MIS Quarterly

SPECIAL ISSUE: BUSINESS INTELLIGENCE RESEARCH

METAFRAUD: A META-LEARNING FRAMEWORK FOR DETECTING FINANCIAL FRAUD

Ahmed Abbasi
McIntire School of Commerce, University of Virginia,
Charlottesville, VA 22908 U.S.A. {abbasi@comm.virginia.edu}

Conan Albrecht, Anthony Vance, and James Hansen
Information Systems Department, Marriott School of Management, Brigham Young University,
Provo, UT 84606 U.S.A. {ca@byu.edu} {anthony@vance.name} {james_hansen@byu.edu}

Appendix A

Comparing Context-Based Classifiers Against Baseline Classifies Using Regulators' Cost Settings

This appendix reports the results for the baseline and yearly/quarterly context-based classifiers when using the 1:10 regulator cost setting. Since the AUC values are computed across different cost settings (and are therefore the same for the investor and regulator situations), we report only the legitimate/fraud recall rates. Overall AUC values as well as results for the investor cost setting (1:20) can be found in the subsection "Comparing Context-Based Classifiers Against Baseline Classifiers" of the main paper.

Table A1 shows the results for the baseline classifiers. Tables A2 and A3 show the results for the yearly and quarterly context-based classifiers (i.e., the 14 classifiers coupled with the 84 and 336 yearly/quarterly context-based features, respectively). For all three feature sets, the most balanced results were attained using Logit. In comparison with the baseline classifiers, the yearly and quarterly context-based classifiers had fraud recall rates that were over 20 percent higher on average, coupled with slightly higher average legit recall values. Similar to the investor situation, even when using regulators' cost settings, the yearly and quarterly context-based classifiers' results were quite diverse with respect to their legit and fraud recall rates.

Table A1. Baseline Results

	Legit		Fraud			Legit		Fraud	
Classifier	Prec.	Rec.	Prec.	Rec.	Classifier	Prec.	Rec.	Prec.	Rec.
SVM-Lin	91.0	60.1	6.4	31.5	ADTree	93.8	58.7	10.3	55.0
LogitReg	95.9	71.0	16.1	64.6	RandForest	95.1	72.7	15.3	57.0
J48	94.9	71.0	14.3	56.2	NBTree	95.0	89.8	27.7	45.2
BayesNet	95.3	77.6	17.7	55.8	REPTree	93.6	70.4	11.4	44.3
NaiveBayes	92.1	53.3	8.0	47.2	JRip	93.3	84.7	14.5	30.1
SVM-RBF	89.4	65.5	2.6	10.5	NNge	92.2	78.1	8.6	23.7
SVM-Poly	91.1	56.0	6.7	36.4	NeuralNet	92.1	60.0	8.0	40.1

Table A2. Yearly Context-Based Classifiers

	Legit		Fraud			Legit		Fraud	
Classifier	Prec.	Rec.	Prec.	Rec.	Classifier	Prec.	Rec.	Prec.	Rec.
SVM-Lin	98.7	50.0	13.7	92.2	ADTree	97.6	65.5	16.9	81.2
LogitReg	96.7	83.5	26.0	67.0	RandForest	96.2	73.4	17.7	66.3
J48	93.6	84.1	15.2	33.0	NBTree	95.9	82.4	22.7	59.7
BayesNet	96.0	75.5	18.3	63.3	REPTree	95.5	63.2	13.3	65.5
NaiveBayes	97.6	61.0	15.5	82.9	JRip	94.6	70.7	13.5	53.1
SVM-RBF	94.1	64.9	11.4	52.6	NNge	93.9	74.7	13.1	44.0
SVM-Poly	97.9	60.0	15.5	85.1	NeuralNet	94.3	68.0	12.3	52.1

Table A3. Quarterly Context-Based Classifiers

	Legit		Fraud			Legit		Fraud	
Classifier	Prec.	Rec.	Prec.	Rec.	Classifier	Prec.	Rec.	Prec.	Rec.
SVM-Lin	98.1	65.6	17.6	85.1	ADTree	97.7	55.4	14.1	85.1
LogitReg	96.0	71.5	16.5	65.3	RandForest	93.6	95.5	32.0	24.7
J48	95.3	92.9	36.2	46.7	NBTree	93.6	93.9	26.6	25.7
BayesNet	95.2	69.5	14.4	59.7	REPTree	96.7	57.0	13.5	77.5
NaiveBayes	96.7	52.3	12.6	79.5	JRip	95.0	64.1	12.8	61.1
SVM-RBF	97.7	62.2	16.0	83.4	NNge	93.0	82.7	12.3	28.1
SVM-Poly	96.2	76.3	19.2	65.3	NeuralNet	94.1	67.0	11.9	51.6

Appendix B

Evaluating Stacked Classifiers Using Regulators' Cost Settings

Rhis appendix reports the results for the yearly, quarterly, and combined stack classifiers when using the 1:10 regulator cost setting. Tables B1 and B2 show the legitimate and fraud recall results for the yearly and quarterly stack classifiers. Not surprisingly, the increased cost of false positives (as compared to false negatives) for the regulator setting resulted in higher legitimate recall rates relative to the investor setting (see the subsection "Evaluating Stacked Classifiers" for the investor setting results). For instance, the annual stack using a top-level Logit classifier improved its legitimate recall rate by over 20 percent when using the regulator cost setting. This is important since it would result in 20 percent fewer unnecessary audits (over 900 fewer based on our test bed), albeit with 13 percent fewer frauds detected. On average, the quarterly context-based classifiers had 15 percent higher fraud recall and 7 percent lower legitimate recall rates.

Table B1. Yearly Stack Classifiers

	Legit		Fraud			Legit		Fraud	
Classifier	Prec.	Rec.	Prec.	Rec.	Classifier	Prec.	Rec.	Prec.	Rec.
SVM-Lin	98.3	69.8	19.8	86.1	ADTree	74.2	33.3	21.1	79.7
LogitReg	97.1	93.5	47.4	68.2	RandForest	82.5	35.2	24.2	64.5
J48	96.9	82.5	25.4	68.9	NBTree	82.7	36.2	24.9	66.5
BayesNet	96.8	85.7	28.9	67.5	REPTree	73.0	30.7	19.3	74.8
NaiveBayes	96.9	82.0	25.1	69.7	JRip	81.5	36.8	24.8	70.9
SVM-RBF	97.2	84.4	28.4	71.6	NNge	79.1	33.4	22.1	68.5
SVM-Poly	96.8	77.5	21.2	70.2	NeuralNet	75.0	32.7	20.8	76.0

Table B2. Quarterly Stack Classifiers

	Legit		Fraud			Legit		Fraud	
Classifier	Prec.	Rec.	Prec.	Rec.	Classifier	Prec.	Rec.	Prec.	Rec.
SVM-Lin	98.2	68.5	19.0	85.8	ADTree	99.6	67.7	20.6	96.8
LogitReg	97.4	86.9	32.8	73.6	RandForest	97.7	77.2	23.1	79.2
J48	98.3	70.7	20.2	85.8	NBTree	99.0	71.7	21.8	91.4
BayesNet	98.4	82.3	29.1	84.1	REPTree	98.8	68.0	19.6	90.5
NaiveBayes	97.4	68.5	17.8	79.0	JRip	99.0	67.9	19.9	92.4
SVM-RBF	99.2	84.5	33.9	92.2	NNge	97.5	74.2	20.7	78.0
SVM-Poly	98.7	69.5	20.3	89.7	NeuralNet	97.9	77.0	23.4	81.2

Table B3 shows the results for the combined stack classifiers, which used the yearly and quarterly context-based classifiers' classifications as input. As with the investor cost setting, the combined stacks using the regulator cost setting leveraged the enhanced legitimate recall rates from the yearly stacks and the improved fraud recall associated with the quarter stacks for enhanced, and generally more balanced performance. For instance, the combined stack using a Logit top-level classifier improved fraud recall by 4 percent over the annual stack, while providing somewhat comparable legitimate recall.

Table B3. Combined Stack Classifiers

	Legit		Fraud			Legit		Fraud	
Classifier	Prec.	Rec.	Prec.	Rec.	Classifier	Prec.	Rec.	Prec.	Rec.
SVM-Lin	97.8	80.3	25.7	79.0	ADTree	99.1	82.0	30.5	91.4
LogitReg	97.5	93.1	47.4	72.2	RandForest	96.4	96.1	56.4	57.9
J48	97.2	85.0	29.1	71.5	NBTree	98.8	79.9	27.6	88.5
BayesNet	97.7	86.1	32.1	76.0	REPTree	98.6	79.9	27.2	87.3
NaiveBayes	97.3	91.4	41.7	70.9	JRip	98.7	82.3	29.8	87.0
SVM-RBF	98.4	85.4	33.2	83.9	NNge	95.3	94.4	41.2	45.7
SVM-Poly	97.2	93.0	46.1	69.2	NeuralNet	98.6	82.0	29.4	86.6

Appendix C

Evaluating Adaptive Semi-supervised Learning using Regulators' Cost Settings ■

This appendix reports the results for the adaptive semi-supervised learning (ASL) classifiers when using the 1:10 regulator cost setting. Table C1 shows the legitimate and fraud recall rates. On average, using ASL improved legitimate and fraud recall by 4.7 percent and 2.2 percent, respectively. Many of the classifiers attained legitimate recall rates over 90 percent. For instance, running ASL with the Logit classifier resulted in the correct classification of approximately 95 percent of the legitimate firm instances. These settings could provide a more practicable decision aid for regulators since the number of false positives would be more manageable (5 percent, or 237 of our legitimate test instances), while still identifying a fair amount of fraud (77 percent, or 315 of our fraud test instances).

Table C1. Adaptive Semi-Supervised Learning									
	Legit		Fraud			Legit		Fraud	
Classifier	Prec.	Rec.	Prec.	Rec.	Classifier	Prec.	Rec.	Prec.	Rec.
SVM-Lin	98.2	88.1	36.9	80.9	ADTree	99.2	89.3	42.5	91.4
LogitReg	98.0	95.0	57.1	77.0	RandForest	97.1	96.5	62.2	66.7
J48	97.5	92.2	44.5	72.6	NBTree	98.7	89.5	41.6	86.8
BayesNet	98.1	92.0	46.2	79.2	REPTree	98.4	88.0	37.5	83.6
NaiveBayes	97.5	92.4	45.4	72.9	JRip	98.6	88.0	38.1	85.6
SVM-RBF	98.7	87.9	38.3	87.0	NNge	96.1	98.5	75.7	54.0
SVM-Poly	97.6	92.8	46.9	73.3	NeuralNet	98.7	86.1	35.0	87.0

Appendix D

Analysis of Error Costs and Savings for the Regulators' Cost Setting

We conducted in-depth analysis of the financial impact of various components of MetaFraud relative to the baseline and comparison classifiers for the regulator cost setting (1:10). Applying a cost of $443,000 to each false positive and $4,100,000 to each false negative, we computed the error cost per firm-year for each technique evaluated in the first three subsections of the "Evaluation" section of the main paper (H1–H4), including the baseline, context-based classifiers, stack classifiers, and ASL. Table D1 shows the mean error cost results (in thousands of dollars), averaged across the 14 classifiers, for each of the aforementioned approaches. The first three columns depict the mean cost of false positives, mean cost of false negatives, and mean error cost per firm-year. In order to provide context, we also computed two naïve baselines: the error costs of auditing every firm (or none) in our test bed. For the former, this cost would simply be $443,000 multiplied by the 4,735 legit instances, while for the latter, it would be $4,100,000 multiplied by the 409 fraud cases in the test set. In both cases, these total costs were divided by the total number of test firm-years to give the costs per case. These values are presented in the first two rows of Table D1 while the last two columns provide an indication of how cost effective various approaches were relative to these two basic situations.

Based on the table, it is evident that each subsequent component of MetaFraud further improved error costs (and savings) over the baseline classifiers as well as the two basic situations. Interestingly, the baseline classifiers only provided a savings of $120.9M over not auditing any firms, underscoring the potentially grave financial ramifications of using highly inaccurate fraud detection methods as decision aids. In contrast, using ASL provided savings of $1.1B over not auditing at all.

Table D1. Mean Error Costs and Savings for Various Approaches					
Approach	Cost of False Positives	Cost of False Negatives	Total Error Cost Per Instance	Savings vs. Audit All	Savings vs. Audit None
Audit All	$407.8	$0.0	$407.8	$0.0	($81.8)
Audit None	$0.0	$326.0	$326.0	$81.8	$0.0
Baseline Classifiers	$122.7	$186.2	$308.9	$98.9	$17.1
Yearly Context-based	$122.6	$113.9	$236.5	$171.2	$89.4
Quarterly Context-based	$113.3	$128.6	$241.9	$165.9	$84.1
Yearly Stacks	$78.9	$93.5	$172.4	$235.4	$153.6
Quarterly Stacks	$107.4	$45.5	$152.9	$254.9	$173.1
Combined Stacks	$53.7	$80.1	$133.8	$274.0	$192.2
ASL	$34.5	$71.4	$102.1	$305.7	$223.9

Table D2 displays the mean error cost results (in thousands of dollars) for MetaFraud (MF) and the three comparison methods. MF's improved total error cost was attributable to enhanced false positive and false negative costs over all seven comparison settings. From an error cost perspective, MF's enhanced performance over Cecchini et al. (2010) was largely attributable to augmented costs for false positives.

Table D2. Mean Error Costs and Savings for MetaFraud and Comparison Methods

Approach	Cost of False Positives	Cost of False Negatives	Total Error Cost Per Instance	Savings vs. Audit All	Savings vs. Audit None
MetaFraud	$40.0	$60.6	$100.5	$307.2	$225.5
Cecchini et al. 2010	$82.2	$67.0	$149.2	$258.6	$176.8
Kirkos et al. 2007 - BayesNet	$112.5	$118.8	$231.2	$176.5	$94.8
Kirkos et al. 2007 – ID3	$58.3	$123.5	$181.8	$225.9	$144.1
Kirkos et al. 2007 - NeuralNet	$129.4	$106.8	$236.2	$171.6	$89.8
Gaganis 2009 - NeuralNet	$122.6	$75.7	$198.4	$209.4	$127.6
Gaganis 2009 – LogitReg	$117.2	$90.9	$208.1	$199.7	$117.9
Gaganis 2009 – SVM-Lin	$117.6	$90.9	$208.4	$199.4	$117.6

Table D3 shows the results when using MetaFraud with the comparison methods' ratios, while D4 shows the results for Tri-Training using the baseline and three comparison sets of ratios. Using MetaFraud with the comparison ratios improved total error cost per firm-year by at least $32.5K to $89.1K over the comparison approaches in Table D2. Similarly, the results in Table D4 shed light on the effectiveness of MetaFraud over alternate adaptive learning methods.

Table D3. Mean Error Costs and Savings for MetaFraud Using Comparison Methods' Ratios

Approach	Cost of False Positives	Cost of False Negatives	Total Error Cost Per Instance	Savings vs. Audit All	Savings vs. Audit None
MF-Cecchini	$56.9	$59.8	$116.7	$291.1	$209.3
MF-Kirkos	$80.2	$67.0	$147.1	$260.6	$178.9
MF-Gaganis	$69.3	$74.1	$143.5	$264.3	$182.5

Table D4. Mean Error Costs and Savings for Tri-Training Using Baseline and Comparison Ratios

Approach	Cost of False Positives	Cost of False Negatives	Total Error Cost Per Instance	Savings vs. Audit All	Savings vs. Audit None
TT-Cecchini-BayesNet	$56.9	$78.1	$135.0	$272.7	$191.0
TT-Kirkos-Logit	$118.5	$74.9	$193.4	$214.4	$132.6
TT-Gaganis-J48	$96.5	$86.9	$183.4	$224.4	$142.6
TT-Baseline-BayesNet	$64.3	$81.3	$145.6	$262.1	$180.4

Appendix E

Analysis of Error Costs and Savings for the Investors' Cost Setting

This appendix presents analysis of the financial impact of various components of MetaFraud relative to the baseline and comparison classifiers for the investor cost setting (1:20). We derived the cost of false negatives as a 20 percent drop in the median stock value ($120M) across all firm instances in our data set (Beneish 1999a, 1999b; Cox and Weirich 2002). The cost of false positives was the opportunity cost of failing

to invest in a legitimate firm; this was computed as a 1 percent increase in the median stock value. Hence, for the investor setting, we used false negative costs of $24 million and false positive costs of $1.2 million. It is important to note that these values are approximations of the median total costs for such firm instances, across all investors. Costs for individual investors would vary depending upon the percentage of total stock acquired in a specific firm.

Using these two values, we computed the error cost for each technique evaluated in the first three subsections of the "Evaluation" section of the main paper (H1–H4), including the baseline, context-based classifiers, stack classifiers, and ASL. Table E1 shows the mean error cost results (in thousands of dollars), averaged across the 14 classifiers, for each of the aforementioned approaches. The first three columns depict the mean cost of false positives, mean cost of false negatives, and mean total error cost per firm-year.

From the table, it is apparent that the various components of MetaFraud improved error costs over the baseline classifiers. In general, error costs decreased with each subsequent phase of MetaFraud with the exception of the quarterly stack classifiers, which had lower false negative rates as compared to the combined stacks. This was largely attributable to the fact that five of the quarterly stack classifiers had fraud recall rates above 90 percent (see Table B2 in Appendix B for details). Given the hefty costs exacted by false negatives on investors, these classifiers' high fraud recall rates caused the quarterly stacks to have the lowest mean total false negative costs in Table E1. However, ASL still had the best total error cost, due to a relatively better balance between false positive and false negative costs. ASL provided cost savings of over $880K per instance over the baseline classifiers.

Table E1. Mean Error Costs and Savings for Various Approaches

Approach	Cost of False Positives	Cost of False Negatives	Total Error Cost Per Instance
Baseline Classifiers	$372.0	$1,002.0	$1,374.0
Yearly Context-based	$392.6	$559.5	$952.2
Quarterly Context-based	$348.7	$661.4	$1,010.1
Yearly Stacks	$279.3	$435.0	$714.3
Quarterly Stacks	$348.7	$201.3	$550.0
Combined Stacks	$198.6	$388.0	$586.5
ASL	$121.3	$371.8	$493.2

Table E2 displays the mean error cost per firm-year results (in thousands of dollars) for MetaFraud (MF) and the three comparison methods. MF had the lowest false positive and false negative costs as compared to the seven comparison setting. From an error cost perspective, MF's enhanced performance over Cecchini et al. (2010) was once again largely attributable to augmented costs for false positives (approximately $165K lower per instance).

Table E2. Mean Error Costs and Savings for MetaFraud and Comparison Methods

Approach	Cost of False Positives	Cost of False Negatives	Total Error Cost Per Instance
MetaFraud	$114.5	$326.6	$441.1
Cecchini et al. 2010	$279.9	$340.6	$620.5
Kirkos et al. 2007 - BayesNet	$383.3	$401.2	$784.5
Kirkos et al. 2007 – ID3	$293.7	$625.2	$918.9
Kirkos et al. 2007 - NeuralNet	$395.2	$466.6	$861.7
Gaganis, 2009 - NeuralNet	$381.6	$373.3	$754.9
Gaganis, 2009 – LogitReg	$349.7	$438.6	$788.3
Gaganis, 2009 – SVM-Lin	$370.0	$405.9	$775.9

Table E3 shows the results when using MetaFraud with the comparison methods' ratios, while E4 shows the results for Tri-Training using the baseline and three comparison sets of ratios. Using MetaFraud with the comparison ratios improved total error cost by at least $109K per firm-year over the comparison results presented in Table E2. Similarly, based on the results presented in Table E4, MetaFraud outperformed Tri-Training by $40K to over $162K in terms of error costs. The results in Table E4 shed light on the effectiveness of MetaFraud over alternate adaptive learning methods.

Table E3. Mean Error Costs and Savings for MetaFraud using Comparison Methods' Ratios			
Approach	**Cost of False Positives**	**Cost of False Negatives**	**Total Error Cost Per Instance**
MF-Cecchini	$168.4	$317.3	$485.7
MF-Kirkos	$320.3	$354.6	$674.9
MF-Gaganis	$209.0	$349.9	$558.9

Table E4. Mean Error Costs and Savings for Tri-Training Using Baseline and Comparison Ratios			
Approach	**Cost of False Positives**	**Cost of False Negatives**	**Total Error Cost Per Instance**
TT-Cecchini-BayesNet	$207.6	$387.2	$594.9
TT-Kirkos-Logit	$351.6	$363.9	$715.5
TT-Gaganis-J48	$367.0	$354.6	$721.5
TT-Baseline-BayesNet	$227.9	$429.2	$657.2

Appendix F

Analysis of ASL and Combined Stacks Performance Across Test Years

Prior research has noted that the average and median times needed to detect fraud are 2.8 and 2.65 years, respectively (Beneish 1999b). This time window, which refers to the time between when the fraud is committed until the time when it is first discovered and announced (generally through media reports which precede the SEC's AAERs), poses constraints on when fraud detection classifiers can be updated. Therefore, it is important to analyze the impact of the lag (i.e., window length) between training data and test instances on detection performance. In this appendix, we present results for ASL and combined stacks using different window sizes *b*, where the "dynamic" ASL and combined stacks tested instances for a given year *a* using all training data up to and including year $a - b$. We used values of 1–5 for *b*. Figure F1 shows the performance of the combined and dynamic stacks across test years. Dynamic stack 1 denotes the legitimate recall (left chart) and fraud recall (right chart) when using a one-year window (i.e., $b = 1$). In other words, this classifier used all data prior to the test year for training. Since performance for this approach was no different than that of the combined stacks in 2000, we only depict results for Dynamic Stack 1 from 2001 onward. Similarly, for all dynamic stacks, we only display results for years when their performance differed from the static combined stacks (i.e., the year 2000+*b*). Not surprisingly, smaller values for *b* attained better legitimate/fraud recall rates, suggesting that the smaller the time window between the training and testing data, the better the performance. Dynamic stack 1 outperformed the combined stack by 3 percent to 4 percent on average in terms of legitimate and fraud recall. Reducing the window size by one year typically improved legitimate and fraud recall rates by 0.5 percent to 1 percent.

Figure F2 shows the results for ASL and the dynamic ASL classifiers. As with the dynamic stacks, using smaller values for *b* improved performance for dynamic ASL as well, with Dynamic ASL 1 attaining the best legitimate recall and fraud. However, the performance gains for dynamic ASL were less pronounced than those attained by the dynamic stacks, as evidenced by the smaller variations between classifiers in Figure F2 as compared to Figure F1.

Figure F3 shows the results for ASL, Dynamic ASL 1, Combined Stacks, and Dynamic Stack 1. Since ASL/Combined stacks and Dynamic ASL/Stack 1 represent significantly contrasting window sizes, they provide reasonable upper and lower limits on the performance of static and dynamic ASL and combined stack classifiers. Comparing ASL against the Dynamic Stack 1 classifier, ASL had better legitimate recall across test years and better fraud recall on 6 out of 7 years. Since the performance for the dynamic ASL classifiers exceeded that of ASL, we can infer that ASL virtually dominated the combined stacks with respect to legitimate and fraud recall for various window sizes.

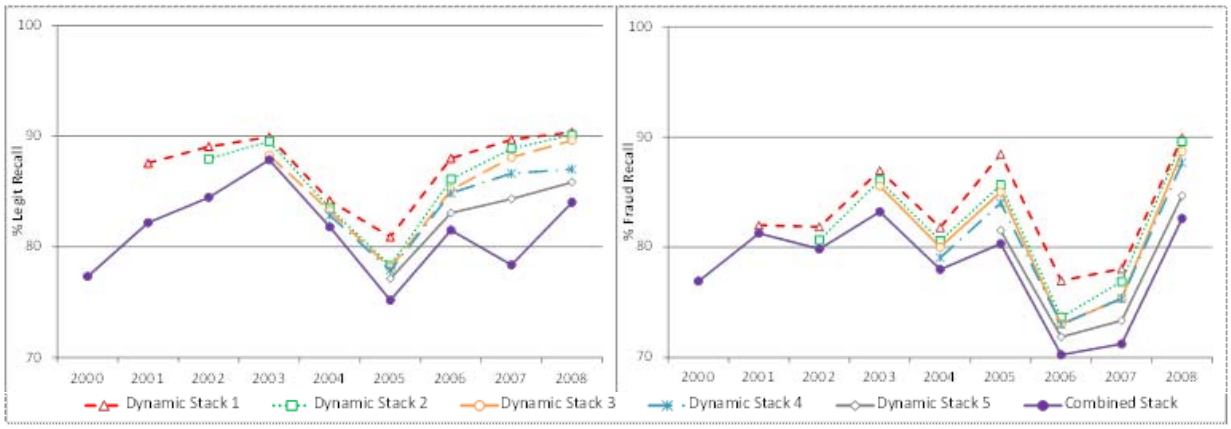

Figure F1. Performance for Combined and Dynamic Stack Across Test Years

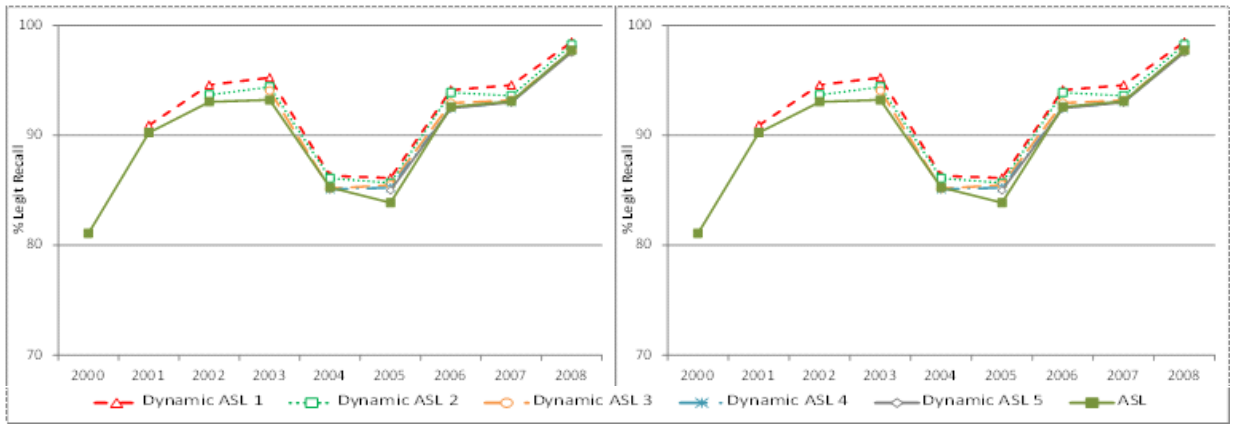

Figure F2. Performance for ASL and Dynamic Stack Across Test Years

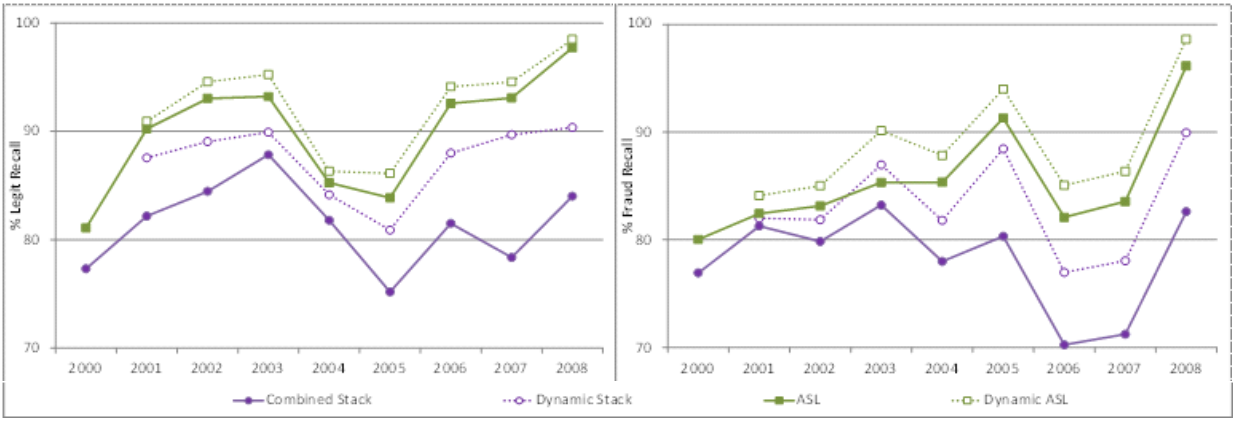

Figure F3. Performance of ASL and Combined Stack Across Test Years Using a Window Size of One Year (*b* = 1) for Dynamic Classifiers

Appendix G

Analysis of MetaFraud's Base, Stack, and ASL Classifiers

In this study, MetaFraud was run using the expanded feature space (comprised of organizational and industry context features), 28 context classifiers (14 quarterly and 14 yearly), 14 stack classifiers, and ASL. In this appendix, we evaluate the impact of model tuning on overall fraud detection performance. The most obvious areas where model tuning could be performed would be to (1) tune the input (base) context-based classifiers used by the combined stacks, in terms of quantity and composition; (2) similarly tune the stack classifiers used by ASL as input; and (3) tune the threshold used by ASL to guide inclusion of test cases in the training set during the next iteration. We focused on all three areas. For each, we attempted to retrospectively derive near-optimal performance levels using results on the test cases, in order to gain insights into potential optimal performance levels attainable using model tuning.

1. Tuning the Quantity and Composition of Context-Based Classifiers Utilized by the Stacks

For #2, we performed analysis to see which of the 28 base context-based classifiers were useful to the 14 stack classifiers' performance. Each of the 14 stack classifiers was run using different combinations of underlying context-based classifiers. Due to computational constraints, we could not evaluate every single combination. With $2^n - 1$ possible combinations, setting n to 28 would have resulted in over 268 million combinations for each stack classifier. Therefore, we set n to 14, and treated both the yearly and quarterly context-based classifiers as a pair that were always simultaneously included or excluded. For instance, assuming that each combination could be represented as a binary string, where a "1" indicated that a particular classifier was included while a "0" signified exclusion, in the combination "01111111111111," the first context-based classifier (SVM-Lin) would be excluded while the remaining 13 would be included. This would result in 26 classifiers included (13 yearly and 13 quarterly), while both the yearly and quarterly SVM-Lin classifiers would be omitted. We ran all 16,383 resulting combinations separately for each of the 14 stack classifiers. For each stack, the single setting that yielded the best performance on the test instances, across both the regulator and investor cost settings, was reported. We refer to these results as local optima since considering the yearly and quarterly classifiers as a dependent pair in the solution space causes us to exclude many viable solutions.

Tables G1 and G2 show the analysis results. Table G1 depicts the number of underlying context-based classifiers employed by each combined stack, as well as the resulting legitimate and fraud recall rates for the investor and regulator cost settings. Table G2 shows the performance gains with respect to legitimate and fraud recall when using the local optima settings, as compared to using all 28 underlying context-based classifiers. As depicted in Table G2, these local optima resulted in average gains in legitimate recall of 1 percent to 2 percent and improvements in fraud recall of 2 percent to 4 percent across the two cost settings. Based on Table G1, it is evident that the ideal quantity of underlying context classifiers varied from 8 to 20, depending on the top-level classifier employed in the stack.

Table G1. Local Optima Performance of Combined Stack Classifiers for Investor and Regulator Settings

		Investor Cost Setting (1:20)				Regulator Cost Setting (1:10)			
		Legit		Fraud		Legit		Fraud	
Classifier	No. of Classifiers	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.
SVM-Lin	14	98.4	88.4	38.3	83.6	97.7	87.8	35.0	76.0
LogitReg	18	98.2	89.1	39.1	81.2	98.1	90.5	42.2	80.2
J48	10	98.6	77.1	24.8	87.8	98.3	83.4	30.2	83.1
BayesNet	16	98.5	81.2	28.3	85.8	98.5	85.7	34.0	85.1
NaiveBayes	14	98.3	84.7	31.9	83.1	97.7	91.2	42.4	74.8
SVM-RBF	12	99.0	76.9	25.4	91.0	98.3	86.0	33.9	83.1
SVM-Poly	20	97.5	88.9	36.6	74.1	98.1	87.7	36.2	80.7
ADTree	16	98.0	84.7	31.1	80.2	98.9	87.1	37.3	88.5
RandForest	8	98.5	81.8	28.8	85.3	98.2	84.5	31.5	82.4
NBTree	8	98.3	84.0	30.9	83.1	98.4	87.7	36.9	83.1
REPTree	8	98.5	81.1	28.1	85.3	98.3	88.1	37.3	82.2
JRip	14	98.7	82.0	29.6	87.5	98.4	86.3	34.6	84.1
NNge	16	97.3	87.9	33.8	71.4	97.4	90.1	38.7	72.1
NeuralNet	8	98.7	83.0	30.7	87.0	98.7	84.0	32.1	87.5

Table G2. Performance Improvements for Local Optima Combined Stack Classifiers

	Investor Cost Setting (1:20)				Regulator Cost Setting (1:10)			
	Legit		Fraud		Legit		Fraud	
Classifier	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.
SVM-Lin	0.4	8.4	12.4	2.5	-0.1	7.6	9.3	-3.0
LogitReg	0.4	4.2	8.2	2.9	0.7	-2.6	-5.2	8.0
J48	1.4	-0.5	2.6	13.5	1.1	-1.6	1.1	11.6
BayesNet	0.5	3.4	4.0	3.7	0.9	-0.4	1.8	9.1
NaiveBayes	0.9	-6.0	-7.9	11.5	0.3	-0.2	0.7	3.9
SVM-RBF	0.1	2.5	2.1	1.0	-0.1	0.6	0.7	-0.7
SVM-Poly	0.0	0.0	0.1	0.2	0.9	-5.3	-9.9	11.5
ADTree	-1.1	8.6	6.1	-12.2	-0.2	5.2	6.8	-2.9
RandForest	1.7	-13.4	-24.2	22.5	1.9	-11.6	-25.0	24.5
NBTree	-0.5	7.0	5.9	-5.9	-0.4	7.8	9.3	-5.4
REPTree	-0.2	5.9	4.6	-2.7	-0.4	8.2	10.0	-5.1
JRip	-0.1	1.9	1.7	-1.3	-0.2	4.0	4.9	-2.9
NNge	0.7	-0.5	1.7	8.1	2.1	-4.2	-2.5	26.4
NeuralNet	-0.1	2.7	2.8	-1.0	0.1	2.0	2.8	1.0

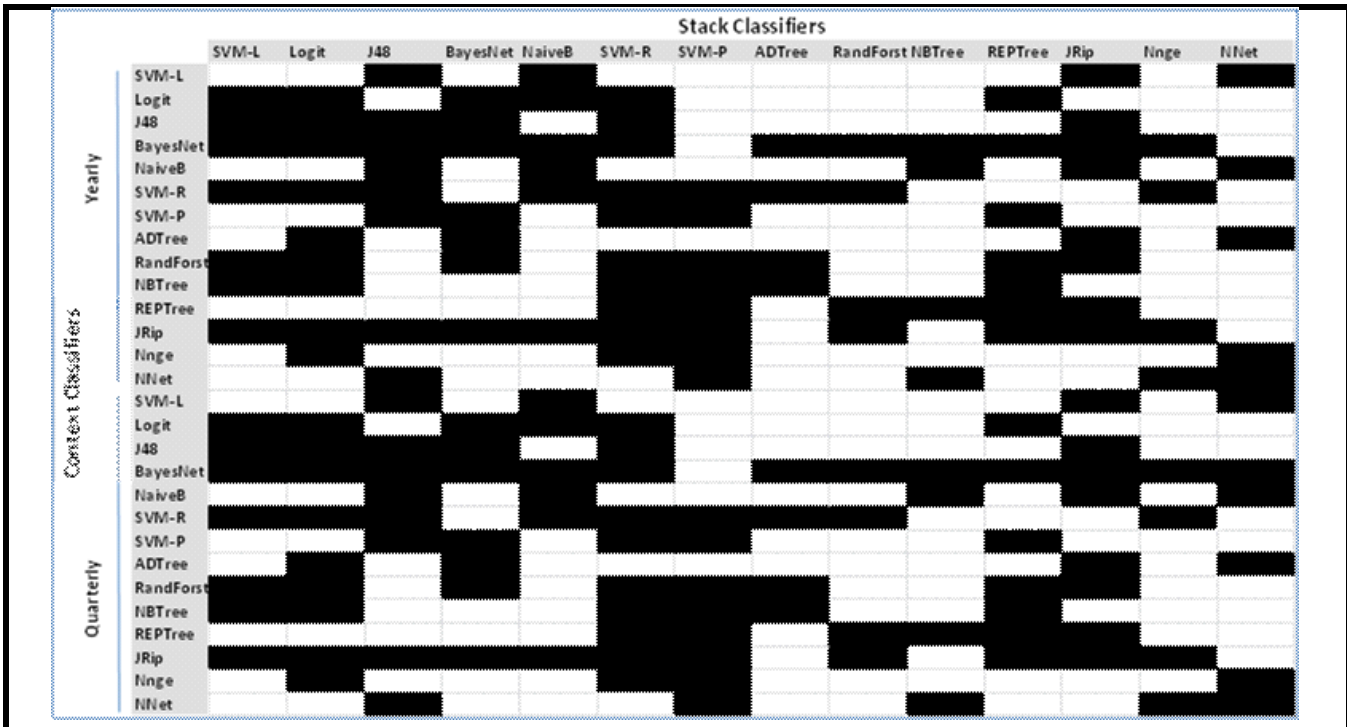

Table G2. Performance Improvements for Local Optima Combined Stack Classifiers

In order to analyze the inclusion frequency/patterns of the underlying context-based classifiers across the 14 stack classifiers, we constructed a co-occurrence matrix (depicted in Figure G1). In the matrix, colored cells indicate that a particular context classifier (row) was included in the stack classifier (column). For example, the Logit stack classifier (second column) used Logit, J48, BayesNet, SVM-RBF, ADTree, RandomForest, NBTree, and JRip, and NNge classifiers (yearly and quarterly). Based on the figure, it is evident that while certain underlying

context classifiers were utilized more than others, all underlying context-based classifiers were used by at least four stack classifiers. Classifiers such as BayesNet and JRip were each used by more than 10 stack classifiers. On average, each underlying context-based classifier was used by 6.5 stack classifiers. The results suggest that a large variety of diverse underlying context-based classifiers are essential to the success of the stack classifiers in MetaFraud.

2. Tuning the Quantity and Composition of Stack Classifiers

Having identified a set of local optima solutions for each stack classifier by evaluating different combinations of underlying (base) context-based classifiers, we turned our attention to exploring the impact of utilizing different quantities of stack classifiers on MetaFraud's overall performance (#3). We repeated the procedure taken in the previous analysis; all $2^n - 1$ stack combinations were evaluated. Each of the 14 stacks was run using the local optima settings described in Table G1 and Figure G1. All 16,383 combined stack combinations were evaluated. The setting yielding the best overall results for MetaFraud was retained. We labeled this setting "MF-BC/SC" since both the base and stack classifier arrangements had been tuned.

The results are depicted in Table G4. MetaFraud (MF) denotes the default setting where all 14 stack classifiers were incorporated, each using the 28 underlying context-based classifiers as input. MF-BC indicates the setting where all 14 local optima stack classifiers were incorporated. MF –BC/SC represents the setting where the subset of the 14 local optima stacks providing the best performance were retained (using the approach described in the previous paragraph). The performance gap between MF-BC and MetaFraud signifies the net gain attributable to tuning the base classifier arrangement, while the gap between MF-BC/SC and MF-BC indicates the marginal gain attributable to using the best arrangement of stack classifiers. The results indicate that configuring the set of context-base classifiers used with each stack, as well as the arrangement of stack classifiers employed, both improve performance. Comparing MF-BC/SC against MF, the net gains in legitimate recall were between 1.5 percent and 2 percent, while fraud recall improved by 0.5 percent to 1.5 percent. These gains resulted in error cost reductions of $13K per firm-year for the regulator setting and $25K per instance for the investor setting.

Table G4. Performance of MetaFraud With and Without Tuning

		Regulators (Cost 1:10)					Investors (Cost 1:20)				
		Legit		Fraud			Legit		Fraud		
Setting	No. of SC	Prec.	Rec.	Prec.	Rec.	Cost Per Instance	Prec.	Rec.	Prec.	Rec.	Cost Per Instance
MetaFraud (MF)	14	98.3	90.2	41.8	81.5	$100.5	98.4	89.6	40.8	82.9	$441.1
MF - BC	14	98.3	90.5	42.9	82.4	$96.1	98.4	90.5	42.9	83.1	$427.4
MF - BC/SC	10	98.4	92.1	47.6	83.1	$87.3	98.4	91.1	44.8	83.4	$415.5

3. Evaluating the Impact of Different ASL Thresholds on Fraud Detection Performance

MetaFraud was run using an ASL threshold of 14. In other words, all 14 combined stack classifiers had to agree on each instance added to the training model during the following iteration of the adaptive semi-supervised learning mechanism. We explored the impact of using four different thresholds: *n*, *n*-2, *n*-4, and *n*-6. These four thresholds were applied to MF as well as MF-BC/SC. Hence, MF was run using thresholds of 14, 12, 10, and 8, while MF-BC/SC (which only used 10 stack classifiers) was run using thresholds of 10, 8, 6, and 4.

The results are displayed in Table G5. For MF, using an ASL threshold of 10 yielded the best performance in terms of legitimate and fraud recall as well as error cost per instance, for both the investor and regulator settings. Using an ASL threshold of 8 provided the best performance for MF-BC/SC; this setting resulted in the best overall performance across all settings investigated in the appendix, with legitimate and fraud recall rates above 92 percent and 84 percent, respectively. MF – BC/SC with an ASL threshold of 8 improved legit recall by over 3 percent and fraud recall by over 2 percent and lowered regulator and investor costs by approximately $19,000 and $68,000 per firm-year, respectively, over MF using an ASL threshold of

Table G5. Impact of ASL Threshold on Performance

Setting	ASL Thresh	Regulators (Cost 1:10)					Investors (Cost 1:20)				
		Legit		Fraud		Cost Per Instance	Legit		Fraud		Cost Per Instance
		Prec.	Rec.	Prec.	Rec.		Prec.	Rec.	Prec.	Rec.	
MetaFraud (MF)	14	98.3	90.2	41.8	81.5	$100.5	98.4	89.6	40.8	82.9	$441.1
	12	98.3	89.8	41.0	82.2	$99.8	98.3	89.2	39.7	82.4	$455.4
	10	98.4	91.8	46.6	82.6	$90.0	98.5	90.0	42.0	83.6	$422.9
	8	98.3	89.5	40.1	81.7	$102.7	98.3	89.1	39.6	82.6	$451.6
MF - BC/SC	10	98.4	92.1	47.6	83.1	$87.3	98.4	91.1	44.8	83.4	$415.5
	8	98.5	92.8	50.2	84.1	$81.2	98.6	92.4	49.1	84.8	$373.3
	6	98.3	91.5	45.5	82.2	$92.8	98.4	91.0	44.4	82.9	$425.7
	4	98.4	91.0	44.4	83.1	$91.7	98.4	90.8	43.7	82.4	$437.4

The analysis performed in this appendix suggests that tuning the base classifier and stack classifier arrangements as well as selecting a threshold for ASL can collectively further improve detection performance by 2 percent to 3 percent. However, the results presented for MF-BC/SC with an ASL threshold of 8 were attained by retrospectively fitting the model configurations to the test data. Hence, they represent a theoretical/potential local optima level of performance. In practice, tuning the base and stack classifiers on the training data without over-fitting the data represents an interesting and important future research direction. Similarly, identifying methods for configuring the ASL threshold to better exploit new information signifies a worthwhile future endeavor. However, it is important to note that even in the absence of extensive tuning, MetaFraud attained excellent performance results. Given that the proposed framework provides enhanced fraud detection capabilities without considerable model tuning suggests that MetaFraud is highly robust and not overly sensitive to different configuration issues or parameter settings.

References

Beneish, M. D. 1999a. "The Detection of Earnings Manipulation," *Financial Analysts Journal* (55:5), pp. 24-36.

Beneish, M. D. 1999b. "Incentives and Penalties Related to Earnings Overstatements that Violate GAAP," *The Accounting Review* (74:4), pp. 425-457.

Cecchini, M., Aytug, H., Koehler, G., and Pathak, P. 2010. "Detecting Management Fraud in Public Companies," *Management Science* (56:7), pp. 1146-1160.

Cox, R. A. K., and Weirich, T. R. 2002. "The Stock Market Reaction to Fraudulent Financial Reporting," *Managerial Auditing Journal* (17:7), pp. 374-382.

Gaganis, C. 2009. "Classification Techniques for the Identification of Falsified Financial Statements: A Comparative Analysis," *International Journal of Intelligent Systems in Accounting and Finance Management* (16), pp. 207-229.

Kirkos, E., Spathis, C., and Manolopoulos, Y. 2007. "Data Mining Techniques for the Detection of Fraudulent Financial Statements," *Expert Systems with Applications* (32), pp. 995-1003.

Zhou, Z. and Li, M. 2005. "Tri-Training: Exploiting Unlabeled Data Using Three Classifiers," *IEEE Transactions on Knowledge and Data Engineering* (17:11), pp. 1529-1541.