direction using the improved incremental algorithm. In I. King, J. Wang, L.-W. Chan, & D. Wang (Eds.), *Neural Information Processing. Lecture Notes in Computer Science* (Vol. 4234, pp. 440–449). Berlin Heidelberg: Springer. http://link.springer.com/chapter/10.1007/11893295_49.

Malkiel, Burton G. (2003). The efficient market hypothesis and its critics. *The Journal of Economic Perspectives, 17*(1), 59–82. http://dx.doi.org/10.2307/3216840.

Malkiel, B. G., & Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work*. *The Journal of Finance, 25*(2), 383–417. http://dx.doi.org/10.1111/j.1540-6261.1970.tb00518.x.

Manahov, V., Hudson, R., & Gebka, B. (2014). Does high frequency trading affect technical analysis and market efficiency? and if so, How? *Journal of International Financial Markets, Institutions and Money, 28*, 131–157. http://dx.doi.org/10.1016/j.intfin.2013.11.002.

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., & Leisch, F. (2012). R package e1071: Misc functions of the department of statistics (e1071). Available at: http://cran.r-project.org/web/packages/e1071/index.html.

Nemenyi. P. B. (1963). *Distribution-free multiple comparisons* (PhD thesis), Princeton University.

Oh, K. J., & Kim, K.-j. (2002). Analyzing stock market tick data using piecewise nonlinear model. *Expert Systems with Applications, 22*(3), 249–255. http://dx.doi.org/10.1016/S0957-4174(01)00058-6.

Ou, P., & Wang, H. (2009). Prediction of stock market index movement by ten data mining techniques. *Modern Applied Science, 3*(12), P28. http://dx.doi.org/10.5539/mas.v3n12P28.

Pai, P.-F., & Lin, C.-S. (2005). A hybrid ARIMA and support vector machines model in stock price forecasting. *Omega, 33*(6), 497–505. http://dx.doi.org/10.1016/j.omega.2004.07.024.

Paleologo, G., Elisseeff, A., & Antonini, G. (2010). Subagging for credit scoring models. *European Journal of Operational Research, 201*(2), 490–499.

Patel, J., Shah, S., Thakkar, P., & Kotecha, K. (2015). Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques. *Expert Systems with Applications, 42*, 259–268. http://dx.doi.org/10.1016/j.eswa.2014.07.040.

Pesaran, M. Hashem, & Timmermann, Allan (1995). Predictability of stock returns: Robustness and economic significance. *The Journal of Finance, 50*(4), 1201–1228. http://dx.doi.org/10.1111/j.1540-6261.1995.tb04055.x.

Provost, F., Fawcett, T., & Kohavi, R. (1997). The case against accuracy estimation for comparing induction algorithms. In *Proceedings of the fifteenth international conference on machine learning* (pp. 45–453). Morgan Kaufmann.

Qian, B., & Rasheed, K. (2007). Stock market prediction with multiple classifiers. *Applied Intelligence, 26*(1), 25–33. http://dx.doi.org/10.1007/s10489-006-0001-7.

Rechenthin, M., Street, W. N., & Srinivasan, P. (2013). Stock chatter: using stock sentiment to predict price direction (SSRN Scholarly Paper No. ID 2380419). Social Science Research Network, Rochester, NY.

Rechenthin, M., & Street, W. N. (2013). Using conditional probability to identify trends in intra-day high-frequency equity pricing. *Physica A, 392*, 6169–6188. http://dx.doi.org/10.1016/j.physa.2013.08.003.

Ripley, B. (1996). *Pattern recognition and neural networks*. Cambridge University Press.

Ripley, B. (2013). R package nnet: Feed-forward neural networks and multinomial log-linear models. Available at: http://cran.r-project.org/web/packages/nnet/index.html.

Rodriguez, Pedro N., & Rodriguez, A. (2004). Predicting stock market indices movements. In C. Brebia (Ed.), *Computational Finance and its Applications*. Southampton: Marco Constantino, Wessex Institute of Technology. Available at SSRN: http://ssrn.com/abstract=613042.

Saad, E. W., Prokhorov, D. V., & Wunsch, D. C. (1996). Advanced neural network training methods for low false alarm stock trend prediction. *IEEE International Conference on Neural Networks, 4*, 2021–2026. http://dx.doi.org/10.1109/ICNN.1996.549212.

Saad, E. W., Prokhorov, D. V., & Wunsch, D. C. (1998). Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks. *IEEE Transactions on Neural Networks, 9*(6), 1456–1470. http://dx.doi.org/10.1109/72.728395.

Sarle, W. S. (1997). Should I standardize the input variables (column vectors)?, periodic posting to the Usenet newsgroup comp.ai.neural-nets, Neural Network FAQ, part 2 of 7. URL <ftp://ftp.sas.com/pub/neural/FAQ2.html#A_std_in>.

Schöneburg, E. (1990). Stock price prediction using neural networks: A project report. *Neurocomputing, 2*(1), 17–27. http://dx.doi.org/10.1016/0925-2312(90)90013-H.

Senol, D., & Ozturan, M. (2008). Stock price direction prediction using artificial neural network approach: The case of turkey. *Journal of Artificial Intelligence, 1*(2), 70–77.

Spackman, K. A, (1991). Maximum likelihood training of connectionist models: Comparison with least squares back-propagation and logistic regression. In *Proceedings of the annual symposium on computer application in medical care* (pp. 285–289).

Subha, M. V., & Nambi, S. T. (2012). Classification of Stock Index movement using K-nearest neighbours (k-NN) algorithm. *WSEAS Transactions on Information Science & Applications, 9*(9), 261–270. P259.

Tan, X., Chen, S., Zhou, Z.-H., & Zhang, F. (2005). Recognizing partially occluded, expression variant faces from single training image per person with SOM and Soft k-NN ensemble. *IEEE Transactions on Neural Networks, 16*(4), 875–886.

Tibshirani, R. (1996). Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society: Series B Methodology, 58*, 267–288.

Tsinaslanidis, P. E., & Kugiumtzis, D. (2014). A prediction scheme using perceptually important points and dynamic time warping. *Expert Systems with Applications, 41*, 6848–6860. http://dx.doi.org/10.1016/j.eswa.2014.04.028.

Venables, W. N., & Ripley, B. D. (2002). *Modern Applied Statistics with S* (4th ed.). New York: Springer.

Wang, Y.-F. (2002). Predicting stock price using fuzzy grey prediction system. *Expert Systems with Applications, 22*(1), 33–38. http://dx.doi.org/10.1016/S0957-4174(01)00047-1.

Wang, J.-L., & Chan, S.-H. (2007). Stock market trading rule discovery using pattern recognition and technical analysis. *Expert Systems with Applications, 33*(2), 304–315. http://dx.doi.org/10.1016/j.eswa.2006.05.002.

Wei, L.-Y., & Cheng, C.-H. (2011). A hybrid recurrent neural networks model based on synthesis features to forecast the Taiwan stock market. *International Journal of Innovative Computing, Information and Control, 8*(8), 5559–5571.

Widom, J. (1995). Research problems in data warehousing. In *Proceedings of the fourth international conference on information and knowledge management, CIKM '95* (pp. 25–30). New York, NY, USA: ACM. 10.1145/221270.221319.

Wu, M.-C., Lin, S.-Y., & Lin, C.-H. (2006). An effective application of decision tree to stock trading. *Expert Systems with Applications, 31*(2), 270–274. http://dx.doi.org/10.1016/j.eswa.2005.09.026.

Yeh, I.-C., & Hsu, T.-K. (2014). Exploring the dynamic model of the returns from value stocks and growth stocks using time series mining. *Expert Systems with Applications, 41*, 7730–7743. http://dx.doi.org/10.1016/j.eswa.2014.06.036.

Zhou, Z.-H. (2012). *Ensemble methods: Foundations and algorithms, machine learning & pattern recognition series*. Boca Raton FL: Chapman & Hall/CRC.

Zikowski, K. (2015). Using volume weighted support vector machines with walk forward testing and feature selection for the purpose of creating stock trading strategy. *Expert Systems with Applications, 42*, 1797–1805. http://dx.doi.org/10.1016/j.eswa.2014.10.001.