*Auditing: A Journal of Practice & Theory*
Vol. 30, No. 2
May 2011
pp. 19–50

American Accounting Association
DOI: 10.2308/ajpt-50009

# Financial Statement Fraud Detection: An Analysis of Statistical and Machine Learning Algorithms

**Johan Perols**

**SUMMARY:** This study compares the performance of six popular statistical and machine learning models in detecting financial statement fraud under different assumptions of misclassification costs and ratios of fraud firms to nonfraud firms. The results show, somewhat surprisingly, that logistic regression and support vector machines perform well relative to an artificial neural network, bagging, C4.5, and stacking. The results also reveal some diversity in predictors used across the classification algorithms. Out of 42 predictors examined, only six are consistently selected and used by different classification algorithms: auditor turnover, total discretionary accruals, Big 4 auditor, accounts receivable, meeting or beating analyst forecasts, and unexpected employee productivity. These findings extend financial statement fraud research and can be used by practitioners and regulators to improve fraud risk models.

**Keywords:** analytical auditing; financial statement fraud; fraud detection; fraud predictors; classification algorithms.

**Data Availability:** A list of fraud companies used in this study is available from the author upon request. All other data sources are described in the text.

## INTRODUCTION

The cost of financial statement fraud is estimated at $572 billion[1] per year in the U.S. (Association of Certified Fraud Examiners [ACFE] 2008). In addition to direct costs, financial statement fraud negatively affects employees and investors and undermines the

*Johan Perols is an Assistant Professor at the University of San Diego.*

This study is based on one of my three dissertation papers completed at the University of South Florida. I thank my dissertation co-chairs, Jacqueline Reck and Kaushal Chari, and committee members, Uday Murthy and Manish Agrawal. I am also grateful to Robert Knechel (associate editor), two anonymous reviewers, and Ann Dzuranin for their helpful suggestions. An earlier version of this paper was awarded the 2009 AAA Information System Section Outstanding Dissertation Award.

Editor's note: Accepted by Robert Knechel.

*Submitted: November 2008*
*Accepted: March 2010*
*Published Online: May 2011*

[1] The ACFE (2008) report provides estimates of occupational fraud cost, median cost per fraud category, and number of cases. To derive the estimate for total cost of financial statement fraud, it was assumed that the relative differences among the fraud categories in mean costs are similar to differences in median costs.

reliability of corporate financial statements, which results in higher transaction costs and less efficient markets. Auditors, both through self-regulation and legislation, are responsible for providing reasonable assurance that financial statements are free of material misstatement caused by fraud. Earlier auditing standards, i.e., Statement on Auditing Standards (SAS) No. 53, only indirectly addressed this responsibility through references to "irregularities" (American Institute of Certified Public Accountants [AICPA] 1988). However, more recent auditing standards, SAS No. 82 and later, make this responsibility explicit. Auditors must provide "reasonable assurance about whether the financial statements are free of material misstatements, whether caused by error or fraud" (AICPA 1997, AU 110.02).

To improve fraud detection, recent accounting research (e.g., Lin et al. 2003; Kirkos et al. 2007) has focused on testing the utility of various statistical and machine learning algorithms, such as logistic regression and artificial neural networks (ANN), in detecting financial statement fraud. This research is important since the financial statement fraud domain is unique. Distinguishing characteristics that make this domain unique include: (1) the ratio of fraud to nonfraud firms is small (high class imbalance; the prior fraud probability[2] is low, i.e., there are many more nonfraud firms than fraud firms); (2) the ratio of false positive[3] to false negative[4] misclassification costs is small (high cost imbalance; it is more costly to classify a fraud firm as a nonfraud firm than to classify a nonfraud firm as a fraud firm); (3) the attributes used to detect fraud are relatively noisy, where similar attribute values can signal both fraudulent and nonfraudulent activities; and (4) fraudsters actively attempt to conceal the fraud, thereby making fraud firm attribute values look similar to nonfraud firm attribute values. Because of these unique characteristics, it is not clear, without empirical evaluation, whether statistical and machine learning algorithms (henceforth, classification algorithms) that perform well in other domains will also perform well in the financial statement fraud domain. Therefore, research that focuses specifically on financial statement fraud detection is needed.

Prior research focusing on evaluating the effectiveness of different classification algorithms in detecting fraud has typically introduced different variations of ANN (e.g., Green and Choi 1997; Fanning and Cogger 1998; Lin et al. 2003) and compared these algorithms to logistic regression without taking into account important dimensions of the distinguishing characteristics. Furthermore, other classification algorithms, like support vector machines[5] (SVM), decision trees,[6] and ensemble-based methods,[7] have become increasingly popular in other domains. Financial statement fraud researchers have recently begun evaluating these additional classification algorithms (e.g., Kotsiantis et al. 2006; Kirkos et al. 2007), but without considering the distinguishing characteristics. The distinguishing characteristics, like class and cost imbalance,

[2] Prior fraud probability is the probability that a given firm is a fraud firm if randomly selected from a dataset. In this paper, the *training* prior fraud probability refers to the number of fraud firms divided by the total number of firms in a given training sample, while the *evaluation* prior fraud probability refers to the assumed probability of fraud in the population of interest. The former probability is manipulated by changing the ratio of fraud to nonfraud firms (undersampling the majority class) in the sample, while the latter is manipulated by changing the assumed ratio of fraud to nonfraud firms of the population when calculating performance measures.

[3] Nonfraud firms erroneously classified as fraudulent.

[4] Fraud firms erroneously classified as not fraudulent.

[5] SVM algorithms classify data points by learning hyperplanes (linear models) that provide the best separation of positive instances from negative instances (Platt 1999).

[6] Decision trees like C4.5 examine the information gain provided by different attributes and split the data using the attribute that provides the highest information gain (Quinlan 1993).

[7] An ensemble-based method is a type of classification algorithm that combines probability estimates of a group (ensemble) of base-classifiers to create an ensemble probability estimate. Different ensemble-based methods include different types of base-classifiers (e.g., stacking) in the ensemble, train the base-classifiers differently (e.g., bagging), and use different algorithms to combine base-classifier probability outputs into an overall ensemble decision.

introduce problems that typically undermine classification performance and are, therefore, important to consider (Weiss 2004).

Based on these issues, my research objective is to compare the utility of a fairly comprehensive set of classification algorithms in financial statement fraud detection while taking into account the distinguishing characteristics. More specifically, I examine:

**RQ1:** Given different assumptions about prior fraud probabilities and misclassification costs during evaluation, what classification algorithm(s) provide the most utility?

**RQ2:** What prior fraud probability and misclassification cost should be used when training classifiers?

**RQ3:** What predictors are useful to these classification algorithms?

The answers to these questions are of practical value to auditors and to institutions like the Securities and Exchange Commission (SEC). The results provide guidance as to what algorithms and predictors to use when creating new models for financial statement fraud detection under specific class and cost imbalance ratios. The results, furthermore, provide insights into appropriate class distributions to use when training the classifiers. Auditors can use these findings to improve client selection, audit planning and analytical procedures, while the SEC can leverage the findings to target companies that are more likely to have committed financial statement fraud.

The reminder of the paper is organized as follows. I next provide an overview of related research, and then a description of the experimental variables and data used to evaluate the classification algorithms. This is followed by an explanation of the experimental procedure and results. The results are summarized in the final section, along with a discussion of research contributions and limitations, and suggestions for future research.

## RELATED RESEARCH

Research that evaluates the effectiveness of different fraud classification algorithms[8] has typically introduced different variations of ANN and compared these algorithms to logistic regression (see Table 1 for an overview of this research). Green and Choi (1997) and Fanning and Cogger (1998) showed that ANN provide good performance relative to random guessing, discriminant analysis, and logistic regression when (1) the prior probability of fraud is assumed to be 0.5, (2) false positive and false negative classification costs are assumed to be equal, and (3) using a performance measure that does not take into account class and cost imbalances. In reality, the prior probability of fraud is much smaller than 0.5 (Bell and Carcello 2000) and the cost of false negative classifications are, on average, much larger than the cost of false positive classifications (Bayley and Taylor 2007). Furthermore, the relative performance of two algorithms typically depends on the specific class and cost imbalance level assumptions used in the comparison (Provost et al. 1998). Thus, it is an empirical question whether the results in Green and Choi (1997) and Fanning and Cogger (1998) generalize to more realistic data imbalance[9] assumptions.

[8] Please refer to the "Experimental Variables and Data" section for classification algorithm descriptions.

[9] Bell and Carcello (2000) estimate that the probability of fraud is 0.6 percent, while Bayley and Taylor (2007) estimate, based on financial statement fraud effects reported in Beneish (1999), that the ratio of false positive classification costs to false negative classification costs is between 1:20 and 1:40, i.e., it is 20 to 40 times as costly to classify a fraud firm as a nonfraud firm as it is to classify a nonfraud firm as a fraud firm.

**TABLE 1**

**Prior Literature Overview**

| Author | Algorithm[a] | Dataset[b] | Assumptions: Evaluation p(fraud)[c] | Assumptions: Evaluation Costs[d] | Assumptions: Training[e] p(fraud) and Costs | Assumptions: Classification Threshold[f] | Results |
|---|---|---|---|---|---|---|---|
| Green and Choi (1997) | ANN and Random Guessing | 86 SEC fraud cases matched with 86 nonfraud cases | Balanced | Balanced | Balanced | Optimal Not Determined | ANN outperforms random guessing on the training sample, but not on the evaluation sample. |
| Fanning and Cogger (1998) | ANN, DA, and Logistic Regression | 102 SEC fraud cases matched with 102 nonfraud cases | Balanced | Balanced | Balanced | Optimal Not Determined | ANN is more accurate but has lower true positive rate than DA and Logistic Regression. |
| Feroz et al. (2000) | ANN and Logistic Regression | 42 SEC fraud cases matched with 90 nonfraud cases | Realistic | Manipulated | Balanced[g] | Optimal Not Determined | Both studies show that in terms of ERC, ANN outperforms logistic regression when no cost imbalance is assumed during evaluation. While not concluded, the two studies also show that logistic regression outperforms ANN at more realistic evaluation cost imbalance levels. |
| Lin et al. (2003) | ANN and Logistic Regression | 40 SEC fraud cases matched with 160 nonfraud cases | Realistic | Manipulated | Balanced | Optimal Not Determined | |
| Kotsiantis et al. (2006) | C4.5, ANN, K2, 3-NN, RIPPER, SMO, Logistic Regression, MP and MLR Stacking, Grading, and Simple Voting | 41 Greek fraud cases matched with 123 Greek nonfraud cases | Mild Imbalance | Balanced | Mild Imbalance | Optimal Not Determined | MP stacking is more accurate than the other algorithms. Logistic regression and ANN performed relatively poorly. |
| Kirkos et al. (2007) | ANN, Bayesian Belief Network, and Decision Tree | 38 Greek fraud cases matched with 38 Greek nonfraud cases | Balanced | Balanced | Balanced | Optimal Not Determined | Bayesian Belief Network outperforms the other algorithms in terms of both fraud and nonfraud classification accuracy. |

*(continued on next page)*

**TABLE 1** (continued)

[a] ANN = Artificial Neural Network; DA = Discriminant Analysis; NN = Nearest Neighbor; SMO = Sequential Minimal Optimization Support Vector Machine.
[b] SEC = Dataset obtained from SEC releases.
[c] *Evaluation p(fraud)* refers to the ratio of fraud to nonfraud firms used when evaluating the classifiers. *Balanced* means that the number of fraud firms is the same as the number of nonfraud firms. *Mild* means that the number of fraud firms is slightly lower than the number of nonfraud firms.
[d] *Evaluation costs* refer to the ratio of false positive misclassification cost to false negative misclassification cost used when evaluating the classifiers. *Balanced* means that the cost of a false positive misclassification is equal to the cost of a false negative misclassification. *Manipulated* means that the relative performance of the classifiers was compared at different ratios of false positive misclassification cost to false negative misclassification cost.
[e] *Training p(fraud) and costs* refers to the prior fraud probability and relative error costs used when training the classifiers. *Balanced* means that the classifiers were trained assuming that (1) the number of fraud firms is the same as the number of nonfraud firms, and (2) the cost of a false positive misclassification is equal to the cost of a false negative misclassification.
[f] *Classification threshold* refers to the cutoff value that is compared to classifier fraud probability predictions to determine firm class memberships (fraud or nonfraud). *Optimal not determined* means that classification thresholds that maximize classification performance were not empirically determined.
[g] Feroz et al. (2000) used balanced training data for the experimental comparison in which performance was measured using ERC.

Feroz et al. (2000) compared the utility of an ANN model with logistic regression based on Hit-Rate,[10] Overall Error Rate,[11] and Estimated Relative Costs of misclassification[12] (ERC), while Lin et al. (2003) compared a fuzzy ANN to logistic regression using the same performance measures. The results in Feroz et al. (2000) showed that logistic regression performed better than ANN at relative error costs from 1:1 to 1:40, and that the ANN performed better than logistic regression at relative error costs of 1:50. Similarly, the results in Lin et al. (2003) showed that logistic regression performed better than ANN at relative error costs from 1:1 to 1:30, and that the ANN performed better than logistic regression at relative error costs of from 1:40 to 1:100. These results corroborate the findings in Fanning and Cogger (1998) by showing that ANN outperforms logistic regression for more balanced datasets. Furthermore, while these studies concluded that ANNs outperform logistic regression, they actually showed that when the prior fraud probability and relative error cost levels are adjusted to more realistic levels during evaluation, logistic regression outperforms ANN.

In their ERC analyses, Feroz et al. (2000) and Lin et al. (2003), however, did not manipulate three important factors that, given the distinguishing characteristics of the fraud domain, are important to consider: *training* prior fraud probabilities (the prior fraud probability used for classifier training), *training* relative error costs (the relative error costs used for classifier training), and classification thresholds.[13] These factors are explained in more detail in the "Experimental Variables and Data" section below. Thus, it is an empirical question whether the relative performance of ANN and logistic regression changes if these factors are manipulated. A more important question is whether other statistical and machine learning algorithms could improve classification performance. Machine learning research outside the fraud domain has developed and evaluated classification algorithms other than ANN and logistic regression. Of particular interest to the fraud domain are studies that examine classification algorithm performance using imbalanced datasets or datasets that are similar to the fraud domain in other respects. Such machine learning studies have found that classification algorithms like SVM (Fries et al. 1998; Fan and Palaniswami 2000; Shin et al. 2005), C4.5 (Phua et al. 2004), and bagging (West et al. 2005) perform relatively well.

More recently, financial statement fraud researchers have examined whether some of these classification algorithms could be used to improve fraud classification performance. Kotsiantis et al. (2006) used 41 fraud and 123 nonfraud firms in Greece to examine 11 classification algorithms: C4.5, RBF, K2, 3-NN, RIPPER, SMO, logistic regression, MP stacking, MLR stacking, Grading, and Simple Voting. The results, in terms of overall accuracy, showed that MP stacking provides the best performance, while logistic regression and ANN provide relatively poor performance. Kirkos et al. (2007) used 38 fraud and 38 nonfraud firms to investigate the relative utility of an ANN, a Bayesian belief network, and a decision tree learner. The reported class accuracies indicated that the Bayesian belief network outperforms the ANN and decision

[10] Hit-Rate (or True Positive Rate) refers to the number of accurately classified fraud firms to the total number of fraud firms in the sample.

[11] Overall error rate takes into account differences in *evaluation* prior fraud probability, but assumes equal classification error costs.

[12] ERC takes into account both *evaluation* prior fraud probability and *evaluation* classification error costs. Note, however, that if the *training* prior fraud probability and *training* relative error costs are not adjusted when training the classifiers, the models might not perform optimally for the different *evaluation* prior probability and *evaluation* relative error costs combinations. Thus, this performance measure might be misleading if the different models are not retrained for each *evaluation* prior probability and *evaluation* relative error cost combination examined.

[13] A classification threshold is a cutoff value that is compared to classifier fraud probability predictions to determine firm class memberships (fraud or nonfraud); referred to as cutoff probability in (Beneish 1997).

tree. While these studies indicate that stacking and Bayesian belief networks perform well, both studies assumed that the error costs were the same for false positive and false negative classification errors, and their datasets contained the same or almost the same number of fraud firms as nonfraud firms. Furthermore, both studies used accuracy, which does not take class and cost imbalance into account, to measure performance.

To summarize, extant research has offered insights into the relative performance of different classification algorithms. ANN performs well relative to logistic regression when the dataset is balanced. Although concluding that ANN outperforms logistic regression, this research has also shown that logistic regression performs well relative to ANN when class and cost imbalance was taken into account during *evaluation*. More recent studies have examined the performance of additional classification algorithms under relatively balanced conditions and have shown that meta-classifiers have the best classification accuracy. Using the prior research as a foundation, this study extends this literature by evaluating the performance of a relatively representative set of classification algorithms. Furthermore, this study uses a performance measure that takes into account the class and cost imbalance, and examines under what specific *evaluation* prior fraud probability and *evaluation* relative error cost levels these algorithms perform well. This study also examines what prior fraud probability and misclassification cost should be used when *training* the classifiers, and what predictors provide utility to these classification algorithms.

## EXPERIMENTAL VARIABLES AND DATA

The next three subsections describe three factors that were manipulated in the experiment: classification algorithms, prior fraud probability, and relative error costs. The final two subsections describe the dependent variable and the data sample.

### Classification Algorithms

The overarching goal of this research was to examine the performance of different classification algorithms in fraud detection. The primary experimental factor of interest was, therefore, classification algorithm. The classification algorithms were obtained from Weka, an open source data mining tool. Using an open source tool facilitates the replication and extension of this study. Weka implements a relatively complete set of classification algorithms, including many of the most popular. Based on the related research (i.e., prior financial statement fraud research and prior data mining research in domains with imbalanced datasets mentioned earlier), six algorithms were selected from Weka: (1) J48, (2) SMO, (3) MultilayerPerceptron, (4) Logistics, (5) stacking, and (6) bagging. J48 is a decision tree learner and Weka's implementation of C4.5 version 8. SMO is a support vector machine (SVM) and Logistics is Weka's logistic regression implementation. Both these classification algorithms are linear functions. MultilayerPerceptron is Weka's backpropagation ANN implementation, and stacking and bagging are two ensemble-based methods. Please refer to the "Classification Algorithm Descriptions" and "Classifier Tuning" sections in the Appendix for further details.

Logistic regression, ANN, and stacking were included, as they had performed well in prior fraud research (Feroz et al. 2000; Lin et al. 2003; Kotsiantis et al. 2006). However, it was not clear if these classification algorithms would perform well under realistic conditions and relative to not-yet-examined classification algorithms. Bagging, C4.5, and SVM were included because prior data mining research (Fries et al. 1998; Phua et al. 2004; West et al. 2005) found that these classification algorithms performed well in domains with imbalanced data; that is, where the

majority class was larger than the minority class, which is true in the financial statement fraud domain. It was, however, not known how these classification algorithms would perform in fraud detection.

### Prior Fraud Probability

The prior probability of fraud impacts both classifier *training* and *evaluation*. Two classifiers that are based on the same classification algorithm can produce different results if they are *trained* on data with different prior probabilities. Furthermore, the relative performance of already trained classifiers can change if the *evaluation* prior probabilities change. The classifiers, therefore, have to be both trained using appropriate prior probabilities in the training sample and evaluated using relevant assumptions of prior probabilities. To determine appropriate training prior fraud probabilities, classification algorithm performance was examined after undersampling the majority class at different ratios of fraud to nonfraud cases. Please refer to the "Preprocessing" section in the Appendix for further details. Undersampling the majority class, i.e., randomly deleting majority class cases, is a common, simple, and relatively effective approach to deal with class (and cost) imbalance (Drummond and Holte 2003).

Furthermore, for results to generalize to the population of interest, the *evaluation* prior fraud probability should reflect the prior probability of fraud in the population (the naturally occurring prior fraud probability). Bell and Carcello (2000) estimate that only around 0.6 percent of all firm years are fraudulent. However, this estimate is likely to change over time and be different for different populations of interest. Therefore, I manipulated the evaluation prior fraud probability at three levels: low, medium, and high. I defined medium as prior fraud probability of 0.006, the estimate from Bell and Carcello (2000), low as 50 percent of medium or 0.003, and high as 200 percent of medium or 0.012. As the classifiers do not learn from evaluation data, manipulating this data does not impact the final classification model. Following prior fraud research (Feroz et al. 2000; Lin et al. 2003), the evaluation prior fraud probability was, therefore, manipulated in the calculation of the dependent variable, ERC (see the "Dependent Variable" subsection below), and not by undersampling the evaluation data.

To summarize, the prior probability of fraud was manipulated both when training classifiers for classifier tuning purposes and when evaluating the performance of the classification algorithms. When the prior fraud probability was manipulated for training purposes, the prior fraud probability in the training sample was changed by undersampling the data. When the prior fraud probability was manipulated for evaluation purposes, the manipulation only impacted the calculation of the dependent variable and did not change the data.

### Classification Cost

Given a binary problem like fraud, there are four potential classification outcomes: (1) true positive, a fraud firm is correctly classified as a fraud firm; (2) false negative, a fraud firm is incorrectly classified as a nonfraud firm; (3) true negative, a nonfraud firm is correctly classified as a nonfraud firm; and (4) false positive, a nonfraud firm is incorrectly classified as a fraud firm. False negative and false positive classifications are associated with different misclassification costs. Similar to prior fraud probability, the ratios of these costs impact both training and evaluation of classifiers. The classifiers, therefore, have to be both trained using appropriate cost ratios in the training sample and evaluated using relevant cost ratio assumptions. When determining appropriate training cost ratios, classification algorithm performance was examined after undersampling the

majority class instead of manipulating the cost ratio,[14] which was not possible as the examined classification algorithms are not cost-sensitive. Please refer to the "Preprocessing" section in the Appendix for further details. As stated earlier, undersampling the majority class is a common, simple, and effective approach to deal with cost and class imbalances (Drummond and Holte 2003).

When evaluating the classification algorithms using specific assumptions about relative error costs, the results might not hold for other relative error cost levels. Therefore, the relative error costs used in the *evaluation* should reflect the relative error costs in the population. These costs are, however, difficult to estimate. Researchers typically examine the classification performance over a wide range of evaluation relative error costs (Feroz et al. 2000; Lin et al. 2003), which reduces the risk of cost misspecification and provides richer information to other researchers and practitioners. Following prior research (Lin et al. 2003), classification performance was evaluated over a wide range of evaluation relative error costs, from 1:1 through 1:100 (false positive to false negative costs). A more focused evaluation was also performed using 1:20, 1:30, and 1:40 evaluation relative error costs, based on relative error cost estimates from Bayley and Taylor (2007), who estimated that these costs are on average between 1:20 and 1:40. Thus, this more focused analysis provides insights into the relative performance of the classification algorithms under what is estimated to be realistic circumstances.

During evaluation, classifiers' fraud probability estimates do not change based on different assumptions about the relative error costs used for evaluation. To clarify, after a classifier has estimated the class membership of an observation, e.g., the classifier estimates that there is a 0.8 probability of the observation being fraudulent, then that estimate does not change if the relative error costs used for evaluation change. Therefore, relative error cost was not manipulated by undersampling the evaluation sample. Instead, different evaluation relative error costs were used when calculating ERC (see the "Dependent Variable" subsection).

To summarize, the relative error cost was manipulated both when training classifiers for classifier tuning purposes and when evaluating the performance of the classification algorithms. When the relative error cost was manipulated for training purposes, the prior fraud probability, instead of the relative error cost, was changed in the training sample by undersampling the data. When the relative error cost was manipulated for evaluation purposes, the manipulation only impacted the calculation of the dependent variable and did not change the underlying data.

### Dependent Variable

It is important to use an appropriate measure when evaluating classification performance in domains with high class and cost imbalance, like the fraud domain (Weiss 2004). If the class and cost imbalance can be reasonably estimated, then classification cost measures based on specific

[14] Altering the priors (i.e., the class imbalance), in this case through undersampling, is equivalent to altering the cost matrix (Breiman et al. 1984). To clarify, assume two different classifiers, one that is cost-sensitive and one that is not. The cost-sensitive classifier learns and classifies a dataset with 20 fraud and 1,000 nonfraud cases, where the cost of misclassifying a fraud case is $100 and the cost to misclassify a nonfraud case is $10 (i.e., it is ten times more expensive to misclassify a fraud case than it is to misclassify a nonfraud case). If we assume that half the fraud and half the nonfraud cases are misclassified, then the ratio of total fraud to total nonfraud misclassification cost is 1 to 5 ($10 * 100$ to $500 * 10$). The second classifier, which is not cost-sensitive, learns and classifies the same dataset, but because the cost for fraud and nonfraud relative error cost has to be assumed to be balanced, the nonfraud firm class is undersampled to 10 percent of the original class size (the inverse of the ratio of the cost of a fraud to the cost of a nonfraud misclassification), i.e., the dataset still has 20 fraud cases but now only contains 100 nonfraud cases. If we, again, assume that half the fraud and half the nonfraud cases are misclassified, then the ratio of total fraud to total nonfraud misclassification cost is again 1 to 5 ($10 * 1$ to $50 * 1$). Thus, the relative importance of correctly classifying fraud cases and nonfraud cases is the same in both scenarios. The reader is referred to (Breiman et al. 1984) for a formal proof of this equivalence.

prior fraud probabilities and relative error costs are preferred.[15] Consistent with prior financial statement fraud research (Feroz et al. 2000; Lin et al. 2003), performance was measured using ERC (Dopuch et al. 1987). Given specific classification results and manipulation levels of *evaluation* prior fraud probability and *evaluation* relative error costs, ERC is calculated as:

$$ERC = n^{FN}/n^{P} \times C^{FN} \times P(Fraud) + n^{FP}/n^{N} \times C^{FP} \times P(Nonfraud) \quad (1)$$

where $P(Fraud)$ and $P(Nonfraud)$ are the evaluation prior fraud and nonfraud probabilities, respectively; $C^{FP}$ is the cost of false positive classifications and $C^{FN}$ is the cost of false negative classifications, both deflated by the lower of $C^{FP}$ or $C^{FN}$; $n^{FP}$ is the number of false positive classifications, $n^{FN}$ is the number of false negative classifications, $n^{P}$ is the number of positive instances in the dataset; and $n^{N}$ is the number of negative instances in the dataset. As in Beneish (1997), *ERC* was derived for each classifier at the classification threshold (termed cutoff probability in Beneish [1997]) that minimized the *ERC* for given levels of evaluation prior fraud probabilities and evaluation relative error costs.

## Data Sample

### *Classification Objects: Fraud and Nonfraud Firms Data*

Fraudulent observations were located by performing a keyword search and reading SEC fraud investigations reported in Accounting and Auditing Enforcement Releases (AAER) from the fourth quarter of 1998 through the fourth quarter of 2005. A total of 745 potential observations were obtained from this initial search (see Table 2). The dataset was then reduced by eliminating: duplicates; financial companies; firms without the first fraud year specified in the SEC release; nonannual financial statement fraud; foreign corporations; releases related to auditors; not-for-profit organizations; and fraud related to registration statements, 10-KSB, or IPO. These observations were deleted from the sample because the nature of the fraud committed or the underlying structure of the data in the deleted categories are dissimilar from the selected sample. For example, registration statements, 10-KSB, and IPO lack much of the required data needed to create the fraud predictors, and regulations governing financial firms are substantially different from those governing other types of firms. These observations were, therefore, deleted to reduce problems associated with case disjuncts and noisy measures. Furthermore, the research that has developed the predictors used in this study has typically eliminated these categories and then designed the fraud predictors with the existing observations in mind. Thus, the predictors used in this study were not designed to estimate fraud in the deleted categories. An additional 75

[15] In domains with high class and cost imbalance, but where the priors and costs cannot be reasonably estimated, Receiver Operating Characteristics (ROC) curves and the related measure, Area Under the ROC Curve (AUC), are typically used. ROC curves show the performance of classifiers in terms of true positive rate (i.e., the percentage of fraud firms accurately classified as fraud firms) on the y-axis and false positive rate on the x-axis (i.e., the percentage of nonfraud firms incorrectly classified as fraud firms) as the classification threshold (probability cutoff) is varied. ROC curves are used to visually compare classifiers, while AUC provides a single measure related to the ROC curve that can then be used in statistical analysis. However, because the AUC provides a single measure of the entire area under the ROC curve, AUC is an average comparison of classifiers (Chawla 2005). Thus, the AUC measure includes regions of the AUC curve that might not be of interest given specific cost and priors. When the cost and class imbalance can be reasonably estimated, then specific points on the ROC curve will provide optimal tradeoffs between true positive rate and false positive rate (Provost and Fawcett 1997). This is equivalent to using an evaluation measure that takes cost and class imbalance estimates into account and then finding the classification threshold that minimizes the misclassification cost (or maximizes the benefit), i.e., we move to the optimal point on the ROC curve that has true positive and false positive rates that minimizes the misclassification cost.

**TABLE 2**

**Sample Selection**

| | |
|---|---|
| **Panel A: Fraud Firms** | |
| Firms investigated by the SEC for fraudulent financial reporting from 4Q 1998 through 4Q 2005 | 745 |
| Less: Financial companies | (35) |
| Less: Not annual (10-K) fraud | (116) |
| Less: Foreign companies | (9) |
| Less: Not-for-profit organizations | (10) |
| Less: Registration, 10-KSB, and IPO-related fraud | (78) |
| Less: Fraud year missing | (13) |
| Less: Duplicates | (287) |
| Remaining Fraud Observations | 197 |
| Add: Fraud firms from Beasley (1996) | 75 |
| Less: Not in Compustat or CompactD for first fraud year or four prior years or I/B/E/S for first fraud year | (221) |
| Usable Fraud Observations | 51 |
| **Panel B: Nonfraud Firms** | |
| Nonfraud Observations | 15,934 |

fraud firms[16] from Beasley (1996) were added to the remaining 197 fraud firms, for a total of 272 fraud firms. From these 272 fraud firms, 221 firms[17] with missing Compustat (financial statement data), Compact D/SEC (executive and director names, titles, and company holdings), or I/B/E/S (one-year-ahead analyst earnings per share forecasts and actual earnings per share) data needed to create the measures used in this study were deleted from the sample. To these remaining 51 fraud firms, 15,934 nonfraud firm years[18] were added to obtain P(fraud) $\approx$ 0.003 (0.00319).

### *Object Features: Financial Statement Fraud Predictors*

Financial statement fraud predictor research has evaluated a large number of potential financial statement fraud predictors. The experiment included predictors that were found to be significant in prior research and that were available from electronic sources. Other variables were excluded, since they were less likely to be used in practice due to the difficulty in obtaining them. See Table 3 for

[16] These 75 fraud observations were kindly provided by Mark Beasley. Beasley (1996) collected the data from 348 AAERs released between 1982 and 1991 (67 observations) and from the *Wall Street Journal* Index caption of "Crime—White Collar Crime" between 1980 and 1991 (eight observations).

[17] Most of the firms deleted due to missing data were older observations (for example, 74 of the 75 firms received from Beasley [1996] were deleted) or very small firms, for which I/B/E/S (in particular) and Compact D/SEC, and to some extent Compustat, are sparse.

[18] Note that matching is typically used to increase internal validity by controlling for variables not manipulated or measured in the experiment. However, the goal of this research is not to improve the understanding of factors that explain financial statement fraud, but rather to establish what classification algorithms and predictors are useful in predicting financial statement fraud. A dataset was, therefore, created in which the performance impact of various *training* prior fraud probabilities could be examined. Assuming that a lower prior fraud probability in the training data than what is used for evaluation purposes will not improve performance when the minority class is already sparse, the maximum number of nonfraud firms needed is equal to the number of fraud firms divided by the lowest prior fraud probability tested minus the number of fraud firms, i.e., (51/0.003) − 51 = 16,949. Higher prior fraud probabilities can then be obtained for training purposes by undersampling the majority class.

**TABLE 3**

**Fraud Predictors[a]**

| Predictor | Definition[b] | Data Source | Reference |
|---|---|---|---|
| Accounts Receivable | (data2) | Compustat | Green and Choi (1997); Lin et al. (2003) |
| Accounts Receivable to Sales | (data2/data12) | Compustat | Green and Choi (1997); Feroz et al. (2000); Lin et al. (2003); Kaminski et al. (2004) |
| Accounts Receivable to Total Assets | (data2/data6) | Compustat | Green and Choi (1997); Lin et al. (2003) |
| AFDA | (data67) | Compustat | Green and Choi (1997); Lin et al. (2003) |
| AFDA to Accounts Receivable | (data67/data2) | Compustat | Green and Choi (1997); Lin et al. (2003) |
| AFDA to Net Sales | (data67/data12) | Compustat | Green and Choi (1997); Lin et al. (2003) |
| Altman Z-score | 3.3 $*$ (data18 + data15 + data16)/data6 + 0.999 $*$ data12/data6 + 0.6 $*$ data25 $*$ data199/data181 + 1.2 $*$ data179/data6 + 1.4 $*$ data36/data6 | Compustat | Feroz et al. (2000) |
| Big 4 auditor | IF 0 $<$ data149 $<$ 9 then 1 else 0 | Compustat | Fanning and Cogger (1998) |
| Current minus Prior Year Inventory to Sales | (data3)/(data12) $-$ (data3$_{t-1}$)/(data12$_{t-1}$) | Compustat | Summers and Sweeney (1998) |
| Days in Receivables Index[c] | (data2/data12)/(data2$_{t-1}$/data12$_{t-1}$) | Compustat | Beneish (1997); Chen and Sennetti (2005) |
| Debt to Equity | (data181/data60) | Compustat | Fanning and Cogger (1998) |
| Demand for Financing (*ex ante*) | IF ((data308 $-$ (data128$_{t-3}$ + data128$_{t-2}$ + data128$_{t-1}$)/3)/(data4) $<$ $-0.5$ then 1, else 0 | Compustat | Dechow et al. (1996) |
| Declining Cash Sales dummy | IF (data12 $-$ (data2 $-$ data2$_{t-1}$) $<$ (data12$_{t-1}$ $-$ (data2$_{t-1}$ $-$ data2$_{t-2}$) then 1, else 0 | CompactD | Beneish (1997) |
| Evidence of CEO Change[d] | IF CEO_Name $<>$ CEO_Name$_{t-1}$ OR CEO_Name$_{t-1}$ $<>$ CEO_Name$_{t-2}$ OR CEO_Name$_{t-2}$ $<>$ CEO_Name$_{t-3}$ then 1, else 0 | CompactD | Dechow et al. (1996); Feroz et al. (2000) |
| Evidence of CFO Change[e] | IF CFO_Name $<>$ CFO_Name$_{t-1}$ OR CFO_Name$_{t-1}$ $<>$ CFO_Name$_{t-2}$ OR CFO_Name$_{t-2}$ $<>$ CFO_Name$_{t-3}$ then 1, else 0 | CompactD | Fanning and Cogger (1998); Feroz et al. (2000) |
| Fixed Assets to Total Assets | data7/data6 | Compustat | Kaminski et al. (2004) |

*(continued on next page)*

**TABLE 3** (continued)

| Predictor | Definition[b] | Data Source | Reference |
|---|---|---|---|
| Four-Year Geometric Sales Growth Rate[f] | (data12/data12$_{t-3}$)^(1/4) − 1 | Compustat | Fanning and Cogger (1998); Bell and Carcello (2000) |
| Gross Margin[g] | (data12 − data41)/data12 | Compustat | Green and Choi (1997); Lin et al. (2003); Chen and Sennetti (2005) |
| Holding Period Return in the Violation Period | (data199 – data199$_{t-1}$)/data199 | Compustat | Beneish (1999) |
| Industry ROE minus Firm ROE | data172/data 60 | Compustat | Feroz et al. (2000) |
| Insider Holdings to Total Board Holdings | SUM(IF relationship code = CB, D, DO, H, OD then Insider_Holdings, else 0)/ SUM(Insider_Holdings) | CompactD | Dechow et al. (1996) |
| Inventory to Sales[h] | data3/data12 | Compustat | Kaminski et al. (2004) |
| Net Sales | data12 | Compustat | Green and Choi (1997); Lin et al. (2003) |
| Positive Accruals dummy | IF (data18 − data308) > 0 and (data18$_{t-1}$ − data308$_{t-1}$) > 0 then 1, else 0 | Compustat | Beneish (1997) |
| Percentage Officers on the Board of Directors[i] | SUM(IF Executive_Name = Director_Name then 1, else 0)/ Number_Of_Directors | CompactD | Beasley (1996); Dechow et al. (1996); Fanning and Cogger (1998); Uzun et al. (2004) |
| Prior Year ROA to Total Assets Current Year | (data172$_{t-1}$/data6$_{t-1}$)/data6 | Compustat | Summers and Sweeney (1998) |
| Property Plant and Equipment to Total Assets | data8/data6 | Compustat | Fanning and Cogger (1998) |
| Sales to Total Assets | data12/data6 | Compustat | Fanning and Cogger (1998); Kaminski et al. (2004); Chen and Sennetti (2005) |
| the Number of Auditor Turnovers | IF data149 < > data149$_{t-1}$ then 1, else 0 + IF data149$_{t-1}$ < > data149$_{t-2}$ then 1, else 0 + IF data149$_{t-2}$ < > data149$_{t-3}$ then 1, else 0 | Compustat | Feroz et al. (2000) |
| Times Interest Earned | (data18 + data15 + data16)/data15 | Compustat | Feroz et al. (2000) |
| Total Accruals to Total Assets[j] | (data18 − data308)/data6 | Compustat | Beneish (1997); Dechow et al. (1996); Beneish (1999); Lee et al. (1999) |
| Total Debt to Total Assets | data181/data6 | Compustat | Beneish (1997); Dechow et al. (1996); Lee et al. (1999) |

*(continued on next page)*

**TABLE 3** (continued)

| Predictor | Definition[b] | Data Source | Reference |
|---|---|---|---|
| Total Discretionary Accrual | $DA_{t-1} + DA_{t-2} + DA_{t-3}$, where, DA = TA/A-estimated(NDA); TA/A = (data18 − data308) /$data6_{t-1}$; NDA = 1/$data6_{t-1}$ + (data12 − $data12_{t-1}$ − data2 + $data2_{t-1}$)/$data6_{t-1}$ + (data308 − $data308_{t-1}$)/$data6_{t-1}$ + data7/$data6_{t-1}$ | Compustat | Perols and Lougee (2009) |
| Unexpected Employee Productivity | FIRM((data12/data29 − $data12_{t-1}$/$data29_{t-1}$)/($data12_{t-1}$/$data29_{t-1}$)) − INDUSTRY((data12/data29 − $data12_{t-1}$/$data29_{t-1}$)/($data12_{t-1}$/$data29_{t-1}$)) | Compustat | Perols and Lougee (2009) |
| Value of Issued Securities to Market Value | IF data396 > 0 then data396 * data199/(data25 * data199) else IF (data25 − $data25_{t-1}$) > 0 then ((data25 − $data25_{t-1}$) * data199)/(data25 * data199), else 0 | Compustat | Dechow et al. (1996) |
| Whether Accounts Receivable > 1.1 of Last Year's | IF (data2/$data2_{t-1}$) > 1.1 then 1, else 0 | Compustat | Fanning and Cogger (1998) |
| Whether Firm was Listed on AMEX | IF ZLIST = 5, 15, 16, 17, 18 then 1, else 0 | Compustat | Lee et al. (1999) |
| Whether Gross Margin Percent > 1.1 of Last Year's | IF ((data12 − data41)/data12)/(($data12_{t-1}$ − $data41_{t-1}$)/$data12_{t-1}$) > 1.1 then 1, else 0 | Compustat | Fanning and Cogger (1998) |
| Whether LIFO | IF data59 = 2 then 1, else 0 | Compustat | Fanning and Cogger (1998) |
| Whether Meeting or Beating Analyst Forecast | IF EPS − Analyst_Forecast >= 0 then 1, else 0 | I/B/E/S | Perols and Lougee (2009) |
| Whether New Securities were Issued | IF (data25 − $data25_{t-1}$) > 0 OR data396 > 0 then 1, else 0 | Compustat | Dechow et al. (1996); Lee et al. (1999) |
| Whether SIC Code Larger (Smaller) than 2999 (4000) | IF 2999 < DNUM < 4000 then 1, else 0 | Compustat | Lee et al. (1999) |

[a] All predictors found to be significant determinants of financial statement fraud in prior research and that were relatively easy to obtain were included in the experiment. *AFDA* refers to allowance of doubtful accounts; *ROE* refers to return on equity; *ROA* refers to return on assets; and *LIFO* refers to last in, first out.

[b] data# refers to specific items in Compustat based on the numbering system in existence as of April 17, 2008; *t* is the first fraud year and *t*–1, *t*–2, *t*–3, and *t*–4 are each of the four years leading up to the first fraud year *t*.

[c] Days in receivables index (Beneish 1997) was included in the experiment. Accounts receivable turnover (Chen and Sennetti 2005) was excluded because of its similarity to days in receivables index.

[d] Evidence of CEO change was included in the experiment based on its similarity to number of CEO turnovers (Feroz et al. 2000) and whether the CEO is the founder (Dechow et al. 1996), which were excluded.

[e] Evidence of CFO change (Fanning and Cogger 1998) was included in the experiment. The number of CFO turnovers (Feroz et al. 2000) was excluded because of its similarity to evidence of CFO change.

**TABLE 3** (continued)

[f] Four-year geometric sales growth rate (Fanning and Cogger 1998) was included in the experiment. Rapid company growth (Bell and Carcello 2000) was excluded because of its similarity to four-year geometric sales growth rate.
[g] Gross margin (Green and Choi 1997; Lin et al. 2003; Chen and Sennetti 2005) was included in the experiment. Net profit margin (Chen and Sennetti 2005) was excluded because of its similarity to gross margin.
[h] Inventory to sales (Kaminski et al. 2004) was included in the experiment. Inventory to current assets (Kaminski et al. 2004) was excluded because of its similarity to inventory to sales.
[i] Percentage officers on the board of directors (Dechow et al. 1996) was included in the experiment. Percentage outside directors (Beasley 1996; Fanning and Cogger 1998, Uzun et al. 2004) and whether board has over 50 percent inside directors (Dechow et al. 1996) were excluded because of their similarity to percentage officers on the board of directors.
[j] Total accruals to total assets (Beneish 1997) was included in the experiment. Total accruals in year of manipulation (Lee et al. 1999) and discretionary accruals in violation period (Dechow et al. 1996; Beneish 1999) were excluded because of their similarity to total accruals to total assets.

the final selection of the 42 predictors included in the experiment and how these predictors are calculated.

## PREPROCESSING AND EXPERIMENTAL PROCEDURES

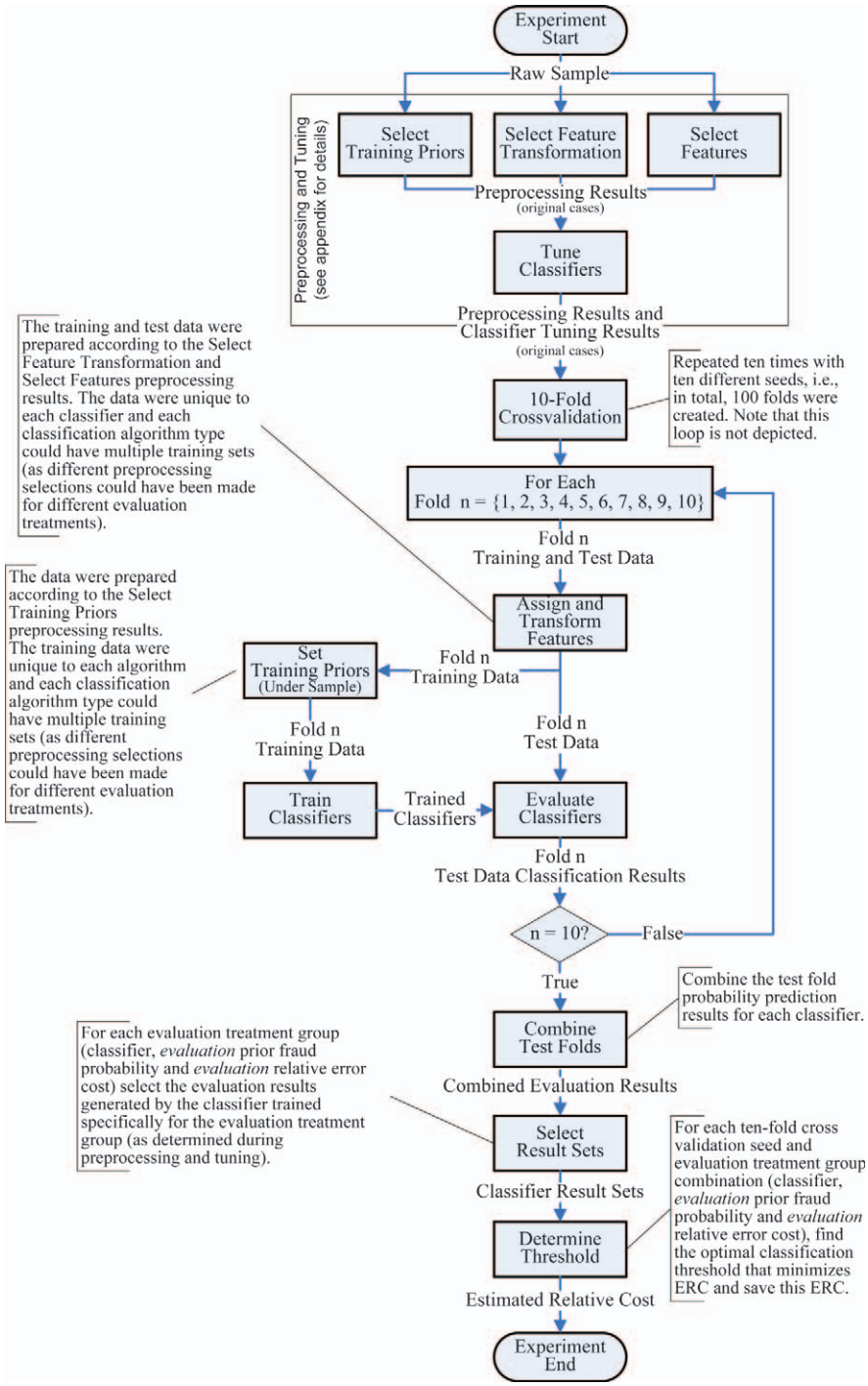
### Preprocessing

Before comparing the classification algorithms, four preprocessing procedures were performed (see Figure 1 for an overview). Using ten-fold stratified cross-validation,[19] the performance of the classifiers was first examined after training the classifiers on data with ten different *training* prior fraud probability levels: 0.3,[20] 0.6, 1, 1.5, 2.5, 5, 10, 20, 40, and 60[21] percent. The performance of the classifiers was also examined, again using ten-fold stratified cross-validation, after normalizing, discretizing, standardizing, and not filtering the continuous fraud predictor variables. In the third preprocessing step, the relative utility of the different fraud predictors to each classifier was examined using ten-fold cross-validation. For classification algorithm-specific tuning, a total of 72 different classifier configurations were evaluated using ten-fold stratified cross-validation.

The preprocessing steps were performed somewhat independently in that different preprocessing combinations were not examined. For example, when examining the training prior fraud probability (preprocessing step 1), the different feature transformation methods (preprocessing step 2) were not examined at each training prior fraud probability level. In each preprocessing step, except in the feature selection, optimal classification thresholds were empirically determined, and the best (one of each) training prior fraud probability, filtering method, and classification

[19] In ten-fold stratified cross-validation, the size of each fold and the ratio of fraud and nonfraud cases are held approximately constant across folds by randomly sampling within each class. In regular ten-fold cross-validation, only the size of the fold is held approximately constant, and fraud and nonfraud firms are not separated before being randomly sampled. Each fold will, therefore, on average, contain five fraud firms, but this number can be lower or higher. Given the relatively small number of fraud firms, to ensure that each fold contains at least a few fraud firms, ten-fold stratified cross-validation was used in this study (note that the sampling within the two classes is still random).
[20] The lowest *evaluation* prior fraud probability adjusted for the highest relative error cost of 1:1 is 0.3 percent, calculated as $51/(51 + (1–0.003) * 51/(0.003 * 1))$.
[21] The highest *evaluation* prior fraud probability adjusted for the lowest relative error cost of 1:100 is 54.8 percent, calculated as $51/(51 + (1–0.012) * 51/(0.012 * 100))$.

**FIGURE 1**
**Overview of Experimental Procedures**



algorithm tuning configuration were selected for each evaluation treatment group, i.e., classification algorithm, evaluation prior fraud probability, and evaluation relative error cost treatment combination. The entire data sample was used in the preprocessing steps to ensure that enough data were available to determine what training prior fraud probabilities and training classification error costs should be used for training classifiers (Research Question 2), and what predictors are useful to these algorithms (Research Question 3). However, note that ten-fold cross-validation was used in each preprocessing step. The same data were, therefore, never at the same time used for both training and evaluating the classifiers. Please refer to the Appendix for preprocessing and classifier tuning procedure details.

Table 4 shows the summarized preprocessing results (details tabulated in the Appendix). The preprocessing results show that selected training prior fraud probabilities are increasing in evaluation prior fraud probabilities and decreasing in evaluation relative error costs (i.e., increasing in the cost of a false negative classification holding the cost of a false positive classification constant). With regard to transforming the data, data normalization and no transformation generally provide the best results. The attribute selection results show that Big 4 auditor, the most frequently selected attribute, is selected by all classification algorithms except for stacking. Auditor turnover, total discretionary accruals, and accounts receivable are selected by four of six classification algorithms, while meeting or beating analyst forecasts and unexpected employee productivity are selected by three of six classification algorithms. All the other predictors are selected by less than 50 percent of the classification algorithms.

### Classification Algorithm Comparison Experiment

To evaluate the classification algorithms, ten-fold stratified cross-validation runs were repeated ten times on the entire dataset, each time using a different sampling seed (see Figure 1).

**TABLE 4**

**Preprocessing Result Overview: Selected *Training* Prior Fraud Probabilities, Data Filtering Methods, and Predictors**

| Classifiers | Training Prior Fraud Probability | Data Filtering | Predictors[a] |
|---|---|---|---|
| ANN | 0.6%, 60% | Normalize | 2, 3, 4, 6, 9, 12, 16, 20, 21, 31 |
| SVM | 20%, 60% | Normalize | 1, 2, 3, 4, 5, 8, 13, 14, 22, 24, 33, 35 |
| C4.5 | 5%, 10%, 40%, 60% | No Filter, Standardize | 1, 3, 6, 7, 8, 19, 22, 27, 28 |
| Logistic | 1.5%, 10%, 20% | Normalize | 1, 2, 3, 4, 5, 8, 11, 12, 23 |
| Bagging | 60% | No Filter, Normalize | 1, 2, 3, 7, 12, 18, 27 |
| Stacking | 60% | No Filter, Normalize | 4, 9, 10, 11, 17, 18, 19, 25 |

[a] Predictor numbers represent the following predictors: 1 = number of auditor turnovers; 2 = total discretionary accruals; 3 = Big 4 auditor; 4 = accounts receivable; 5 = allowance for doubtful accounts; 6 = accounts receivable to total assets; 7 = accounts receivable to sales; 8 = whether meeting or beating forecast; 9 = evidence of CEO change; 10 = sales to total assets; 11 = inventory to sales; 12 = unexpected employee productivity; 13 = Altman Z-score; 14 = percentage of executives on the board of directors; 16 = whether accounts receivable grew by more than 10 percent; 17 = allowance for doubtful accounts to net sales; 18 = current minus prior year inventory to sales; 19 = gross margin to net sales; 20 = evidence of CFO change; 21 = holding period return in the violation period; 22 = property plant and equipment to total assets; 23 = value of issued securities to market value; 24 = fixed assets to total assets; 25 = days in receivables index; 27 = industry ROE minus firm ROE; 28 = positive accruals dummy; 31 = whether gross margin grew by more than 10 percent; 33 = allowance for doubtful accounts to accounts receivable; and 35 = total debt to total assets.

For each seed and the ten folds within each seed, both the training and test sets were adjusted based on the attribute selection and attribute transformation preprocessing results. The training data, but not the test data, were filtered based on the training prior fraud probability preprocessing results, and the classifiers were tuned based on the classifier tuning preprocessing results. The preprocessing results used in the classification algorithm comparison experiment are summarized in Table 4.[22] The classifiers were then trained using the training data and evaluated using the evaluation data.

After the ten-fold cross-validation for the last seed had been completed, the results from the ten test folds for each seed were combined and optimal thresholds were determined and used to calculate ten ERC scores for each classifier, evaluation relative error cost and evaluation prior fraud probability combination.[23] Note that for each treatment level, the best, as determined during preprocessing, classifier tuning configuration, attribute transformation, and training prior fraud probability was used. These preprocessing combinations were, however, not necessarily the combinations that produced the best ERC for the test folds, as different random seeds were used during evaluation. This evaluation procedure generated a final result set containing ten observations (one observation for each ten-fold cross-validation seed) per classification algorithm type, evaluation relative error costs, and evaluation prior fraud probability treatment group.

By using the entire data sample in the preprocessing steps, and then using the same data and the results from the preprocessing steps to evaluate the classification algorithms, the classification algorithms are compared under near optimal performance for each classification algorithm given the examined attributes, training prior fraud probabilities, attribute transformation methods, and classifier configurations. Furthermore, during preprocessing and evaluation, the same dataset is never used at the same time to both train and evaluate the classifiers; instead, ten-fold cross-validation is used. This should improve the generalizability of the results, with the assumption that it is possible to select the best attributes, training prior fraud probability,

[22] To evaluate the appropriateness of the classifier preprocessing performed and to give insights into whether these preprocessing steps provide utility to the classifiers, an additional experiment was performed. In this experiment, logistic regression, one of the best performing classifiers (see "Results" section), was compared to a logistic regression model as implemented in Feroz et al. (2000), i.e., logistic regression with the attributes used by Feroz et al. (2000) and without altering the *training* prior fraud probability or transforming the attributes. However, to put the two implementations on an even playing field, optimal thresholds at the different *evaluation* relative error cost and *evaluation* prior fraud probability levels were determined and used for both classifiers. Furthermore, the original data sample was randomly split in two, with each subset containing approximately half the fraud cases and half the nonfraud cases. One of the subsets was then used to select *training* prior fraud probability attributes, attribute transformation, and classifier tuning configurations, while the second subset was used to evaluate the two classifiers. The experiment used regular ten-fold cross-validation instead of ten-fold stratified cross-validation (to assess whether it is possible that the reported results in the main experiment are biased due to the usage of stratified rather than regular ten-fold cross-validation). The experiments were otherwise performed as described for the main experiment. A regression model, with a dummy variable for the two classifiers and blocking variables for the effects of the different *evaluation* treatment levels, was used to evaluate the relative performance of the two implementations. Using two-tailed p-values, the results showed that the logistic regression implementation following the preprocessing steps outlined in this paper significantly ($p = 0.012$) outperformed the comparison model.

[23] Although different sampling seeds were used for each of the ten ten-fold cross-validation rounds, each cross-validation round used the entire dataset, and the results from the different rounds were, thus, not truly independent. Following Witten and Frank (2005), all reported t-statistics were, therefore, adjusted by changing the t-statistic denominator from $\sqrt{(\sigma 2/k)}$ to $\sqrt{(\sigma 2 * (1/k + 0.1/0.9))}$. This adjustment reduces the probability of committing a Type I error, i.e., concluding that there is a performance difference between two classification algorithms, when in fact there is no difference.

**TABLE 5**

**Descriptive Statistics of Classifier ERC[a]**

| Classifier | Min | Median | Max | Mean | Standard Deviation |
|---|---|---|---|---|---|
| Logistic | 0.0026 | 0.2167 | 0.9100 | 0.2916 | 0.2367 |
| Bagging | 0.0028 | 0.2400 | 0.8858 | 0.2978 | 0.2275 |
| SVM | 0.0025 | 0.2306 | 0.8946 | 0.2989 | 0.2453 |
| ANN | 0.0030 | 0.2400 | 0.8912 | 0.3046 | 0.2320 |
| C4.5 | 0.0028 | 0.2400 | 1.0614 | 0.3301 | 0.2730 |
| Stacking | 0.0030 | 0.2400 | 0.9880 | 0.3414 | 0.2905 |

[a] Classifier performance is measured using ERC. Note that lower values are preferred over higher values.

attribute transformation method, and classifier tuning configuration for each classification algorithm.[24]

## RESULTS

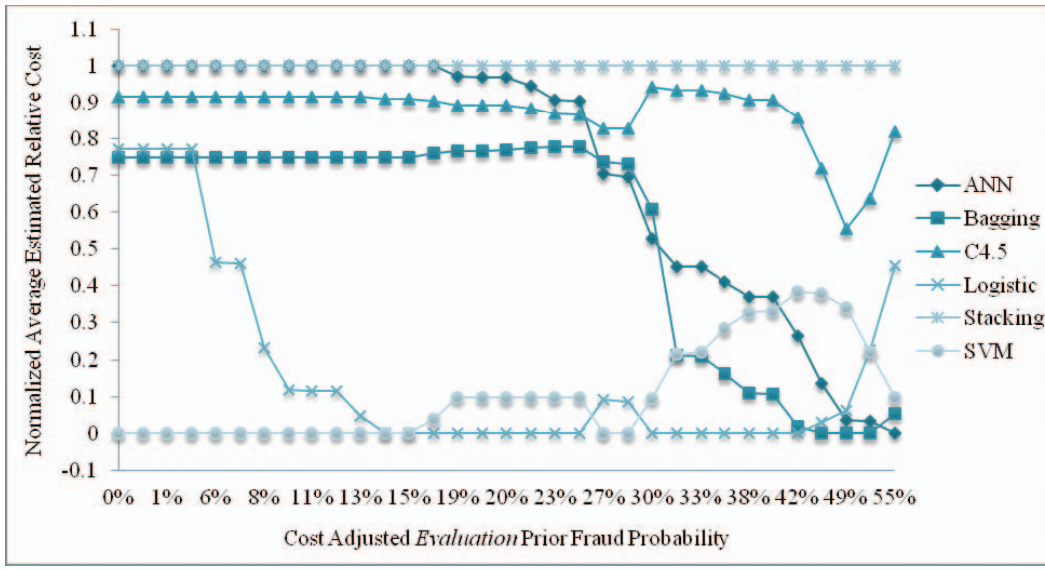
Table 5 contains descriptive classifier performance statistics. The reported ERC is the average for each classification algorithm across all treatment levels. Thus, it is not surprising that the range of ERC is high and that the standard deviation is almost as high as the mean. For example, the standard deviation and mean ERC for logistic regression are 0.2367 and 0.2916, respectively. The descriptive statistics provide an initial indication that logistic regression, bagging, and SVM perform well. It is, however, important to remember that these statistics report on the performance of the classification algorithms on average and that they do not show under what specific evaluation conditions logistic regression, bagging, and SVM outperform the other classification algorithms.

Figure 2 shows the performance of the classification algorithms after normalizing ERC within each evaluation prior fraud probability and evaluation relative error cost treatment group. In Figure 2, the normalized performance of the classification algorithms is shown against a cost adjusted evaluation prior fraud probability.[25] As seen in Figure 2, when the cost adjusted

[24] To examine the generalizability of the results to situations where it is not possible to select the best attributes, *training* prior fraud probability, attribute transformation method, and classifier tuning configuration for each classifier, a second additional experiment was performed. In this experiment, the original dataset was randomly split in two, with each subset containing approximately half the fraud cases and nonfraud cases. One of the subsets was then used to select the attributes, *training* prior fraud probability, attribute transformation method, and classifier tuning configuration for the different classifiers, while the second subset was used to evaluate the classifiers. Furthermore, only two classifiers, logistic regression and ANN, were compared in this experiment, and the experiment used regular ten-fold cross-validation instead of ten-fold stratified cross-validation (to assess whether it is possible that the reported results in the main experiment are biased due to the usage of stratified rather than regular ten-fold cross-validation). The experiment was otherwise performed as described for the main experiment. Using two-tailed p-values, the results were similar to those of the main experiment, with logistic regression significantly (moderately) outperforming the ANN (1) at *evaluation* prior fraud probability treatment levels of (0.003), 0.006, and 0.009, when blocking for *evaluation* relative error cost, and (2) at *evaluation* relative error costs treatment levels of 1:30, 1:40, 1:50, 1:60, 1:70, 1:80, and 1:90, when controlling for *evaluation* prior fraud probability. There was no significant difference between the two classifiers at *evaluation* relative error costs treatment levels of 1:1, 1:10, 1:20, and 1:100.

[25] The evaluation prior fraud probability is adjusted for the different evaluation relative error cost levels based on the equivalence noted in Breiman et al. (1984). Please refer to footnote 12 for a discussion of this equivalence.

**FIGURE 2**
**Classifier Performance—Normalized Average Estimated Relative Cost by Cost Adjusted *Evaluation* Prior Fraud Probability**




evaluation prior fraud probability is between 0 and 13 percent, SVM appears to outperform all the other classification algorithms, followed by logistic regression, and then bagging. In this range, ANN and stacking appear to perform particularly poorly. In the cost adjusted evaluation prior fraud probability range from 13 to 30 percent, logistic regression and SVM appear to outperform all the other classification algorithms, again followed by bagging. In higher cost adjusted evaluation prior fraud probability ranges, logistic regression appears to perform relatively well between 30 and 49 percent, while bagging appears to perform relatively well between 42 and 55 percent. In addition to logistic regression, the ANN also appears to perform well in the very highest range (49 to 55 percent), and even appears to dominate all the other classification algorithms at 55 percent.

Based on this graphical analysis and the descriptive statistics, it appears that logistic regression is a robust performer that either outperforms or performs on par with the other classification algorithms. SVM appears to provide good performance over relevant ranges, but does not appear to provide a performance advantage over logistic regression, except for at very low cost adjusted evaluation prior fraud probability levels. Finally, bagging and ANN appear to perform relatively well at certain, though perhaps less relevant, ranges, which explains why bagging and ANN performed relatively well on average (Table 5).

To more formally examine the relative performance of the classification algorithms at different levels of evaluation prior fraud probability and evaluation relative error cost, three evaluation relative error cost groups (henceforth, cost groups), low (1:80, 1:90, and 1:100), medium (1:30, 1:40, 1:50, 1:60, and 1:70), and high (1:1, 1:10, and 1:20) were created. With these three cost groups and the three original evaluation prior fraud probability levels, nine treatment groups were created. An ANOVA model, using the classifier algorithm factor as the main effect and blocking for evaluation relative error cost, was then examined within each of these nine treatment groups:

$$ERC = \alpha_0 + \alpha_1 \text{Classification Algorithm} + block + \varepsilon \quad (2)$$

The *post hoc* analysis using Tukey-Kramer HSD is reported in Table 6. Table 6 contains three panels showing the relative performance results at different evaluation prior fraud probabilities. Each panel shows the results for all three cost groups, i.e., low (1:80–1:100), medium (1:30–1:70), and high (1:1–1:20). Thus, a total of nine different result sets (three cost groups in each of the three panels) are shown in Table 6. Each result set shows the relative performance of the classifiers using a connected letters report, which rank orders the classifiers according to their relative performance (the bottom classifier in each result set has the lowest ERC, i.e., the best performance). The performance difference between two classification algorithms is significant[26] when they are *not* connected by the same letter.

The results in Table 6 corroborate the findings presented in Figure 2. More specifically, SVM significantly outperforms all other classification algorithms, and logistic regression significantly outperforms stacking and ANN when the evaluation prior fraud probability is low and evaluation relative error costs is high. Logistic regression and SVM, furthermore, significantly outperform all the other classification algorithms at: (1) low and medium evaluation relative error costs when the prior fraud probability is 0.3 percent; (2) medium and high evaluation relative error costs when the evaluation prior fraud probability is 0.6 percent; and (3) high evaluation relative error cost when the evaluation prior fraud probability is 1.2 percent. Additionally, logistic regression significantly outperforms all the other classification algorithms except for bagging at: (1) low evaluation relative error costs when the evaluation prior fraud probability is 0.6 percent; and (2) medium evaluation relative error costs when the evaluation prior fraud probability is 1.2 percent. At a low evaluation relative error cost and a high evaluation prior fraud probability, stacking and C4.5 perform significantly worse than all the other classification algorithms. Overall, logistic regression and SVM perform well at all evaluation prior fraud probability and relative error cost levels.

As described earlier, relative error costs between 1:20 and 1:40 and prior fraud probability of 0.6 percent are believed to be good estimates of actual costs and prior probabilities associated with financial statement fraud (Bell and Carcello 2000; Bayley and Taylor 2007). Assuming that these are good estimates of actual fraud costs and prior probabilities, the ANOVA Model (2), used earlier, was also examined within each of these three treatment groups.

The *post hoc* analysis using Tukey-Kramer HSD and pairwise t-test, reported in Table 7, show that logistic regression and SVM consistently outperform the other classification algorithms at what are believed to be good estimates of actual real-world prior fraud probability and relative error cost.

## DISCUSSION

The experiments show that logistic regression, a well known and established classification algorithm, and SVM outperform or perform as well as a relatively representative set of classification algorithms. This finding is somewhat surprising considering that prior fraud research typically concludes that ANN outperforms logistic regression. However, this study differs from prior fraud studies in that it (1) evaluates the classification algorithms using a highly imbalanced dataset; (2) manipulates the *evaluation* prior fraud probability and *evaluation* relative error cost and examines the performance of the classification algorithms using optimal classification threshold levels, *training* prior fraud probabilities, *training* relative error costs, and attribute transformation for the different classification algorithms given specific *evaluation* manipulations; and (3) compares classification algorithms not only by including a relatively complete set of attributes, but also by using a wrapper method to select attributes for each classifier. Thus, while the result that logistic

[26] Significance measured at a p-value of 0.05; all reported p-values are two-tailed.

**TABLE 6**

**Classifier Performance Comparison Tukey-Kramer HSD Connected Letters Report[a]**

**Panel A: Prior Fraud Probability = 0.3%**

| Relative Error Cost Range | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **1:1–1:20 (High)** | | | | **1:30–1:70 (Medium)** | | | | **1:80–1:100 (Low)** | | | |
| **Classifier** | **Tukey** | | | **Classifier** | **Tukey** | | | **Classifier** | **Tukey** | | |
| ANN | A | | | ANN | A | | | Stacking | A | | |
| Stacking | A | | | Stacking | A | | | ANN | A | | |
| C4.5 | A | B | | C4.5 | A | B | | C4.5 | A | | |
| Bagging | A | B | | Bagging | | B | | Bagging | A | | |
| Logistic | | B | | Logistic | | | C | SVM | | B | |
| SVM | | | C | SVM | | | C | Logistic | | B | |

**Panel B: Prior Fraud Probability = 0.6%**

| Relative Error Cost Range | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **1:1–1:20 (High)** | | | | **1:30–1:70 (Medium)** | | | | **1:80–1:100 (Low)** | | | |
| **Classifier** | **Tukey** | | | **Classifier** | **Tukey** | | | **Classifier** | **Tukey** | | |
| ANN | A | | | Stacking | A | | | Stacking | A | | |
| Stacking | A | | | C4.5 | A | B | | C4.5 | A | | |
| C4.5 | A | | | ANN | A | B | | ANN | | B | |
| Bagging | A | | | Bagging | | B | | SVM | | B | |
| Logistic | | B | | SVM | | | C | Bagging | | B | C |
| SVM | | B | | Logistic | | | C | Logistic | | | C |

**Panel C: Prior Fraud Probability = 1.2%**

| Relative Error Cost Range | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| **1:1–1:20 (High)** | | | | **1:30–1:70 (Medium)** | | | | **1:80–1:100 (Low)** | | | |
| **Classifier** | **Tukey** | | | **Classifier** | **Tukey** | | | **Classifier** | **Tukey** | | |
| Stacking | A | | | Stacking | A | | | Stacking | A | | |
| ANN | A | | | C4.5 | A | | | C4.5 | | B | |
| C4.5 | A | | | SVM | | B | | Logistic | | | C |
| Bagging | A | | | ANN | | B | | SVM | | | C |
| SVM | | B | | Bagging | | B | C | ANN | | | C |
| Logistic | | B | | Logistic | | | C | Bagging | | | C |

[a] The three panels show the relative performance results at different evaluation prior fraud probabilities. Each panel shows the results for all three cost groups, i.e., low (1:80–1:100), medium (1:30–1:70), and high (1:1–1:20). Thus, a total of nine different result sets (three cost groups in each of the three panels) are shown in Table 6. Each result set shows the relative performance of the classifiers using a connected letters report, which rank orders the classifiers according to their relative performance (the bottom classifier in each result set has the lowest ERC, i.e., the best performance). The performance difference between two classification algorithms is significant when they are not connected by the same letter. Significance measured at a p-value of 0.05 using Tukey-Kramer HSD and blocking for the effect of evaluation prior fraud probability and evaluation relative error cost on ERC; all reported p-values are two-tailed.

**TABLE 7**

**Classifier ERC at Best Estimates of Relative Error Cost and Prior Fraud Probability**

**Panel A: Prior Fraud Probability = 0.6% and Relative Error Cost = 1:20**

| Classifier | Tukey-Kramer HSD[a] | Pair-Wise t-tests | | | | |
|---|---|---|---|---|---|---|
| | | Logistic | SVM | Bagging | C4.5 | Stacking |
| ANN | A | 0.0100 (p = 0.0001) | 0.0113 (p < 0.0001) | 0.0028 (p = 0.1191) | 0.0009 (p = 0.5978) | 0.0000 (p = 1.000) |
| Stacking | A | 0.0100 (p = 0.0001) | 0.0113 (p < 0.0001) | 0.0028 (p = 0.1191) | 0.0009 (p = 0.5978) | |
| C4.5 | A | 0.009 (p = 0.0002) | 0.0104 (p = 0.0001) | 0.0019 (p = 0.2758) | | |
| Bagging | A | 0.0072 (p = 0.0012) | 0.0085 (p = 0.0003) | | | |
| SVM | B | 0.0013 (p = 0.4492) | | | | |
| Logistic | B | | | | | |

**Panel B: Prior Fraud Probability = 0.6% and Relative Error Cost = 1:30**

| Classifier | Tukey-Kramer HSD[a] | Pair-Wise t-tests | | | | |
|---|---|---|---|---|---|---|
| | | SVM | Logistic | Bagging | C4.5 | Stacking |
| ANN | A | 0.0169 (p < 0.0001) | 0.0169 (p < 0.0001) | 0.0042 (p = 0.1136) | 0.0015 (p = 0.5517) | 0.0000 (p = 1.0000) |
| Stacking | A | 0.0169 (p < 0.0001) | 0.0169 (p < 0.0001) | 0.0042 (p = 0.1136) | 0.0015 (p = 0.5517) | |
| C4.5 | A | 0.0154 (p = 0.0001) | 0.0154 (p = 0.0001) | 0.0027 (p = 0.2927) | | |
| Bagging | A | 0.0127 (p = 0.0003) | 0.0127 (p = 0.0003) | | | |
| Logistic | B | 0.0000 (p = 1.0000) | | | | |
| SVM | B | | | | | |

**Panel C: Prior Fraud Probability = 0.6% and Relative Error Cost = 1:40**

| Classifier | Tukey-Kramer HSD[a] | Pair-Wise t-tests | | | | |
|---|---|---|---|---|---|---|
| | | Logistic | SVM | Bagging | C4.5 | ANN |
| Stacking | A | 0.0243 (p < 0.0001) | 0.0219 (p = 0.0001) | 0.0056 (p = 0.1581) | 0.0026 (p = 0.5010) | 0.0007 (p = 0.8461) |
| ANN | A | 0.0235 (p = 0.0001) | 0.0212 (p = 0.0001) | 0.0049 (p = 0.2151) | 0.0019 (p = 0.6289) | |
| C4.5 | A | 0.0217 (p = 0.0001) | 0.0193 (p = 0.0003) | 0.0031 (p = 0.4304) | | |
| Bagging | A | 0.0186 (p = 0.0004) | 0.0163 (p = 0.0011) | | | |
| SVM | B | 0.0024 (p = 0.5381) | | | | |
| Logistic | B | | | | | |

[a] Classification algorithms not connected by the same letter are significantly different at a p-value of 0.05 using Tukey-Kramer HSD and blocking for the effect of *evaluation* prior fraud probability and *evaluation* relative error cost on ERC.

regression and SVM outperform or perform as well as the other classification algorithms is somewhat surprising, it does not necessarily contradict prior research findings. Rather, the results show that when taking these additional factors into account, logistic regression and SVM perform well in the fraud domain. In fact, when taking a closer look at the prior fraud research that compares ANN and logistic regression (Feroz et al. 2000; Lin et al. 2003), these studies provide, although they conclude that ANN performs better than logistic regression, some initial indications that logistic regression is a relatively good performer under certain circumstances. More specifically, as discussed in the "Related Research" section, these studies find that logistic regression outperforms ANN when the prior fraud probability is 0.01 and the relative cost of false positive and false negative classifications is from 1:1 to 1:30 fraud, ranges examined in this study.

A potential explanation as to why logistic regression (and SVM that establish proper probability estimates by fitting logistic regression models to the output) performs well is that logistic regression produces relatively accurate probability estimates (Perlich et al. 2003). Since the probability estimates generated by the different classifiers are compared to various thresholds to find the threshold that minimizes ERC, the relative performance of logistic regression will be better than if performance is measured using classification results based on the default threshold of 0.5, which has been used in most prior fraud research (Fanning and Cogger 1998; Feroz et al. 2000; Lin et al. 2003; Kotsiantis et al. 2006; Kirkos et al. 2007). Another potential explanation as to why logistic regression performs well is that logistic regression performs relatively well when it is difficult to separate signal from noise (Perlich et al. 2003). However, the area under the curve for logistic regression (AUC = 0.823), the measure of signal separability used in Perlich et al. (2003), is between the low- and high-separability groups found in their study.

Although the results are somewhat surprising, the experimental findings are encouraging since neither logistic regression nor SVM require extensive tuning and do not require a large amount of computing resources for training and evaluation purposes. Furthermore, logistic regression is widely used and accepted and produces results that are relatively easy to interpret.

The results show that classifiers can benefit from having the prior fraud probability in the *training* sample adjusted for different assumptions about *evaluation* prior fraud probability and *evaluation* relative error cost in the population. As the assumed evaluation prior fraud probability decreases and relative error cost increases, the classifiers can benefit from lowering the training prior fraud probability. Furthermore, out of 42 variables that have been found to be good predictors in prior fraud research, logistic regression uses a subset of only nine variables: auditor turnover, total discretionary accruals, Big 4 auditor, accounts receivable, allowance for doubtful accounts, meeting or beating analyst forecasts, inventory to sales, unexpected employee productivity, and value of issued securities to market value. Across all classification algorithms, only six variables are selected by three or more classification algorithms: auditor turnover, total discretionary accruals, Big 4 auditor, accounts receivable, meeting or beating analyst forecasts, and unexpected employee productivity.

A limitation of this study is that the entire data sample was used in both preprocessing and classification algorithm evaluation. The entire sample was used in both steps to ensure that enough data were available to determine what training prior fraud probabilities and training classification error costs should be used for training classifiers, and what predictors are useful to these algorithms. Furthermore, by using the entire sample in both steps, the classification algorithms could be compared under near optimal performance for each classification algorithm given the examined attributes, training prior fraud probabilities, attribute transformation methods, and classifier configurations. This is, nevertheless, a limitation that makes it more difficult to assess the generalizability of the relative classifier performance results to situations where it is not possible to select the best attributes, training prior fraud probability, attribute transformation method, and classifier tuning configuration for each classifier. To address this limitation, an additional

experiment was performed (see footnote 24) whereby the original dataset was randomly split in two. The first subset was used for preprocessing and the second subset was used for classification algorithm comparison, in which ANN and logistic regression were compared. The results in this additional experiment corroborate the result in the main experiment and, thus, provide empirical support for the generalizability of the results in the main experiment.

The preprocessing and classification algorithm results can be used by practitioners as guidance for selecting training prior fraud probabilities, attribute transformation methods, attributes, and classification algorithms when building fraud detection models. Improvement in fraud detection models can be useful to auditors during client selection, audit planning, and analytical procedures. Furthermore, the SEC can leverage the findings to target companies that are more likely to have committed financial statement fraud. Another implication of the results, specifically the attribute selection results, is that researchers developing new fraud predictors need to examine the utility of the fraud predictors using more than one classification algorithm. In addition to using logistic regression, other classification algorithms like SVM and bagging should be used when examining the utility of fraud predictors.

A natural extension of this research is to examine additional classification algorithms. While classification algorithms were selected based on findings in prior research, it is possible that other classification algorithms will performance well in financial statement fraud detection. Future research can also leverage data mining research that focuses on the class imbalance problem, which has proposed a number of sampling techniques, such as SMOTE, to improve classification performance (Chawla et al. 2002). The utility of these techniques in detecting fraud needs to be evaluated. Future research can also follow Cecchini et al. (2010) and develop artifacts that are designed specifically for the fraud domain. Such artifacts could, for example, be designed to address the distinguishing characteristics of the fraud domain.

## APPENDIX

### Classification Algorithm Descriptions

The C4.5 decision tree algorithm examines the information gain provided by each attribute and splits the data using the attribute that provides the highest information gain. The created branches are then split further by again examining the information gain. The minimum number of instances permissible at a leaf can be manipulated. To avoid overfitting, the branches are pruned based on estimates of classification errors established using a confidence value, which can also be manipulated (Quinlan 1993).

Logistic regression is a statistical algorithm that estimates the probability of an event occurring by applying maximum likelihood estimation after transforming the dependent variable into the natural log of the odds of the event. ANN is a nonlinear machine learning algorithm designed based on biological neural networks with interconnected input, hidden, and output nodes (Green and Choi 1997). Both logistic regression and ANN have been used extensively in prior accounting and fraud research.

SVM algorithms classify data points by learning hyperplanes (linear models) that provide the best separation of positive instances from negative instances. In a classification context with n object features, a hyperplane is a linear model with n−1 dimensions intersecting a space of n dimensions into two parts. The objective is to find the hyperplane that maximizes the separation of the data points of the two different classes. The hyperplane that provides the best separation is found by solving a large quadratic programming optimization problem. To improve training speed, sequential minimal optimization (SMO) solves the quadratic programming problem by breaking it up into a series of smaller problems that are then solved analytically (Platt 1999).

Stacking is an ensemble-based method that combines the output of heterogeneous base-classifiers, i.e., different types of classifiers, trained on the same data. The base-classifier output is combined using a meta-classifier. The meta-classifier can be any classification algorithm, but is typically a relatively simple linear model or decision tree (Wolpert 1992).

Bagging is an ensemble-based method that combines the output of classification algorithms that are of the same type but trained using different data. The training data for the base-classifiers are generated by sampling with replacement from the original training data. The base-classifiers' probability estimates are then combined by taking the average of the individual estimates (Breiman 1996).

## Preprocessing

The preprocessing procedures contain four steps designed to empirically determine good (1) *training* prior fraud probabilities, (2) attribute transformation methods, (3) attributes, and (4) algorithm tuning configurations to use to train the classifiers. The classifiers were not tuned in steps 1, 2, and 3, and were instead implemented using their default settings. These four preprocessing steps are described next.

### *Training Prior Fraud Probability*

In order to determine prior fraud probabilities for *training* the classifiers, the performances of the classifiers were compared after training the classifiers using ten different training prior fraud probability levels: 0.3, 0.6, 1, 1.5, 2.5, 5, 10, 20, 40, and 60 percent. Figure 3 shows an overview of the experimental procedures performed in this evaluation.

The incoming data (the raw dataset with 51 fraud firms and 15,934 nonfraud firms) was first split into training and test sets using ten-fold cross-validation. In each of the ten cross-validation rounds, the training set was first manipulated by undersampling the majority class. Next, the undersampled training set was used to train the six classifiers. The cross-validation test set (not undersampled) was then given to the trained classifiers to classify. The training data undersampling, classifier training, and classifier evaluation steps were repeated within each of the ten cross-validation rounds ten times (once for each training prior fraud probability level, i.e., 0.3, 0.6, 1, 1.5, 2.5, 5, 10, 20, 40, and 60 percent). The folds for each unique classifier and training prior fraud probability combination were then combined to generate a total of 60 datasets, where each dataset contained fraud probability estimates for the 51 fraud firms and 15,934 nonfraud firms. For each of the 60 datasets, classification thresholds that minimized ERC were determined for each *evaluation* prior fraud probability and *evaluation* relative error cost treatment combination (three and 11 levels respectively), for a total of 1,980 ERC measures (33 ERC measures for each of the 60 datasets). Finally, the best *training* prior fraud probability was selected for each classifier, *evaluation* prior fraud probability, and *evaluation* relative error cost combination. The results in Table 8 shows, as expected, that the selected training prior fraud probability increases as evaluation prior fraud probability increases and evaluation relative error costs decreases.

### *Attribute Transformation*

The preprocessing was continued by evaluating three data transformation methods. These methods normalized, discretized, and standardized the data. The utility provided by each of the three methods and no transformation was compared for each classifier at the *training* prior fraud probabilities that minimized ERC at a cost ratio of 1:50 and a prior fraud probability of 0.6 percent, i.e., the median treatment level of the two evaluation factors.

The feature transformation preprocessing procedures were very similar to the procedures used in training prior fraud probability preprocessing (presented in Figure 3). The feature transformation preprocessing, however, generated four training and test data copies in each of the ten cross-

**FIGURE 3**
**Overview of *Training* Prior Fraud Probability Preprocessing**

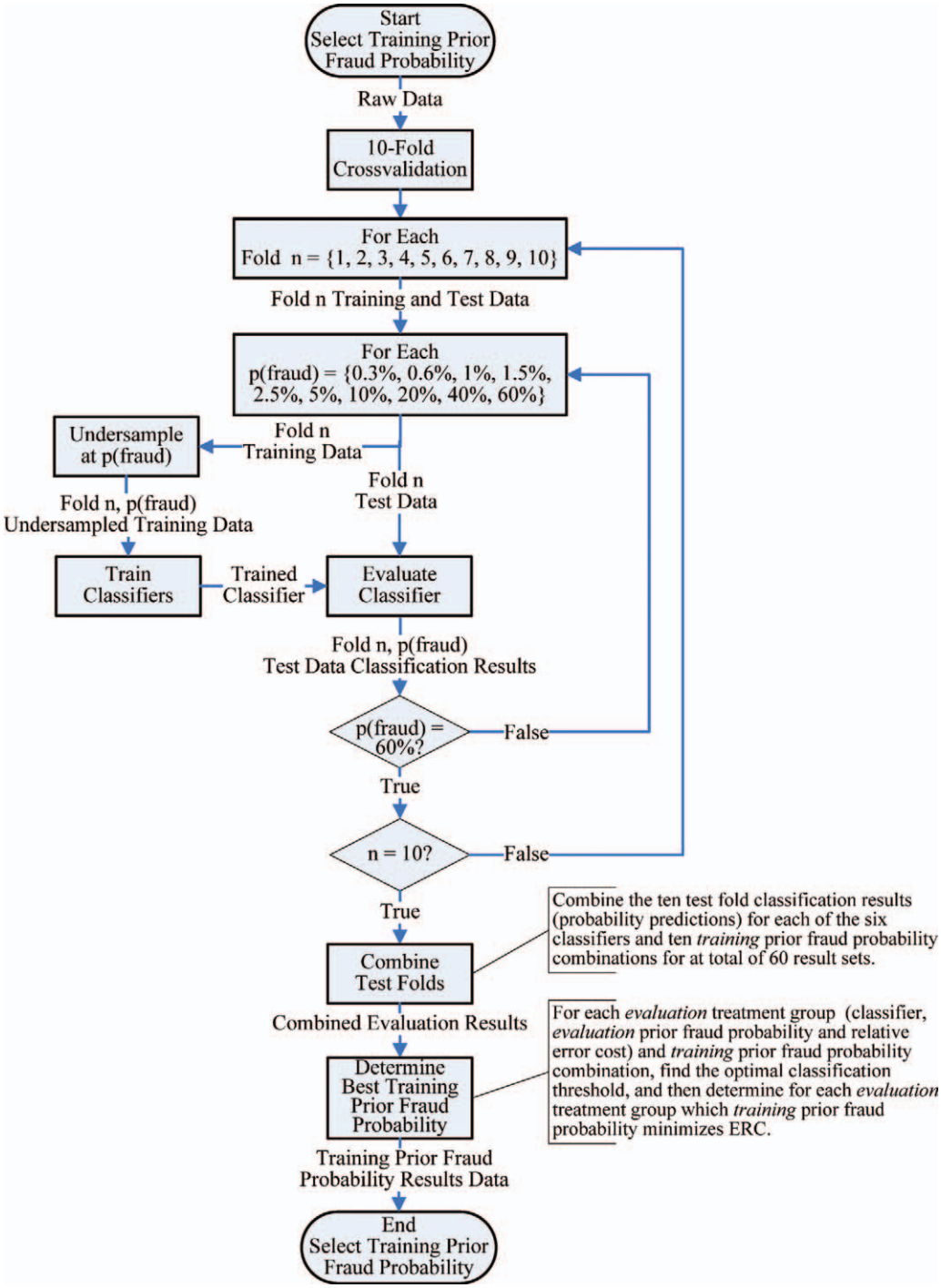


**TABLE 8**

**Preferred *Training* Prior Fraud Probabilities for each Classifier at Different *Evaluation* Prior Fraud Probability and Relative Error Cost Levels[a]**

| ANN | | | SVM | | | C4.5 | | |
|---|---|---|---|---|---|---|---|---|
| Evaluation | | Training | Evaluation | | Training | Evaluation | | Training |
| Fraud Prob. | Relative Error Cost | Fraud Prob. | Fraud Prob. | Relative Error Cost | Fraud Prob. | Fraud Prob. | Relative Error Cost | Fraud Prob. |
| 0.3% | 1:1–1:50 | 0.6% | 0.3% | 1:1–1:90 | 20% | 0.3% | 1:1–1:100 | 5% |
| 0.3% | 1:60–1:100 | 60% | 0.3% | 1:100 | 60% | 0.6% | 1:1–1:50 | 5% |
| 0.6% | 1:1–1:20 | 0.6% | 0.6% | 1:1–1:40 | 20% | 0.6% | 1:60–1:90 | 10% |
| 0.6% | 1:30–1:100 | 60% | 0.6% | 1:50–1:100 | 60% | 0.6% | 1:100 | 40% |
| 1.2% | 1:1–1:10 | 0.6% | 1.2% | 1:1–1:20 | 20% | 1.2% | 1:1–1:40 | 10% |
| 1.2% | 1:20–1:100 | 60% | 1.2% | 1:30–1:100 | 60% | 1.2% | 1:50–1:80 | 40% |
| | | | | | | 1.2% | 1:90–1:100 | 60% |
| Logistic | | | Bagging | | | Stacking | | |
| 0.3% | 1:1–1:100 | 1.5% | 0.3% | 1:1–1:100 | 60% | 0.3% | 1:1–1:100 | 60% |
| 0.6% | 1:1–1:100 | 1.5% | 0.6% | 1:1–1:100 | 60% | 0.6% | 1:1–1:100 | 60% |
| 1.2% | 1:1–1:50 | 1.5% | 1.2% | 1:1–1:100 | 60% | 1.2% | 1:1–1:100 | 60% |
| 1.2% | 1:60–1:80 | 10% | | | | | | |
| 1.2% | 1:90–1:100 | 20% | | | | | | |

[a] For each *evaluation* treatment group (two columns to the left for each classifier) the classifiers were evaluated using different *training* prior fraud probabilities. The *training* prior fraud probability that generated the lowest ERC for each classifier in each *evaluation* treatment group was then selected (the rightmost column for each classifier).

validation rounds. The features in the respective training and test set pairs were then normalized, standardized, discretized, or not transformed. The classifiers were subsequently trained using the training data with transformed features and then made fraud probability estimates for test data cases that also contained transformed features. In the end, the best feature transformation method (or no method) was selected for each classifier, *evaluation* prior fraud probability, and *evaluation* relative error cost combination.

The results reported in Table 9 show that classifiers trained with data that were normalized generally produced the lowest ERC, and that no filter was preferred by some of the classifiers at lower *evaluation* prior fraud probability and higher *evaluation* relative error cost combinations.

### *Fraud Attribute Utility*

One of the research objectives was to examine which fraud attributes provide utility to the different classifiers. Answering this question can facilitate more efficient data collection as predictors that provide little or no utility to the classifiers do not have to be collected. Furthermore, this knowledge can provide the foundation for reducing dataset dimensionality (reducing the number of attributes), which can improve the performance of the classifiers. To evaluate the utility of the attributes, a wrapper attribute selection technique was used that has been shown in prior research to be effective (Hall and Holmes 2003). For each classification algorithm, the wrapper compares the accuracy of trained classifiers after being trained using datasets with different feature combinations. However, as discussed earlier, accuracy is not a good measure of performance in the fraud domain unless the test data is altered to take into account the actual prior fraud probability and relative error costs in the domain. Therefore, assuming an average prior fraud probability of 0.006

**TABLE 9**

**Preferred Feature Transformation Methods at Different *Evaluation* Prior Fraud Probability and Relative Error Cost Levels[a]**

| ANN | | | SVM | | | Logistics | | |
|---|---|---|---|---|---|---|---|---|
| Evaluation | | | Evaluation | | | Evaluation | | |
| Fraud Prob. | Relative Error Cost | Transformation Method | Fraud Prob. | Relative Error Cost | Transformation Method | Fraud Prob. | Relative Error Cost | Transformation Method |
| 0.3% | 1:1–1:100 | Normalize | 0.3% | 1:1–1:100 | Normalize | 0.3% | 1:1–1:100 | Normalize |
| 0.6% | 1:1–1:100 | Normalize | 0.6% | 1:1–1:100 | Normalize | 0.6% | 1:1–1:100 | Normalize |
| 1.2% | 1:1–1:100 | Normalize | 1.2% | 1:1–1:100 | Normalize | 1.2% | 1:1–1:100 | Normalize |
| C4.5 | | | Bagging | | | Stacking | | |
| 0.3% | 1:1–1:100 | No Filter | 0.3% | 1:1–1:100 | No Filter | 0.3% | 1:1–1:80 | No Filter |
| 0.6% | 1:1–1:100 | No Filter | 0.6% | 1:1–1:60 | No Filter | 0.3% | 1:90–1:100 | Normalize |
| 1.2% | 1:1–1:80 | No Filter | 0.6% | 1:70–1:100 | Normalize | 0.6% | 1:1–1:40 | No Filter |
| 1.2% | 1:90–1:100 | Standardize | 1.2% | 1:1–1:30 | No Filter | 0.6% | 1:50–1:100 | Normalize |
| | | | 1.2% | 1:40–1:100 | Normalize | 1.2% | 1:1–1:20 | No Filter |
| | | | | | | 1.2% | 1:30–1:100 | Normalize |

[a] For each *evaluation* treatment group (two columns to the left), the classifiers were evaluated using different data transformation methods. The data transformation method that generated the lowest ERC for each classifier in each *evaluation* treatment group was then selected.

(Bell and Carcello 2000) and an average relative error cost of 1:30 (Bayley and Taylor 2007), a dataset with 51 fraud firms and 282 nonfraud firms ($(1 - 0.006) * 51/(0.006 * 30)$) was used for both training and evaluating the classifiers. Within the wrapper, a genetic search algorithm was used to search for optimal attribute sets.

The feature selection preprocessing procedures were also very similar to the procedures shown in Figure 3. However, during feature selection, the features in both the training set and the test set were normalized for logistic regression, SVM, ANN, and stacking, while no feature transformation was used for bagging and C4.5. In each cross-validation fold, the wrapper then selected attributes for each classifier. After all ten cross-validation folds had been processed, the features selected most frequently by each classifier over the ten cross-validation folds were then selected for each classifier. Table 4 shows a summary of the variables selected for the different classifiers.

### *Classifier Tuning*

The classification algorithms were tuned following prior research. For C4.5, three confidence values (15, 20, and 25 percent) and three minimum number of instances at a leaf (two, three, and five) were examined, for a total of nine C4.5 configurations (Witten and Frank 2005). Logistic regression was not tuned, meaning that logistic regression was used with parameters set to their default values. Following Feroz et al. (2000) and Green and Choi (1997), the ANN was tuned by first determining a good learning time without manipulating the other settings. Subsequently, the learning rate and momentum were both manipulated at three levels: 0.1, 0.3, and 0.5. The number of hidden nodes was manipulated at four levels: 4, 8, 12, and 16. Thus, after the learning time was determined, a total of 27 ANN configurations were included in the experiment.

Following Shin et al. (2005), SVM was tuned by manipulating the complexity parameter C at five levels: 1, 10, 50, 75, and 100. Shin et al. (2005) also manipulated a radial basis kernel parameter, but since Weka uses a polynomial kernel function, the exponent of the polynomial kernel was instead manipulated at five levels: 0.5, 1, 2, 5, and 10. Thus, 25 SVM configurations were included in the experiment. Furthermore, buildLogisticModels was set to true. This setting creates probability estimates by fitting logistic regression models to the SVM outputs.

Stacking was configured using the default Weka setting for the number of cross-validation folds (set at 10). Following Kotsiantis et al. (2006), all the other experimental classifiers were used as base-classifiers. For these classifiers, all classifier configurations that provided the best performance for a given classification algorithm at one or more experimental treatment levels were used as base-classifiers. Based on recommendations to use a relatively simple meta-classifier (Wolpert 1992) and experiments performed by Chan et al. (1999) and Prodromidis et al. (2000), NaiveBayes was used as the meta-classifier. In Weka, NaiveBayes can be configured to use either kernel estimation or a single normal distribution for modeling numeric attributes. There is also an option to use supervised discretization to process numeric attributes. These parameter settings were manipulated for a total of four stacking configurations. The bagging implementation was based on Breiman (1996) and used decision trees as the base-classifiers, more specifically C4.5, and set the number of sampling iterations to 50. Finally, the size of each bag (75, 100, or 125 percent) and whether to calculate out-of-bag error (yes or no) were manipulated for a total of six bagging configurations.

The 72 C4.5, SVM, ANN, logistic regression, stacking, and bagging configurations described above were evaluated using ten-fold stratified cross-validation. The ten test folds generated from the ten-fold cross-validation were then combined for each classifier tuning configuration, for a total of 72 datasets. Next, optimal thresholds were found for each of the 72 datasets and used to calculate the best ERC for each *evaluation* prior fraud probability (three levels) and *evaluation* relative error cost (11 levels) combination. This generated at total of 2,376 ERC scores ($3 * 11 * 72$), from which the best classifier tuning configuration was selected for each classifier, *evaluation* prior fraud probability, and *evaluation* relative error cost combination.

## REFERENCES

American Institute of Certified Public Accountants (AICPA). 1988. *The Auditor's Responsibility to Detect and Report Errors and Irregularities*. Statement on Auditing Standards (SAS) No. 53. New York, NY: AICPA.

American Institute of Certified Public Accountants (AICPA). 1997. *Consideration of Fraud in a Financial Statement Audit*. Statement on Auditing Standards (SAS) No. 82. New York, NY: AICPA.

Association of Certified Fraud Examiners (ACFE). 2008. *Report to the Nation on Occupational Fraud and Abuse*. Austin, TX: ACFE.

Bayley, L., and S. Taylor. 2007. Identifying earnings management: A financial statement analysis (red flag) approach. Working paper ABN AMRO and University of New South Wales.

Beasley, M. 1996. An empirical analysis of the relation between the board of director composition and financial statement fraud. *The Accounting Review* 71 (4): 443–465.

Bell, T., and J. Carcello. 2000. A decision aid for assessing the likelihood of fraudulent financial reporting. *Auditing: A Journal of Practice & Theory* 19 (1): 169–184.

Beneish, M. 1997. Detecting GAAP violation: Implications for assessing earnings management among firms with extreme financial performance. *Journal of Accounting and Public Policy* 16: 271–309.

Beneish, M. 1999. Incentives and penalties related to earnings overstatements that violate GAAP. *The Accounting Review* 74 (4): 425–457.

Breiman, L. 1996. Bagging predictors. *Machine Learning* 24 (2): 123–140.

Breiman, L., J. Friedman, R. Olshen, and C. Stone. 1984. *Classification and Regression Trees*. Boca Raton, FL: Chapman and Hall/CRC Press.

Cecchini, M., H. Aytug, G. Koehler, and P. Pathak. 2010. Detecting management fraud in public companies. *Management Science* 56 (7): 1146–1160.

Chan, P. K., W. Fan, A. L. Prodromidis, and S. J. Stolfo. 1999. Distributed data mining in credit card fraud detection. *IEEE Intelligent Systems and Their Applications* 14 (6): 67–74.

Chawla, N. V. 2005. Data mining for imbalanced datasets: An overview. In *The Data Mining and Knowledge Discovery Handbook*, edited by Maimon, O., and L. Rokach, 853–867. Secaucus, NJ: Springer-Verlag New York, Inc.

Chawla, N. V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research* 16: 321–357.

Chen, C., and J. Sennetti. 2005. Fraudulent financial reporting characteristics of the computer industry under a strategic-systems lens. *Journal of Forensic Accounting* 6 (1): 23–54.

Dechow, P., R. Sloan, and A. Sweeney. 1996. Causes and consequences of earnings manipulations: An analysis of firms subject to enforcement actions by the SEC. *Contemporary Accounting Research* 13 (1): 1–36.

Dopuch, N., R. Holthausen, and R. Leftwich. 1987. Predicting audit qualifications with financial and market variables. *The Accounting Review* 62 (3): 431–454.

Drummond, C., and R. C. Holte. 2003. C4.5, class imbalance, and cost sensitivity: Why undersampling beats over-sampling. In *The Proceedings of the Workshop on Learning from Imbalanced Datasets II*, International Conference on Machine Learning, Washington, D.C.

Fan, A., and M. Palaniswami. 2000. Selecting bankruptcy predictors using a support vector machine approach. *Neural Networks* 6: 354–359.

Fanning, K., and K. Cogger. 1998. Neural network detection of management fraud using published financial data. *International Journal of Intelligent Systems in Accounting, Finance and Management* 7 (1): 21–41.

Feroz, E., T. Kwon, V. Pastena, and K. Park. 2000. The efficacy of red flags in predicting the SEC's targets: An artificial neural networks approach. *International Journal of Intelligent Systems in Accounting, Finance and Management* 9 (3): 145–157.

Fries, T., N. Cristianini, and C. Campbell. 1998. The Kernel-Adatron algorithm: A fast and simple learning procedure for support vector machines. In *The Proceedings of the 15th International Conference on Machine Learning*, Madison, WI.

Green, B. P., and J. H. Choi. 1997. Assessing the risk of management fraud through neural network technology. *Auditing: A Journal of Practice & Theory* 16 (1): 14–28.

Hall, M., and G. Holmes. 2003. Benchmarking attribute selection techniques for discrete class data mining. *IEEE Transactions on Knowledge and Data Engineering* 15 (3): 1–16.

Kaminski, K., S. Wetzel, and L. Guan. 2004. Can financial ratios detect fraudulent financial reporting? *Managerial Auditing Journal* 19 (1): 15–28.

Kirkos, E., C. Spathis, and Y. Manolopoulos. 2007. Data mining techniques for the detection of fraudulent financial statements. *Expert Systems with Applications* 32 (4): 995–1003.

Kotsiantis, S., E. Koumanakos, D. Tzelepis, and V. Tampakas. 2006. Forecasting fraudulent financial statements using data mining. *International Journal of Computational Intelligence* 3 (2): 104–110.

Lee, T. A., R. W. Ingram, and T. P. Howard. 1999. The difference between earnings and operating cash flow as an indicator of financial reporting fraud. *Contemporary Accounting Research* 16 (4): 749–786.

Lin, J., M. Hwang, and J. Becker. 2003. A fuzzy neural network for assessing the risk of fraudulent financial reporting. *Managerial Auditing Journal* 18 (8): 657–665.

Perlich, C., F. Provost, and J. Simonoff. 2003. Tree induction vs. logistic regression: A learning-curve analysis. *Journal of Machine Learning Research* 4: 211–255.

Perols, J., and B. Lougee. 2009. Prior earnings management, forecast attainment, unexpected revenue per employee, and fraud. In *The Proceedings of the American Accounting Association Western Region Annual Meeting*, San Diego, CA.

Phua, C., D. Alahakoon, and V. Lee. 2004. Minority report in fraud detection: Classification of skewed data. *SIGKDD Explorations* 6 (1): 50–59.

Platt, J. 1999. Fast training of support vector machines using sequential minimal optimization. In *Advances in Kernel Methods: Support Vector Learning*, edited by Scholkopf, B., C. J. C. Burges, and A. J. Smola, 185–208. Cambridge, MA: MIT.

Prodromidis, A., P. Chan, and S. Stolfo. 2000. Meta-learning in distributed data mining systems: Issues and approaches. In *Advances in Distributed and Parallel Knowledge Discovery*, edited by Kargupta, H., and P. Chan, 81–114. Menlo Park, CA: AAAI/MIT.

Provost, F., and T. Fawcett. 1997. Analysis and visualization of classifier performance: comparison under imprecise class and cost distributions. In *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, Menlo Park, CA.

Provost, F., T. Fawcett, and R. Kohavi. 1998. The case against accuracy estimation for comparing induction algorithms. In *The Proceedings of the Fifteenth International Conference on Machine Learning*, Madison, WI.

Quinlan, J. R. 1993. *C4.5: Programs for Machine Learning*. San Francisco, CA: Morgan Kaufmann Publishers.

Shin, K. S., T. Lee, and H. J. Kim. 2005. An application of support vector machines in bankruptcy prediction model. *Expert Systems with Application* 28: 127–135.

Summers, S. L., and J. T. Sweeney. 1998. Fraudulently misstated financial statements and insider trading: An empirical analysis. *The Accounting Review* 73 (1): 131–146.

Uzun, H., S. H. Szewczyk, and R. Varma. 2004. Board composition and corporate fraud. *Financial Analysts Journal* 60 (3): 33–43.

Weiss, G. M. 2004. Mining with rarity: A unifying framework. *ACM SIGKDD Explorations Newsletter* 6 (1): 7–19.

West, D., S. Dellana, and J. Qian. 2005. Neural network ensemble strategies for decision applications. *Computer and Operations Research* 32 (10): 2543–2559.

Witten, I. H., and E. Frank. 2005. *Data Mining: Practical Machine Learning Tools and Techniques*. San Francisco, CA: Morgan Kaufmann Publishers.

Wolpert, D. 1992. Stacked generalization. *Neural Networks* 5 (2): 241–259.