Detecting stock market manipulation using supervised learning algorithms

Article · March 2015

DOI: 10.1109/DSAA.2014.7058109

CITATIONS
23

READS
6,237

3 authors, including:

Osmar R. Zaïane
University of Alberta
414 PUBLICATIONS **9,976** CITATIONS
SEE PROFILE

David Diaz
University of Chile
38 PUBLICATIONS **473** CITATIONS
SEE PROFILE

Detecting Stock Market Manipulation using Supervised Learning Algorithms

Koosha Golmohammadi, Osmar R. Zaiane
University of Alberta
Department of Computing Science
Edmonton, Canada
{golmoham, zaiane}@ualberta.ca

David Díaz
Universidad de Chile
Departamento de Administración, Facultad de Economía y Negocios
Santiago,Chile
ddiaz@unegocios.cl

***Abstract*— Market manipulation remains the biggest concern of investors in today's securities market, despite fast and strict responses from regulators and exchanges to market participants that pursue such practices. The existing methods in the industry for detecting fraudulent activities in securities market rely heavily on a set of rules based on expert knowledge. The securities market has deviated from its traditional form due to new technologies and changing investment strategies in the past few years. The current securities market demands scalable machine learning algorithms supporting identification of market manipulation activities. In this paper we use supervised learning algorithms to identify suspicious transactions in relation to market manipulation in stock market. We use a case study of manipulated stocks during 2003. We adopt CART, conditional inference trees, C5.0, Random Forest, Naïve Bayes, Neural Networks, SVM and kNN for classification of manipulated samples. Empirical results show that Naïve Bayes outperform other learning methods achieving F_2 measure of 53% (sensitivity and specificity are 89% and 83% respectively).**

Keywords: *supervised learning, classification, data mining, fraud detection, market manipulation, stock market manipulation*

I. INTRODUCTION

Market capitalization exceeded $18 trillion in USA, $2 trillion in Canada and $3.6 trillion in China[1] in 2012 (GDP of USA, Canada and China in 2012 were $16.8, $1.8 and $8.2 trillion respectively). Providing a fair and orderly market for market participants is a challenging task for regulators. During 2010, and just considering Canada, over 200 individuals from 100 companies were prosecuted resulting in over $120 million in fines and compensation[2]. However, the actual losses caused by fraudulent activities in securities market and economy is much higher than these numbers. "Securities fraud broadly refers to deceptive practices in connection with the offer and sale of securities". FBI divides securities fraud into 5 categories[3]: high yield investment fraud, broker embezzlement, late-day trading and market manipulation. Market manipulation remains the biggest concern of investors in today's market, despite fast and strict responses from regulators and exchanges[4]. Market manipulation schemes involve individuals, or a group of people attempting to interfere with a fair and orderly market to gain profit. Market manipulation is forbidden in Canada[5] and in USA[6].

The existing approach in industry for detecting market manipulation is a top-down approach that is based on a set of known patterns and predefined thresholds. Market data such as price and volume of securities (i.e. the number of shares or contracts that are traded in a security) are monitored using a set of rules and red flags trigger notifications. Then, transactions that are associated with the detected periods are investigated further as they might be associated with fraudulent activities. These methods are based on expert knowledge but suffer from two issues i) detecting abnormal periods that are not associated with known symptoms (i.e. unknown manipulative schemes), ii) adapting to the changing market conditions whilst the amount of transactional data is exponentially increasing (this is due to the rapid increase in the number of investors and listed securities) which makes designing new rules and monitoring the vast data challenging. Data mining methods may be used as a bottom-up approach to detect market manipulation based on modeling historical data. These models can be used to identify market manipulation on a new dataset without relying on expert knowledge. The initial results of such models in the literature are encouraging. However, there are many challenges involved in developing data mining methods for detecting fraudulent activities and market manipulation in securities market including **heterogeneous data** (different forms such as news data (e.g. Factiva[7]), analytical data (Trade And Quote (TAQ) from exchanges) and fundamental data (e.g.

[1] http://data.worldbank.org/indicator/CM.MKT.LCAP.CD
[2] Canadian Securities Administrators 2010 report: http://www.osc.gov.on.ca/documents/en/About/csa_20110222_csarpt-enf-2010.pdf
[3] FBI report 2010: http://www.fbi.gov/stats-services/publications/financial-crimesreport-2010-2011
[4] http://economictimes.indiatimes.com/markets/stocks/market-news/market-manipulation-continues-to-be-the-biggest-concerns-for-investors/articleshow/12076298.cms
[5] Bill C-46: *Criminal Code, RSC 1985, c C-46, s 382*. 1985
[6] *Criminal Code, RSC 1985, c C-46, s 382*. 1985
[7] global.factiva.com/

COMPUSTAT[8])), **unlabeled data** (labeled data is very rare because (a) it is very costly and typically requires investigation by auditors, (b) the number of positive samples (fraud cases) constitute a tiny percentage of the total number of samples (also known as imbalanced classes)), **massive datasets** (NASDAQ stock exchange with over 2700 securities listed facilitates more than 5000 transactions per second using its trading platform SuperMontage. Another factor is High Frequency Trading - algorithms that could submit many orders in millisecond [9]), **performance measures** (we discuss this in Section 3) and **complexity**. The problem of detecting market manipulation in securities market is a big data problem where rapidly increasing heterogeneous data from different sources and in different forms are integrated for training prediction models. The impacts on the market, privacy and the training of auditors are other issues that need to be addressed but are not in the scope of this paper. In this paper we focus on adopting supervised learning algorithms for detecting market manipulation in stock market. We present a case study and use these algorithms to build models for predicting transactions that are potentially associated with market manipulation. We extend the work of Diaz et. al. [1] through an extensive set of experiments and adopting learning algorithms to build effective models for detecting market manipulation. We discuss performance measures that are appropriate for this domain and build models accordingly.

For our purposes, we define market manipulation in securities (based on the widely accepted definition in academia and industry) as: *market manipulation involves intentional attempts to deceive investors by affecting or controlling the price of a security or interfering with the fair market to gain profit.* We divide known market manipulation schemes into three groups based on the definition:

1. Marking the close: buying or selling a stock near the close of the day or quarter to affect the closing price. This might be done to help prevent a takeover or rights issue, to avoid margin calls (when a position is financed through borrowing funds) or to affect the performance of a fund manager's portfolio at the end of a quarter (*window dressing*). A typical indicator is trading in small amounts before the market closes,
2. Wash trades: pre-arranged trades that will be reversed later and impose no actual risk to neither buying nor selling parties. These trades aim to give the appearance that purchase and sales have been made (*Pooling* or *churning* can involve wash sales or pre-arranged trades executed in order to give an impression of active trading in a stock),
3. Cornering the market (in a security): to gain control of sufficient amount of the security to control its price.

The rest of this paper is organized as follows. In Section 2 we present a review of data mining techniques for detecting fraudulent activities and market manipulation focusing on supervised learning algorithms. In Section 3, we introduce the case study, the algorithms and the performance measures that we used in our experiments. In Section 4 we present a summary of results and discussion.

II. Related Works

Application of data mining algorithms is a fairly new approach in detecting market manipulation but there has been an increasing number of research works in the past few years. The early theoretical work of Allen and Gorton [2] showed there are opportunities for profitable manipulations in stock market known as trade-based manipulations (e.g. Wash trades, matched order transactions, runs, collusion, etc.). Aggarwal et. al. [3] extended the existing theoretical work combined with an empirical work on market manipulation cases to understand the market manipulation dynamics and economics. Their findings indicate manipulation is typically accompanied with greater stock volatility, great liquidity, and high returns during the manipulation period. The theoretical work by researchers in finance and economics is invaluable for data scientists to identify important features develop heuristics. We presented a comprehensive literature review [4] studying the literature after 2001 to identify (a) the best practices in developing data mining techniques (b) the challenges and issues in design and development, and (c) the proposals for future research, to detect market manipulation in securities market. We identified five categories based on specific contributions of the literature on the data mining approach, goals, and input data:

1. Social Network Analysis: these methods aim to detect trader accounts that collaborate to manipulate the market [5] [6],
2. Visualization: these visualizations go beyond conventional charts enabling auditors to interact with the market data and find predatory patterns [7],
3. Rule Induction: these methods produce a set of rules that can be inspected and used by auditors/regulators of securities market [1],
4. Outlier Detection: the goal of these methods is detecting observations that are inconsistent with remainder of the data (i.e. unknown fraudulent patterns). Also, spikes can be detected effectively using anomaly/outlier detection according to the market conditions, instead of using a predefined threshold to filter out spikes [8] [9],
5. Pattern Recognition using Supervised Learning Methods: the goal of using these methods is detecting

[8] http://www.compustat.com/

[9] HFT accounts for 35% of the stock market trades in Canada and 70% of the stock trades in USA according to the 2010 Report on regulation of trading in financial instruments: Dark Pools & HFT

patterns that are similar to the trends that are known to represent fraudulent activities.

Pattern recognition in securities market typically is performed using supervised learning methods on monthly, daily or intraday data (tick data) where features include statistical averages and returns. Ogut et al. used daily return, average of daily change and average of daily volatility of manipulated stocks and subtracted these numbers from the same parameters of the index [10]. This gives the deviation of manipulated stock from non-manipulated (index) and higher deviations indicate suspicious activities. The assumption in this work is price (consequently return), volume and volatility increases in the manipulation period and drops in the post-manipulation phase. The proposed method is tested using the dataset from Istanbul Stock Exchange (ISE) from an earlier research work on investigating the possibility of gaining profit at the expense of other investors by manipulating the market [11]. Experimental results show that ANN and SVM outperform multivariate statistics techniques (56% compared to 54%) with respect to sensitivity (which is more important in detecting price manipulation as they report correctly classified manipulated data points).

Diaz et al. employed an *open-box* approach in application of data mining methods for detecting intraday price manipulation by mining financial variables, ratios and textual sources [1]. The case study was built based on stock market manipulation cases pursued by the US Securities and Exchange Commission (SEC) during 2003. Different sources of data that were combined to analyze over 100 million trades and 170 thousand quotes in this study include: profiling info (trading venues, market capitalization and betas), intraday trading information (price and volume within a year), and financial news and filing relations. First, using clustering algorithms, a training dataset is created (labeling hours of manipulation, because SEC does not provide this information). Similar cases and Dow Jones Industrial Average (DJI) were used as non-manipulated samples. Second, tree generating classification methods (CART [12], C4.5 [13], QUEST [14]) were used and tested using jack-knife and bootstrapping [15]. Finally, the models were ranked using overall accuracy, measures of unequal importance and sensitivity. A set of rules were generated that could be inspected by securities investigators and be used to detect market manipulation. The highest classification accuracy is reported as 93%.

III. METHODS

The standard approach in application of data mining methods for detecting fraudulent activities in securities market is using a dataset that is produced based on the litigation cases. The training dataset would include fraudulent observations (positive samples) according to legal cases and the rest of observations as would be *normal (negative samples)* [1] [10] [16] [17]. We extend the previous works through a set of extensive experiments, adopting different supervised learning algorithms for classification of market manipulation samples using the data set introduced by Diaz et. al. [1]. We adopt different decision tree algorithms [18], Naïve Bayes, Neural Networks, SVM and kNN.

We define the classification problem as predicting the class of $Y \in \{0,1\}$ based on a feature set of $X_1, X_2, \ldots, X_d$, $X_i \in \mathbb{R}^d$ where Y represents the class of a sample (1 implies a manipulated sample) and X_i represents features such as price change, number of shares in a transaction (i.e. volume), etc. The dataset is divided to training and testing dataset. First, we apply supervised learning algorithms to learn a model on the training dataset, then, the models are used to predict the class of samples in the testing dataset.

A. Case Study

We use the dataset that Diaz et. al. [1] introduced in their paper on analysis of stock market manipulation. The dataset is based on market manipulation cases through SEC between January and December of 2003. The litigation cases that include the legal words related to market manipulation (‘‘manipulation’’, ‘‘marking the close’’ and ‘‘9(a)’’ or ‘‘10(b)’’) are used as manipulated label for that stock and is added to the stock information such as price, volume, the company ticker etc. Standard and Poor's[10] COMPUSTAT database is employed for adding the supplementary information and also including non-manipulated stocks (i.e. control samples). The control stocks are deliberately selected from stocks that are similar to manipulated stocks (the selection is based on similar market capitalization, beta and industry sector). Also, a group of dissimilar stocks were added to the dataset as a control for comparison of manipulated and non-manipulated cases with similar characteristics. These stocks are selected from Dow Jones Industrial (DJI) companies. The dataset includes 175,738 data observations (hourly transactional data) of 64 issuers (31 dissimilar stocks, 8 manipulated stocks and 25 stocks similar to manipulated stocks) between January and December of 2003. There are 69 data attributes (features) in this dataset that represent parameters used in analytical analysis. The dataset includes 27,025 observations for training and the rest are for testing. We only use the training dataset to learn models for identifying manipulated samples.

B. Decision Trees

Decision trees are easy to interpret and explain, non-

[10] Standard and Poor is an American financial services and credit rating agency that has been publishing financial research and analysis on stocks and bonds for over 150 years.

parametric and typically are fast and scalable. Their main disadvantage is that they are prone to *overfiting*, but pruning and ensemble methods such as random forests [19] and boosted trees [20] can be employed to address this issue. A classification tree starts with a single node, and then looks for the binary distinction, which maximizes the information about the class (i.e. minimizing the class impurity). A score measure is defined to evaluate each variable and select the best one as the split:

$$score(S,T) = I(S) - \sum_{i=1}^{p} \frac{N_i}{N} I(S_i)$$

where T is the candidate node that splits the input sample of S with size N into p subsets of size $N_i (i = 1, \dots, p)$ and $I(S)$ is the impurity measure of the output for a given S. Entropy and Gini index are two of the most popular impurity measures and in our problem (i.e. binary classification) are:

$$I_{entropy}(S) = -(\frac{N_+}{N}\log\frac{N_+}{N}) - (\frac{N_-}{N}\log\frac{N_-}{N})$$

$$I_{gini}(S) = \left[\frac{N_+}{N}\left(1 - \frac{N_+}{N}\right)\right] + \left[\frac{N_-}{N}\left(1 - \frac{N_-}{N}\right)\right]$$

where N_+ represents the number of manipulated samples (i.e. positive samples), N_- represents the number of non-manipulated samples (negative samples) in a given subset. This process is repeated on the resulting nodes until it reaches a stopping criterion. The tree that is generated through this process is typically too large and may *overfit*, thus, the tree is pruned back using a validation technique such as cross validation. CART [12] and C4.5 [21] are two classification tree algorithms that follow the greedy approach for building the decision tree (above description). CART uses the Gini index and C4.5 uses the entropy as their impurity function (C5.0 that we used in our experiments is an improved version of C4.5).

Although pruning a tree is effective in reducing the complexity of the tree, generally it is not effective in improving the performance. Algorithms that aggregate different decision trees can improve performance of the decision tree. Random forest [19] is a prominent algorithm that builds each tree using a bootstrap sample. The principle behind random forest is using a group of *weak learners* to build a *strong learner*. Random forest involves an ensemble (*bagging*) of classification trees where a random subset of samples is used to learn a tree in each split. At each node a subset of variables (i.e. features) is selected and the variable that provides the best split (based on some objective function) is used for splitting. The same process is repeated in the next node. After training, a prediction for a given sample is done through averaging votes of individual trees. There are many decision tree algorithms but it has been shown random forest, although very simple, generally outperforms other decision tree algorithms in the study on different datasets by Rich Caruana et. al. [22]. Therefore, experimental results using random forest provide a reasonable proxy for utilizing decision trees in our problem.

C. Naïve Bayes

Applying the Bayes theorem for computing $P(Y = 1|X)$ we have

$$P(Y = 1|X = x_k) = \frac{P(X = x_k|Y = 1)P(Y = 1)}{\sum_j P(X = x_k|Y = y_j)P(Y = y_j)}$$

where the probability of Y given kth sample of X (i. e. x_k) is divided by sum over all legal values for Y (i.e. 0 and 1). Here the training data is used to estimate $P(X|Y)$ and $P(Y)$ and the above Bayes rule is used to resolve the $P(Y|X = x_k)$ for the new x_k. The Naïve Bayes makes the conditional independence assumption (i. e. for given variables X, Y and Z, $(\forall i,j,k)\ P(X = x_i|Y = y_j; Z = z_k) = P(X = x_i|Z = z_k)$) to reduce the number of parameters that need to be estimated. This assumption simplifies $P(X|Y)$ and the classifier that determines the probability of Y, thus

$$P(Y = 1|X_1 \dots X_n) = \frac{P(Y = 1)\prod_i P(X_i|Y = 1)}{\sum_j P(X|Y = y_j)\prod_i P(X_i|Y = y_j)}$$

The above equation gives the probability of Y for the new sample $X\langle X_1 \dots X_n\rangle$ where $P(X_i|Y)$ and $P(Y)$ are computed using the training set. However we are only interested in the maximum likelihood in the above equation and the simplified form is:

$$\hat{y} = \underset{y_k}{\operatorname{argmax}}\, P(Y = y_k)\prod_i P(X_i|Y = y_k)$$

D. Neural Networks

An Artificial Neural Network in contrast to Naïve Bayes estimates the posterior probabilities directly. A Neural Network to learn a model for classification of manipulated samples can be viewed as the function, $F: \mathbb{R}^d \to \{0,1\}$, where X is a d-dimensional variable. This is a function that minimizes the overall mean squared error [23]. The output of the network can be used as the sign predictor for predicting a sample as positive (i.e. manipulated). We adopt the back propagation algorithm of neural networks [24]. The principle behind neural networks, taken from the function of a human neuron, is a nonlinear transformation of the activation into a *prescribed reply*. Our neural network consists of three layers, input layer (the number of nodes in this layer is equal to the number of features, X_i), hidden layer (it is possible to consider multiple hidden layers) and output layer (there is a single node in this layer representing Y). Each node is a neuron and the network is fully

connected (i.e. all neurons, except the neurons in the output layer have axioms to the next layer). The weight of neurons in each layer is updated in the training process using $a_j = \sum_{i=1}^{d} X_i W_{ij}$ and the response of a neuron is calculated using the sigmoid function, $f(a_j) = \frac{1}{1+\exp(-a_j)}$ which is fed forward to the next layer. The weights are updated in the training process such that the overall mean squared error, $SSE = \frac{1}{2}\sum_{j=1}^{N}(Y - \hat{Y})^2$ is minimized, where Y is the actual value, $\hat{Y}$ is the network output and N is the number of samples.

E. Support Vector Machines

We adopt binary SVM for classification [25] of manipulated samples where $Y \in \{-1,1\}$ (i.e. 1 represents a manipulated sample). The main idea behind SVM is finding the *hyperplane* that maximizes the marginal distance (i.e. sum of shortest distances) to data points in a class. The samples in input space are mapped to a feature space using a kernel function to find the *hyperplane*. We use the linear kernel in our experiments (other widely used kernels for SVMs are polynomial, radical basis function (RBF) and sigmoid [15]). The SVM is trying to find w and b in the hyperplane $w.x - b = \pm 1$ which means the marginal distance of $\frac{2}{\|w\|}$ should be maximized. This is an optimization problem of minimizing $\|w\|$ subject to $y_i(w.x_i - b) \geq 1$. A simple trick to solve the optimization problem is working with $\frac{1}{2}\|w\|^2$ to simplify derivation. The optimization problem becomes $\text{argmin}_{w,b}\frac{1}{2}\|w\|^2$ subject to $y_i(w.x_i - b) \geq 1$ and this can be solved through standard application of the Lagrange multiplier.

F. k-Nearest Neighbor

kNN [26] is a simple algorithm that assigns the majority vote of k training samples that are most similar to the to the new sample. There are different similarity measures (i.e. distance measures) such as Euclidean distance, Manhattan distance, cosine distance, etc. kNN is typically used with Euclidean distance. The linear time complexity of Euclidean distance (O(n)) makes it an ideal choice for large datasets. We use kNN with Euclidean distance as the similarity measure of the k nearest samples for binary classification.

G. Performance Measure

Misclassification costs are unequal in fraud detection because false negatives are more costly. In other words, missing a market manipulation case (i.e. positive sample) by predicting it to be non-manipulated (i.e. negative sample), hurts performance of the method more than predicting a sample as positive while it is actually a negative sample (i.e. manipulated case). Threshold, ordering, and probability metrics are effective performance measures for evaluating supervised learning methods for fraud detection [27]. According to our studies the most effective metrics to evaluate the performance of supervised learning methods in classification of market manipulation include Activity Monitoring Operating Characteristic (AMOC) [28] (average score versus false alarm rate), Receiver Operating Characteristic (ROC) analysis (true positive rate versus false positive rate), mean squared error of predictions, maximizing Area under the Receiver Operating Curve (AUC), minimizing cross entropy (CXE) [29] and minimizing Brier score [29].

We use ROC analysis in our experiments reporting sensitivity, specificity and F_2 measure. Let True Positive (TP) represent the number of manipulated cases classified correctly as positive, False Positive (FP) be the number of non-manipulated samples that are incorrectly classified as positive, True Negative (TN) be the number of non-manipulated samples that are correctly classified as positive and False Negative (FN) be the number of manipulated samples that are incorrectly classified as negative, the *precision* and *recall* are $P = \frac{TP}{TP+FP}$ and $R = \frac{TP}{TP+FN}$ respectively. Sensitivity or recall measures the performance of the model in correctly classifying manipulated samples as positive, while the Specificity, $SPC = \frac{TN}{TN+FP}$ measures the performance of the model in correctly classifying non-manipulated samples as negative. We use *F_2 measure* because unlike F_1 measure, which is a harmonic mean of *precision* and *recall*, the *F_2 measure* weights recall twice as much as precision. This is to penalize misclassification of TP more than misclassification of TN. The *F-Measure* is defined as

$$F_\beta = (1+\beta^2) * \frac{P * R}{(\beta^2 * P) + R} = \frac{(1+\beta^2) * TP}{(1+\beta^2) * TP + (\beta^2 * FP) + FP}$$

and *F_2 measure* is a special case of *F-Measure* where β is equal to 2.

IV. Results and Discussion

Diaz et. al. [1] and some previous works used the raw price of securities as a feature in their modeling. We argue that although the price is the most important variable that should be monitored for detecting market manipulation, it should not be used in its raw form. The price of a stock does not reflect the size of a company nor the revenue. Also, the wide range of stock prices is problematic when taking the first difference of the prices. We propose using the price percentage change (i.e. return), $R_t = (P_t - P_{t-1})/P_{t-1}$ or

TABLE I. SUPERVISED LEARNING ALGORITHMS PERFORMANCE

Algorithm	Sensitivity	Specificity	Accuracy	F_2 measure
Naïve Bayes	**0.89**	**0.83**	**0.83**	**0.53**
CART	0.54	0.97	0.94	0.51
Neural Networks	0.68	0.81	0.80	0.40
CTree	0.43	0.95	0.93	0.40
C5.0	0.43	0.92	0.89	0.35
Random Forest	0.32	0.96	0.92	0.30
kNN	0.28	0.96	0.93	0.26

$\log(P_t/P_{t-1})$ where R_t and P_t represent return and price of the security at time t respectively. Furthermore, this is a normalization step, which is a requirement for many statistical and machine learning methods (the sample space of R_t is $[-1, M]$ and $M > 0$). We used stock returns in our experiments and removed the raw price variable from the datasets.

The baseline F_2 measure on the testing dataset (6,685 positive/manipulated samples and 137,373 negative samples) is 17%. If a hypothetical model (this would be also ineffective) predicts all samples as *manipulated*, clearly the recall is 100% but the specificity would be 4%, thus, F_2 measure of 17%. Some related works report the accuracy [1] or overall specificity and sensitivity (i.e. combining performance measures on training and testing datasets or including the performance of models in correctly classifying non-manipulated samples). We emphasize that these numbers may be misleading (some of the worst models that we built in our experiments with respect to correctly classifying manipulated samples, easily exceed accuracy rates of 90%) because a) the misclassification costs for manipulated and non-manipulated cases are unequal, b) the number of samples in the manipulated class is typically significantly lower than the number of samples in the non-manipulated class. In our experiments, we focus on performance of the models on correctly classifying manipulated samples.

Table 1 describes a summary of performance measures of the supervised learning algorithms that we adopted to detect market manipulation on the testing dataset. All the algorithms listed in the table outperform the baseline significantly but SVM which fails to improve the baseline (fine-tuning parameters and using other kernel functions are expected to improve results and we will pursue this avenue in our future work). Decision trees generally produce models that rank high in our experiments. These models are relatively fast and it is possible to improve the results slightly with tweaking the parameters (we did not find significant performance improvements) or using a grid to optimize the parameters. We avoided exhaustive search for best parameters as it is a risk factor for *overfitting*. The Naïve Bayes outperform other algorithms in our experiments with sensitivity and specificity of 89% and 83% respectively. Figure 1 to 3 illustrate ROC curves describing the performance of models based on CART, Random Forest and

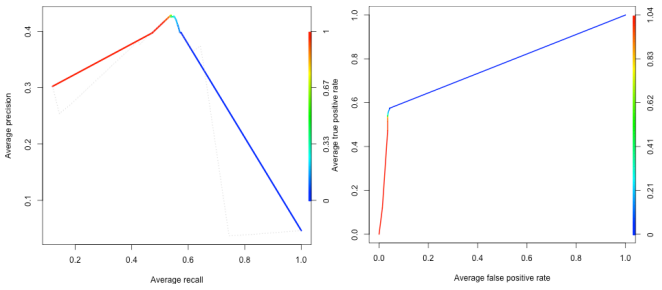

Fig. 1. Performance results using CART – (a) comparing average precision and recall (b) comparing average TP and FP rates

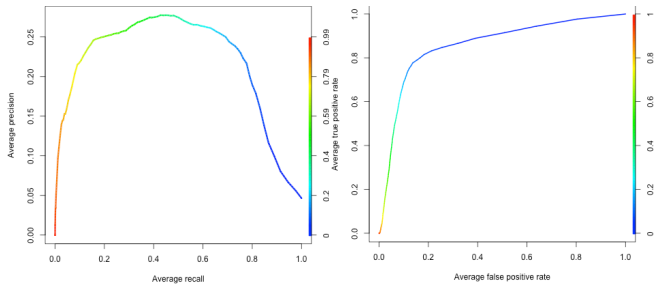

Fig. 2. Performance results using Random Forest – (a) comparing average precision and recall (b) comparing average TP and FP rates

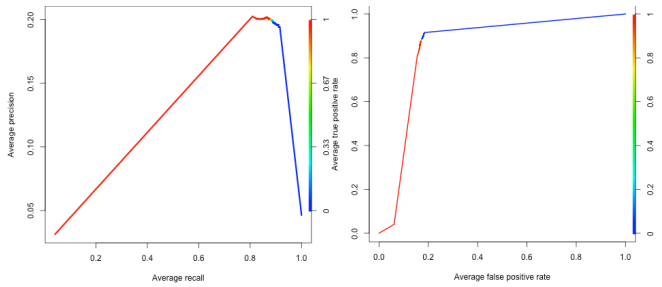

Fig. 3. Performance results using Naïve Bayes – (a) comparing average precision and recall (b) comparing average TP and FP rates

Naïve Bayes.

We use kNN with equal weights and this most likely gives the lower bound performance of kNN on the testing dataset. A future work may use weighted kNN [30] to allow different weights for features (e.g. using Mahalanobis distance [31] to give more weight to features with higher variance). The same principle can be pursued in regression decision trees using a regularizer term to assign different weights to features. Furthermore, we tackle the issue of imbalanced classes by boosting the number of manipulated samples in our datasets through SMOTEBoost [32] and applying decision tree algorithms to the new datasets. The initial results using SMOTEBoost improves performance of the models but the improvements are not significant. We are working on other approaches for boosting the number of samples in the minority class that is highly desired in developing data mining methods for detecting market

manipulation.

The results indicate adopting supervised learning algorithms to identify market manipulation samples using a labeled dataset based on litigation cases is promising. However, a critic may reasonably raise the issue of generality of such models as they are trained using one dataset. We stress the importance of developing techniques to systematically synthesize manipulated samples that can be integrated with actual market data for training and testing data mining methods for detecting market manipulation.

ACKNOWLEDGMENT

We acknowledge the financial support from Natural Science and Engineering Research Council (NSERC), Alberta Innovates Centre for Machine Learning (AICML) and Alberta Innovates Technology Futures (AITF). We acknowledge using the R Project[11] for implementing the experiments and using the high performance computing platform of WestGrid[12].

REFERENCES

[1] D. Diaz, B. Theodoulidis, and P. Sampaio, "Analysis of stock market manipulations using knowledge discovery techniques applied to intraday trade prices," *Expert Syst. Appl.*, vol. 38, no. 10, pp. 12757–12771, Sep. 2011.

[2] F. Allen and G. Gorton, "Stock Price Manipulation, Market Microstructure and Asymmetric Information," *uropean Econ. Rev.*, vol. 36, pp. 624–630, Oct. 1992.

[3] Rajesh K. Aggarwal and Guojun Wu, "Stock Market Manipulations*," *J. Bus.*, vol. 79, pp. 1915–1953, Dec. 2006.

[4] K. Golmohammadi and O. R. Zaiane, "Data Mining Applications for Fraud Detection in Securities Market," in *2012 European Intelligence and Security Informatics Conference*, 2012, pp. 107–114.

[5] J. Neville, D. Jensen, J. Komoroske, K. Palmer, and H. Goldberg, "Using Relational Knowledge Discovery to Prevent Securities Fraud Categories and Subject Descriptors," in *eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, 2005, pp. 449–458.

[6] D. S. Michael Blume, Christof Weinhardt, "Using network analysis for fraud detection in electronic markets," *Inf. Manag. Mark. Eng.*, vol. 4, no. Studies on EOrganisation and Market Engineering, pp. 101–112, 2006.

[7] M. L. Huang, J. Liang, and Q. V. Nguyen, "A Visualization Approach for Frauds Detection in Financial Market," in *2009 13th International Conference Information Visualisation*, 2009, pp. 197–202.

[8] Z. Ferdousi and A. Maeda, "Unsupervised Outlier Detection in Time Series Data," in *22nd International Conference on Data Engineering Workshops (ICDEW'06)*, 2006, pp. x121–x121.

[9] M. Vlachos, K.-L. Wu, S.-K. Chen, and P. S. Yu, "Correlating burst events on streaming stock market data," *Data Min. Knowl. Discov.*, vol. 16, no. 1, pp. 109–133, Mar. 2007.

[10] H. Öğüt, M. Mete Doğanay, and R. Aktaş, "Detecting stock-price manipulation in an emerging market: The case of Turkey," *Expert Syst. Appl.*, vol. 36, no. 9, pp. 11944–11949, Nov. 2009.

[11] R. Aktaş and Doğanay M., "Stock-price manipulation in the Istanbul stock exchange," *Eurasian Rev. Econ. Financ.*, vol. 2, pp. 21–28, 2006.

[12] L. Breiman, J. Friedman, C. Stone, and R. Olshen, "Classification and regression trees," 1984.

[13] J. Quinlan, *C4. 5: programs for machine learning*. 1993.

[14] W. Loh and Y. Shih, "Split selection methods for classification trees," *Stat. Sin.*, 1997.

[15] C. L. Chih-wei Hsu, Chih-chung Chang, "A practical guide to support vector classification," 2010.

[16] J. D. Kirkland, T. E. Senator, J. J. Hayden, T. Dybala, H. G. Goldberg, and P. Shyr, "The NASD Regulation Advanced-Detection System (ADS)," *AI Magazine*, vol. 20, no. 1. p. 55, 15-Mar-1999.

[17] T. E. Senator, "Ongoing management and application of discovered knowledge in a large regulatory organization," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '00*, 2000, pp. 44–53.

[18] A. Zeileis, T. Hothorn, and K. Hornik, "Model-Based Recursive Partitioning," *J. Comput. Graph. Stat.*, vol. 17, no. 2, pp. 492–514, Jun. 2008.

[19] L. Breiman, "Random Forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001.

[20] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods," *Ann. Stat.*, vol. 26, no. 5, pp. 1651–1686, Oct. 1998.

[21] S. L. Salzberg, "C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993," *Mach. Learn.*, vol. 16, no. 3, pp. 235–240, Sep. 1994.

[22] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *Proceedings of the 23rd international conference on Machine learning - ICML '06*, 2006, pp. 161–168.

[23] A. Papoulis and S. Pillai, *Probability, random variables, and stochastic processes*. Tata McGraw-Hill Education, 2002.

[24] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[25] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition," *Data Min. Knowl. Discov.*, vol. 2, no. 2, pp. 121–167, Jun. 1998.

[26] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967.

[27] C. Phua, V. Lee, K. Smith, and R. Gayler, "A Comprehensive Survey of Data Mining-based Fraud Detection Research," *Artif. Intell. Rev.*, 2005.

[28] T. Fawcett and F. Provost, "Activity monitoring: Noticing interesting changes in behavior," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining - KDD 99*, 1999, pp. 53–62.

[29] S. Viaene, R. A. Derrig, and G. Dedene, "A case study of applying boosting naive bayes to claim fraud diagnosis," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 5, pp. 612–620, May 2004.

[30] S. S. Scott Cost, "A Weighted Nearest Neighbor Algorithm for Learning with Symbolic Features," *Mach. Learn.*, vol. 10, no. 1, pp. 57–78, 1993.

[31] K. Q. Weinberger and L. K. Saul, "Distance Metric Learning for Large Margin Nearest Neighbor Classification," *J. Mach. Learn. Res.*, vol. 10, pp. 207–244, Jun. 2009.

[32] K. W. B. Nitesh V. Chawla, Ar Lazarevic, Lawrence O. Hall, "SMOTEBoost: improving prediction of the minority class in boosting," in *Principles of Knowledge Discovery in Databases, PKDD-2003*, 2003, pp. 107–119.

[11] http://www.r-project.org

[12] www.westgrid.ca