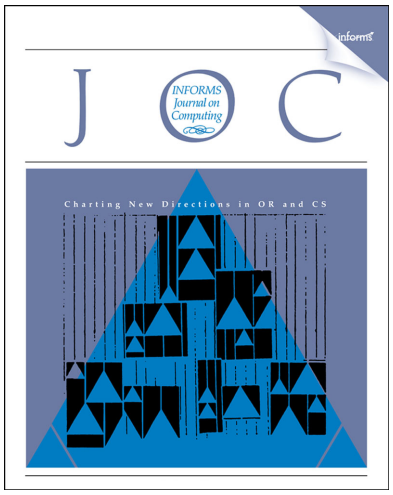


INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

How to Deal with Liars? Designing Intelligent Rule-Based Expert Systems to Increase Accuracy or Reduce Cost

Yuanfeng Cai, Zhengrui Jiang, Vijay Mookerjee

To cite this article:
Yuanfeng Cai, Zhengrui Jiang, Vijay Mookerjee (2017) How to Deal with Liars? Designing Intelligent Rule-Based Expert Systems to Increase Accuracy or Reduce Cost. INFORMS Journal on Computing 29(2):268-286. https://doi.org/10.1287/ijoc.2016.0728

http://pubsonline.informs.org/journal/ijoc/

INFORMS JOURNAL ON COMPUTING
Vol. 29, No. 2, Spring 2017, pp. 268–286
ISSN 1091-9856 (print), ISSN 1526-5528 (online)

How to Deal with Liars? Designing Intelligent Rule-Based Expert Systems to Increase Accuracy or Reduce Cost

Yuanfeng Cai,[a] **Zhengrui Jiang,**[b] **Vijay Mookerjee**[c]

[a] Zicklin School of Business, City University of New York – Baruch College, New York, New York 10010; [b] College of Business, Iowa State University, Ames, Iowa 50011; [c] Naveen Jindal School of Management, University of Texas at Dallas, Richardson, Texas 75080

Contact: yuanfeng.cai@baruch.cuny.edu (YC); zjiang@iastate.edu (ZJ); vijaym@utdallas.edu (VM)

Received: May 25, 2015
Revised: December 9, 2015; June 14, 2016
Accepted: July 3, 2016
Published Online: March 30, 2017

https://doi.org/10.1287/ijoc.2016.0728

Abstract. Input distortion is a common problem faced by expert systems, particularly those deployed with a Web interface. In this study, we develop novel methods to distinguish liars from truth-tellers, and redesign rule-based expert systems to address such a problem. The four proposed methods are termed *split tree* (ST), *consolidated tree* (CT), *value-based split tree* (VST), and *value-based consolidated tree* (VCT), respectively. Among them, ST and CT aim to increase an expert system's accuracy of recommendations, and VST and VCT attempt to reduce the misclassification cost resulting from incorrect recommendations. We observe that ST and VST are less efficient than CT and VCT in that ST and VST always require selected attribute values to be verified, whereas CT and VCT do not require value verification under certain input scenarios. We conduct experiments to compare the performances of the four proposed methods and two existing methods, i.e., the traditional *true tree* (TT) method that ignores input distortion and the *knowledge modification* (KM) method proposed in prior research. The results show that CT and ST consistently rank first and second, respectively, in maximizing the recommendation accuracy, and VCT and VST always lead to the lowest and second lowest misclassification cost. Therefore, CT and VCT should be the methods of choice in dealing with users' lying behaviors. Furthermore, we find that KM is outperformed by not only the four proposed methods, but sometimes even by the TT method. This result further confirms the advantage necessity of differentiating liars from truth-tellers when both types of users exist in the population.

History: Accepted by Ram Ramesh, Area Editor for Knowledge Management and Machine Learning.
Supplemental Material: A supplemental video can be found at https://doi.org/10.1287/ijoc.2016.0728.

Keywords: rule-base expert systems • input distortion • misclassification • value-based methods

1. Introduction

Since their inception in the mid-1960s, expert systems have gained tremendous popularity and have been broadly applied in a wide variety of fields including finance, medical treatment, military, and education (Liao 2005). The benefits of expert systems, such as helping organizations reduce their personnel cost and make better decisions, have also been well documented (Duan et al. 2005).

Based on their methods of reasoning, expert systems can be classified into different categories. In this study, we focus on rule-based expert systems, which is one of the most frequently applied types of expert systems (Liao 2005). Rule-based expert systems are built from a set of IF-THEN rules. Such rules can be provided by domain experts or inferred from historical data. When domain experts construct such decision rules, they typically assume that the inputs provided by users at the time of system consultation are accurate. However, as we will discuss next, this is often not the case in real-world settings.

1.1. Input Distortion

Input distortion occurs when users do not provide true data to a rule-based expert system. For instance, Internet technologies have provided new opportunities for the deployment of expert systems (Power 2000). One of the challenges faced by such systems is users' lying behaviors. Hoffman et al. (1999) find that 95% of the users are reluctant to provide information requested by websites. One reason behind this behavior is the lack of trust between customers and businesses on the web (Metzger 2004). Users' concerns about their privacy and information security prevent them from revealing true information. Consequently, users may falsify input data to protect themselves. Another important factor that contributes to this lying behavior is that self-interested customers may deliberately seek improper benefits by providing incorrect data. For example, during a credit card application, users who are not confident about their financial background may manipulate their personal data to get approval. Regardless of the causes of lying, firms can incur significant costs as a result of input distortion.

One intuitive method to deal with input distortion is to impose penalty to those caught lying. Keefer (2015) has reported that falsifying data in a credit application could lead to jail time and expensive fines. However, punishments are often costly to enforce, and thus may have limited effect on the prevention of input distortion.

Since input distortion is practically impossible to completely eliminate, one may suggest that all user inputs be manually verified to ascertain their accuracy. However, manually verifying user inputs for frequently used expert systems is typically costly and time-consuming. In this study, we focus on automatic approaches to address users' lying to rule-based expert systems.

1.2. Literature Review

Lying behavior has long been studied by researchers. One research stream deals with deception detection. Previous studies suggest that it is possible to use verbal and nonverbal cues to detect deception (Buller and Burgoon 1996, George et al. 2004). In addition, researchers have proposed methods to detect deception via linguistic cues (Zhou et al. 2003, 2008; Zhou and Zhang 2008). However, these techniques for deception detection cannot be directly applied to address users' lying behavior in our problem context. For example, during an online credit card application, an applicant may only input numeric and simple text information (e.g., name and address). Without face-to-face contact, it is impossible to capture nonverbal or verbal cues that are critically important for deception detection. Similarly, without rich text information, the linguistic methods cannot be applied. Furthermore, the aforementioned studies on deception detection do not address input distortion for rule-based expert systems.

In another stream of investigation, researchers have developed sophisticated techniques to detect various types of fraud, such as management fraud (Cecchini et al. 2010), financial fraud (Abbasi et al. 2012), and fake websites (Abbasi et al. 2010). The methods proposed in this stream of research typically require training data that includes a set of fraud cues or contextual information as well as known classifications. Such techniques cannot be directly applied to the problem addressed in this study when the required fraud clues and contextual information are unavailable.

The prior literature has also proposed methods to handle noise in the knowledge base that can affect the effectiveness of expert systems. One way of dealing with noisy data is enhancing the quality of training data. Various solutions, such as class noise identification (Zhu et al. 2003) and missing attribute value imputation (Rubin 2004) have been proposed. More recently, Boylu et al. (2010) adapt support vector machines (SVM) to generate classifications; their method takes into consideration users' possible strategic behavior such as data distortion. These solutions require that a similar error pattern exist in training and testing data, and hence cannot be used to address users' input distortions that only occur at the time of system consultation.

To the best of our knowledge, only one prior study addresses the problem considered in the present research, i.e., distorted data are fed into a rule-based expert system that assumes all input values are correct. To cope with such input distortion, Jiang et al. (2005) proposed two novel methods to improve the accuracy of recommendations. The first method, *knowledge modification* (or KM), generates a new decision tree (termed *KM tree*) based on experts' decision rules as well as users' lying patterns. At the time of system consultation, users' input data is directly fed into the modified decisions tree. The second method, *input modification* (IM), still uses the decision tree built from decision rules provided by experts, but modifies a user's input data at the time of consultation. The prior study shows that both KM and IM lead to a significantly improved accuracy rate than the traditional methods that ignore input distortion, with KM outperforming IM in practically all tested scenarios.

Although the KM method proposed by Jiang et al. (2005) increases the accuracy of recommendations, the method has two limitations. First, the generated KM tree does not differentiate liars from truth-tellers at the time of consultation. Since the KM method essentially assumes that every user is a liar, it is not effective when there are both liars and truth-tellers in the underlying user population. Second, the KM method does not consider misclassification costs when making recommendations. In real-world settings, misclassification costs are often asymmetric. For instance, classifying a nontrustworthy customer as a trustworthy one could be more costly than classifying a trustworthy customer as a nontrustworthy one. The KM method maximizes the expected accuracy of recommendations while completely ignoring such misclassification costs. This could lead to suboptimal decisions under real-world applications.

1.3. Contributions

To the best of our knowledge, no prior study differentiates liars from truth-tellers, and considers misclassification costs when dealing with users' input noise for expert systems. The present study fills this void and makes two important contributions. First, we differentiate between liars and truth-tellers in all methods proposed in this study. By comparing the reported value and the verified true value of a selected attribute, we are able to calculate the probability that a user is a liar, and the user may be treated differently based on the calculated probability. The first two methods we

propose are *accuracy-based* and termed *split tree* (ST) and *consolidated tree* (CT), respectively. The difference between them is that ST includes one tree branch for liars and another for truth-tellers, while CT uses a single tree for all users. Both ST and CT deliver better accuracy than other benchmark methods, with CT being the more superior method.

As the second major contribution, we propose two *value-based* methods that minimize the total misclassification cost. The first value-based method extends ST, therefore it is termed *value-based split tree* (VST). The second method extends CT and hence is named *value-based consolidated tree* (VCT). Both VST and VCT outperform benchmark methods in minimizing the misclassification cost, with a slight edge belonging to VCT.

The rest of the paper is organized as follows. We present the two accuracy-based methods in Section 2 and the two value-based methods in Section 3. In Section 4, we report on the experiments for performance evaluation. In Section 5, we address some practical issues in the selection of attributes for verification. In Section 6, we extend the proposed methods to cope with a more heterogeneous user population with multiple groups. Section 7 discusses the applicability and the practical guidelines of the proposed methods. Finally, we conclude the paper in Section 8 with discussions on managerial implications and a future research direction.

2. Accuracy-Based Methods

To differentiate it from decision trees generated from other methods, we refer to the decision tree built directly from expert-provided decision rules as the *true tree* (TT). When decision rules are formulated, it is implicitly assumed that all attribute values provided by users are accurate. For instance, a decision rule may classify a customer who claims to have medium income and full-time employment as low-risk. An implicit assumption behind this decision rule is that the customer indeed has a medium income and a full-time job. However, as discussed earlier, users may lie when providing inputs to a rule-based expert system. If a customer who claims to have a medium income and a full-time job is actually unemployed with no stable income, then classifying the customer as low-risk can lead to financial losses. As previously mentioned, facing such lying behavior, a *KM tree*, built based on the KM method proposed by Jiang et al. (2005), can replace the TT to serve as the "expert" at the time of consultation. The KM Tree, however, does not differentiate liars from truth-tellers, and hence leaves room for improvement.

In this section, we propose two accuracy-based methods, termed *split tree* (ST) and *consolidated tree* (CT), that differentiate liars from truth-tellers. To illustrate these methods, we first present a credit risk assessment example similar to the one used by Jiang et al. (2005).

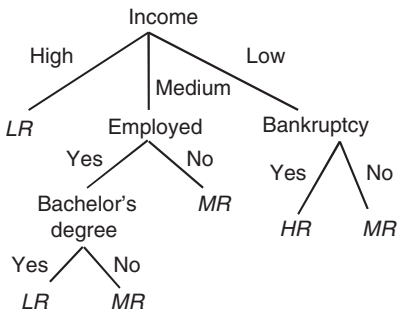
2.1. A Credit Risk Assessment Example

In order to decide whether to provide credit to potential customers, firms need to first assess their creditworthiness. From reliable historical data or their own experience, human experts could derive a set of decision rules that, as illustrated in Figure 1(a), can be used for such an assessment. According to rules represented in this figure, a customer can be classified into three risk levels, i.e., low risk (LR), medium risk (MR), and high risk (HR), based on the values of four attributes: *income* (high, medium, low), *bachelor's degree* (yes, no), *employment* (yes, no), and *bankruptcy* (yes, no). The dash entry ("-") in the figure means that the value of that attribute "does not matter" in a given decision rule. For instance, rule R0 classifies a customer as low-risk as long as the customer's income is high, regardless of whether the customer has a bachelor's degree, a bankruptcy record, or a job. The decision rules shown in Figure 1(a) are derived based on the assumption that all attribute values are accurate, hence they represent the *true table*. Based on heuristic algorithms, the True Table can be translated into a true tree, as shown in Figure 1(b). At the time of consultation, the true tree, instead of the true table, should be used because the former is more efficient than the latter.

Figure 1. True Table and True Tree for Credit Risk Assessment

Rules \ Attributes	Income	Bachelor's degree	Employment	Bankruptcy	Classification
R0	H	—	—	—	LR
R1	M	Y	Y	—	LR
R2	M	—	N	—	MR
R3	M	N	Y	—	MR
R4	L	—	—	Y	HR
R5	L	—	—	N	MR

(a) Decision table



(b) True Tree

2.2. Probability of Being a Liar

All methods proposed in this study require the probability that a user is a liar be estimated. For the purpose of probability estimation, it is necessary to verify the value reported by the user for at least one attribute. We call such an attribute *verifiable attribute* (or VA). In the credit risk assessment example, *bankruptcy* could be used as a VA since its value can be relatively cost-efficiently obtained from a customer's credit report. Given both the reported and true values of a VA, denoted by VA^R and VA^T respectively, the conditional probability that the user is a liar can be derived based on Bayes' Theorem:

$$\begin{aligned}&\mathrm{P}(\text{liar} \mid VA^R, VA^T)\\ &= (\mathrm{P}(VA^R, VA^T \mid \text{liar}) \cdot \mathrm{P}(\text{liar}))\\ &\quad \cdot (\mathrm{P}(VA^R, VA^T \mid \text{liar})\mathrm{P}(\text{liar})\\ &\quad + \mathrm{P}(VA^R, VA^T \mid \text{truth-teller}) \cdot \mathrm{P}(\text{truth-teller}))^{-1}, \quad (1)\end{aligned}$$

where

$$\mathrm{P}(VA^R, VA^T \mid \text{liar}) = \mathrm{P}(VA^R \mid VA^T, \text{liar}) \cdot \mathrm{P}(VA^T, \text{liar}),$$

and

$$\begin{aligned}\mathrm{P}(VA^R, VA^T \mid \text{truth-teller}) = \mathrm{P}(VA^R \mid VA^T, \text{truth-teller})\\ \cdot \mathrm{P}(VA^T, \text{truth-teller}).\end{aligned}$$

Given (1), we have

$$\mathrm{P}(\text{truth-teller} \mid VA^R, VA^T) = 1 - \mathrm{P}(\text{liar} \mid VA^R, VA^T). \quad (2)$$

The conditional and marginal probabilities in (1) can be obtained from historical data or through sampling. During the sampling process, a certain number of users are selected and their true attribute values are verified. Users who lied about at least one attribute are classified as liars; those who did not lie about any attribute are classified as truth-tellers. Based on this classification, we can estimate the distribution of liars and truth-tellers in the population, as illustrated in Figure 2. In addition, we can estimate the *distortion matrixes* for liars and truth-tellers for each attribute, as shown in Figures 3(a) and 3(b). We then calculate the distortion matrices for the entire user population (including both truth-tellers and liars) for all attributes, as shown in Figure 3(c). In a distortion matrix, the rows represent the true values, the columns represent the reported values, and the numbers capture the conditional probability of every reported attribute value given each true value. For instance, the first 0.3 in the liar's distortion matrix for *income* implies that among those whose true income is low, 30% claimed that their income was high. Through sampling, we can also estimate the marginal distribution of true attribute values for both liars and truth-tellers, as shown in Figures 4(a) and 4(b). The marginal distributions for the entire user population are calculated and shown in Figure 4(c). Note that these distortion matrices and marginal distributions are similar to those in Jiang et al. (2005). However, the prior study does not estimate them separately for liars and truth-tellers, but instead uses the population parameter values as shown in Figures 3(c) and 4(c).

Figure 2. Distribution of Liar and Truth-Teller

Liar (L)	0.4
Truth-teller (T)	0.6

Based on the distributions and the conditional probabilities summarized in the distortion matrices, we can estimate the probability that a user is a liar given the *true* and *reported* values of the VA. To illustrate, consider a reported vector (I^R = "H," D^R = "N," E^R = "N," B^R = "N"), representing the reported values of *income, bachelor's degree, employment*, and *bankruptcy*, and a verified true *VA* (bankruptcy) value B^T = "N." Using values shown in Figures 2, 4(a), 4(b), 4(c) and Equation (1), we obtain:

$$\begin{aligned}&\mathrm{P}(B^R = \text{"N,"}\ B^T = \text{"N"} \mid \text{liar})\\ &\quad = \mathrm{P}(B^R = \text{"N"} \mid B^T = \text{"N,"} \text{liar}) \cdot \mathrm{P}(B^T = \text{"N,"} \text{liar})\\ &\quad = 0.90 \cdot 0.55,\\ &\mathrm{P}(B^R = \text{"N,"}\ B^T = \text{"N"} \mid \text{truth-teller})\\ &\quad = \mathrm{P}(B^R = \text{"N"} \mid B^T = \text{"N,"} \text{truth-teller})\\ &\quad\quad \cdot \mathrm{P}(B^T = \text{"N,"} \text{truth-teller}) = 1.0 * 0.97.\end{aligned}$$

Hence, $\mathrm{P}(\text{liar} \mid B^R = \text{"N,"}\ B^T = \text{"N"})$

$$\begin{aligned}&= (\mathrm{P}(B^R = \text{"N,"}\ B^T = \text{"N"} \mid \text{liar}) \cdot \mathrm{P}(\text{liar}))\\ &\quad \cdot (\mathrm{P}(B^R = \text{"N,"}\ B^T = \text{"N"} \mid \text{liar}) \cdot \mathrm{P}(\text{liar})\\ &\quad + \mathrm{P}(B^R = \text{"N,"}\ B^T = \text{"N"} \mid \text{truth-teller}) \cdot \mathrm{P}(\text{truth-teller}))^{-1}\\ &= \frac{0.90 \cdot 0.55 \cdot 0.4}{0.90 \cdot 0.55 \cdot 0.4 + 1.0 \cdot 0.97 \cdot 0.6} = 0.254.\end{aligned}$$

This result shows that even if both B^R and B^T are "N," the user still has a 25.4% chance of being a liar. This has an important implication. Even though the reported value is the same as the true one, it does not guarantee that the customer is a truth-teller. Conversely, we are certain that a user is a liar if the reported value is different from the true one. This can be derived based on (1): If B^R is not equal to B^T, $\mathrm{P}(B^R \mid B^T, \text{truth-teller}) = 0$, hence $\mathrm{P}(\text{liar} \mid B^R \neq B^T) = 1$.

2.3. Split Tree Method (ST)

The *split tree* (ST) method uses a True Tree and a tree specifically built for liars, named *liar tree*, to generate recommendations. As illustrated in Figure 5, whether the true tree or the liar tree is consulted depends on whether the probability of being a liar is below or above a threshold. For instance, if the threshold is set to 0.5, the true tree should be consulted if the probability of a user being a liar is 0.254, as calculated

Figure 3(a). Liars' Distortion Matrices for Each Attribute

Income (I)

	High	Medium	Low
High	0.85	0.075	0.075
Medium	0.225	0.55	0.225
Low	0.3	0.3	0.4

Bachelor's degree (D)

	Yes	No
Yes	0.9	0.1
No	0.4	0.6

Employed (E)

	Yes	No
Yes	0.85	0.15
No	0.90	0.10

Bankruptcy (B)

	Yes	No
Yes	0.11	0.89
No	0.10	0.90

Figure 3(b). Truth-Tellers' Distortion Matrices for Each Attribute

Income (I)

	High	Medium	Low
High	1.0	0.0	0.0
Medium	0.0	1.0	0.0
Low	0.0	0.0	1.0

Bachelor's degree (D)

	Yes	No
Yes	1.0	0.0
No	0.0	1.0

Employed (E)

	Yes	No
Yes	1.0	0.0
No	0.0	1.0

Bankruptcy (B)

	Yes	No
Yes	1.0	0.0
No	0.0	1.0

Figure 3(c). The Entire Population's Distortion Matrices for Each Attribute

Income (I)

	High	Medium	Low
High	0.95	0.025	0.025
Medium	0.09	0.82	0.09
Low	0.245	0.245	0.51

Bachelor's degree (D)

	Yes	No
Yes	0.97	0.03
No	0.2	0.8

Employed (E)

	Yes	No
Yes	0.94	0.06
No	0.42	0.58

Bankruptcy (B)

	Yes	No
Yes	0.2	0.8
No	0.03	0.97

Figure 4(a). Liars' Marginal Distributions for Each Attribute

Income (I)

High	0.5
Medium	0.3
Low	0.2

Bachelor's degree (D)

Yes	0.55
No	0.45

Employed (E)

Yes	0.65
No	0.35

Bankruptcy (B)

Yes	0.45
No	0.55

Figure 4(b). Truth-Tellers' Marginal Distributions for Each Attribute

Income (I)

High	0.67
Medium	0.3
Low	0.03

Bachelor's degree (D)

Yes	0.7
No	0.3

Employed (E)

Yes	0.73
No	0.27

Bankruptcy (B)

Yes	0.03
No	0.97

Figure 4(c). The Entire Population's Marginal Distributions for Each Attribute

Income (I)

High	0.6
Medium	0.3
Low	0.1

Bachelor's degree (D)

Yes	0.64
No	0.36

Employed (E)

Yes	0.7
No	0.3

Bankruptcy (B)

Yes	0.2
No	0.8

in the preceding example. Conversely, the liar tree is consulted if the probability is higher than 0.5. Under the ST method, the true tree is directly constructed from the expert-provided decision rules; the construction of a liar tree is similar to that of a KM tree (Jiang et al. 2005), with the exception that the liar's distortion matrices and marginal distributions, instead of those for all users, are used. We next briefly describe the steps of building the liar tree.

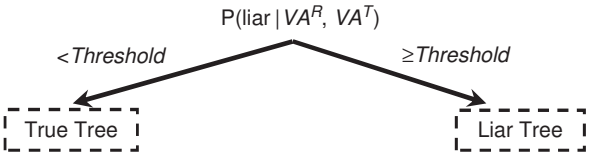
We first construct the *liar's decision table* (or *liar table*), which includes recommendations for all possible reported vectors. For instance, for the credit risk assessment example, there are $3\times2\times2\times2=24$ rules in the liar table. Similarly, if there are 10 binary attributes, the fully enumerated liar table will include 2^{10} rules. The recommendation for each possible reported vector is computed using the following steps:

Step 1. *Given a reported vector, calculate the probabilities associated with every possible true vector*: Denote the reported and a true vector by **Reported** and **True**, respectively. The conditional probability is

$$\mathrm{P}(\mathbf{True}\,|\,\mathbf{Reported})=\frac{\mathrm{P}(\mathbf{Reported}\,|\,\mathbf{True})*\mathrm{P}(\mathbf{True})}{\sum_r \mathrm{P}(\mathbf{Reported}\,|\,\mathbf{True}^r)*\mathrm{P}(\mathbf{True}^r)}, \tag{3}$$

where r is the index over all true vectors. If the number of possible true vectors is large, we can assume

Figure 5. Split Tree Structure



$P(\textbf{True}) = \prod_i P(True_i)$, where $True_i$ represents the value of attribute i in the **True** vector. Similar to the well-known naïve Bayes method, we adopt two other assumptions:

$$P(\textbf{Reported} \mid \textbf{True}) = \prod_i P(\text{Reported}_i \mid \textbf{True}) \quad \text{for all } i, \text{ and}$$

$$P(Reported_i \mid \textbf{True}) = \prod_i P(Reported_i \mid True_i) \quad \text{for all } i.$$

Then,

$$P(\textbf{Reported} \mid \textbf{True}) = \prod_i P(Reported_i \mid True_i). \quad (4)$$

Note that in calculating the conditional probabilities, we need to use the distortion matrices and marginal distributions for liars, as illustrated in Figures 3(a) and 4(a), respectively.

Step 2. *Find the Liar Table recommendation*: For every possible true vector, use the True Tree to obtain the recommendation. Add the conditional probabilities associated with all true vectors that have the same recommendation. The recommendation with the highest total probability is selected as the liar table recommendation for the **Reported** vector, because this recommendation is most likely to be the accurate one for a user with the **Reported** vector.

Step 3. *Generate and Condense the Liar Table*: Repeat Steps 1 and 2 for all possible reported vectors to generate the fully enumerated Liar Table. In the full Liar Table, sometimes the value of a given attribute does not affect the recommendation. Therefore, we can collapse each set of "redundant" decision rules into a single row, similar to those shown in Figure 1(a). Once all such redundancies are removed, we obtain a *condensed* liar table.

Step 4. *Build the Liar Tree from the condensed Liar Table*: The heuristic used to build the True Tree can be used to construct the liar tree (or LT for short).

Using data shown in Figures 2, 4(a), 4(b), and 4(c), and following Steps 1–4, we obtain a Liar Tree for the credit risk assessment example. The completed split tree is shown in Figure 6. For better clarity, the true tree structure is not included in the figure.

2.4. Consolidated Tree Method (CT)

With the ST method, the VA is always verified before we can decide which tree branch to traverse. After additional analysis, we find that under certain scenarios, the true value of the VA does not affect the recommendation. Motivated by this observation, we develop an alternative method to deal with users' input distortion, under which the true value of the VA is simply treated as a *separate* attribute in the decision

Figure 6. Complete Split Tree

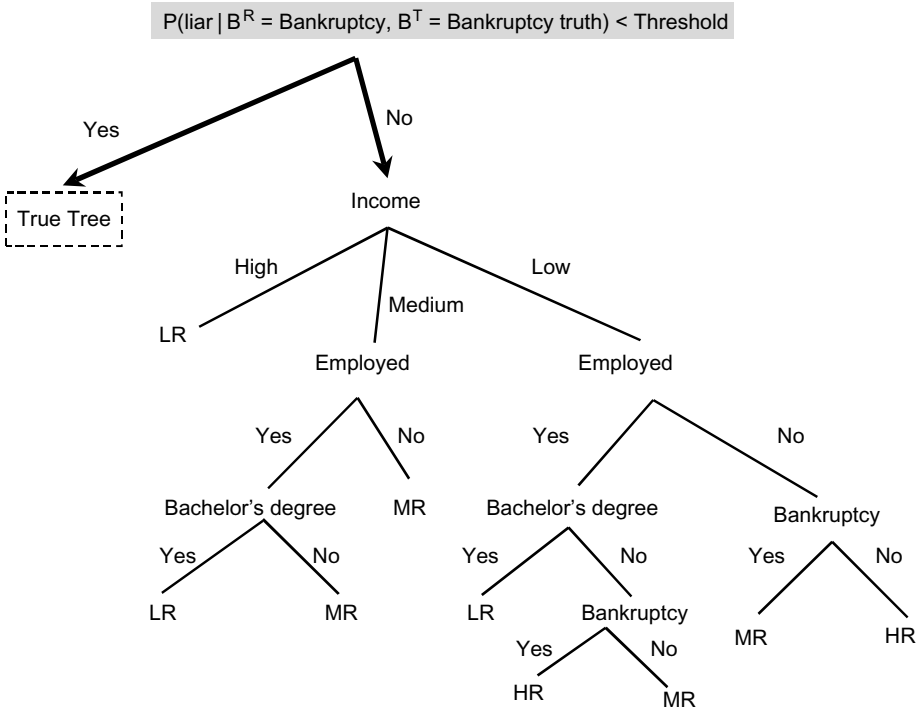


table. Then, each vector in the expanded decision table includes the reported values of all attributes as well as the true value of the VA. The recommendation for this vector is the one with the highest probability of being correct given the available information. The expanded decision table is then condensed and transformed into a single tree. We call this method *consolidated tree* (or CT) method because, unlike ST, there are no separate tree branches for liars and truth-tellers. We next describe the CT method in detail.

For each expanded vector that includes the reported attribute values as well as the true value of the VA, repeat Steps 1–4 to obtain the corresponding CT recommendation.

Step 1. *Find the probability that the user with the given vector is a liar*: Since the given vector includes both the reported value and the true value of the VA, the probability that the user is a liar or a truth-teller can be calculated based on formulas (1) and (2). We denote these probabilities by P(liar $\mid VA^R, VA^T$) and P(truth-teller $\mid VA^R, VA^T$), respectively.

Step 2. *Calculate LT path probability associated with each possible recommendation*: The CT method considers the probability that a user is a liar as well as the probability that she is a truth-teller. If the user is a liar, then similar to Step 1 in building the liar tree, we calculate the conditional probabilities associated with all possible true vectors and then sum up the conditional probabilities of all true vectors that have the same recommendation. Using the credit risk assessment example, we denote the total conditional probabilities associated with low-, medium-, and high-risk by P(LR), P(MR), and P(HR), respectively. Since these probabilities are relevant only if the user is a liar, we multiply each of them by P(liar $\mid VA^R, VA^T$) and obtain P(liar $\mid VA^R, VA^T) \cdot$P(LR) , P(liar $\mid VA^R, VA^T) \cdot$ P(MR), and P(liar $\mid VA^R, VA^T) \cdot$ P(HR). Each of these probabilities is referred to as *LT path probability* associated with a recommendation.

Step 3. *Calculate the TT path probability*: If P(truth-teller $\mid VA^R, VA^T) > 0$, feed the reported values into the True Tree, and obtain the *true recommendation*. Since there is no uncertainty in the truth tree path, we set the *TT path probability* associated with the true recommendation as P(truth-teller $\mid VA^R, VA^T$) and that associated with all other recommendations as zero.

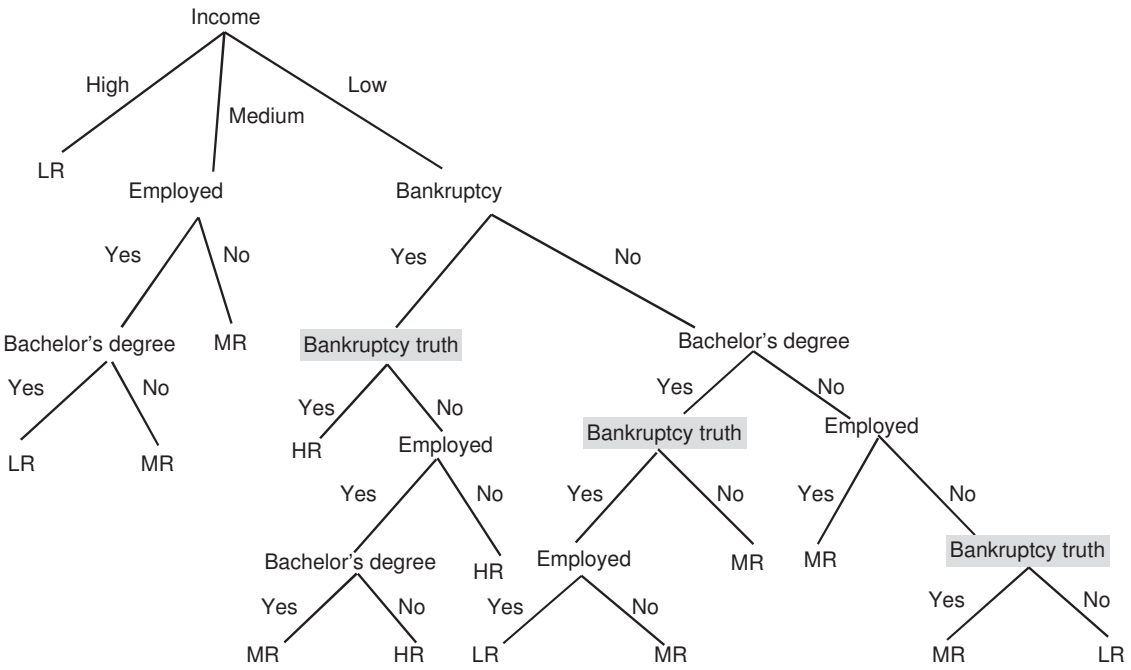
Step 4. *Obtain the CT recommendation for the given vector*: Add the LT and TT path probabilities for the same recommendation. The recommendation with the highest probability sum is selected as the CT recommendation for the given vector. For instance, if the True recommendation is MR in the credit risk assessment example, then we compare P(liar $\mid VA^R, VA^T) \cdot$P(MR) + P(truth-teller $\mid VA^R, VA^T$) with P(liar $\mid VA^R, VA^T) \cdot$ P(LR) and P(liar $\mid VA^R, VA^T) \cdot$ P(HR). The recommendation with the highest probability is selected as the *CT recommendation*.

Step 5. *Repeat Steps* 1–4 *for all possible vectors to generate the fully enumerated CT table.*

Step 6. *Condense the CT table and transform it into a CT tree.*

Following Steps 1–6, we construct a consolidated tree for the credit assessment example as shown in Figure 7. Note that "*bankruptcy truth*" is treated as the fifth attribute in the tree. Comparing it with the split

Figure 7. Consolidated Tree



tree shown in Figure 6, we find that these two trees do not always lead to the same recommendation. Furthermore, Figure 7 shows that a user's true *bankruptcy* value does not need to be verified in every branch. This can lead to significant cost savings because verifying the true value of a VA takes time and incurs costs. Therefore, the CT method is more efficient than the ST method.

3. Value-Based Methods

The ST and CT methods attempt to maximize the accuracy of recommendations. An implicit assumption made in the two methods is that the misclassification cost remains the same for all misclassification scenarios. In reality, this may not be the case. For instance, incorrectly classifying a low-risk customer as a high-risk one may lead to the denial of loan and hence the loss of opportunity to earn interest from the customer. On the other hand, misclassifying a high-risk customer as a low-risk one may lead to default, a potentially much more costly outcome. In this section, we extend the two accuracy-based methods to the corresponding value-based methods: *Value-based split tree* (VST) and *value-based consolidated tree* (VCT). The main difference between the accuracy-based and the value-based methods is that the former maximizes the accuracy of recommendations, while the latter minimizes the misclassification costs.

To use the value-based methods, we need to know the misclassification costs under different scenarios, which can be summarized in a *misclassification cost matrix*. Figure 8 shows a hypothetical matrix for the credit risk assessment example. The columns represent the true class, and the rows represent the recommended class. In real-world settings, misclassification costs can be estimated by domain experts or from historical data. For instance, financial institutions often collect data regarding loan default rates from customers at different risk levels.[1] Such data could be used to construct the misclassification cost matrix.

3.1. Value-Based Split Tree Method (VST)

The input of VST is similar to that of the ST method, except that VST takes into consideration the misclassification costs when deciding the best recommendation. Specifically, the method traverses the true tree branch when the probability of a liar is below a threshold. Otherwise, it traverses the *value-based liar tree* branch.

Figure 8. Misclassification Cost Matrix

	LR	MR	HR
LR	0	10	20
MR	45	0	28
HR	100	50	0

Regarding the construction procedure, the value-based liar tree and liar tree differ only in Step 2 that generates the (value-based) liar table recommendations. In Step 2 of the value-based liar tree construction, we first sum up the probabilities of all possible true vectors with the same recommendation, then calculate the misclassification costs under different scenarios, and finally select the recommendation that minimizes the total misclassification cost instead of the one with the highest probability.

Formally, let GR denote the generated recommendation and AR the accurate recommendation. The recommendation with the lowest misclassification cost can be obtained by

$$\arg\min_{GR^q} \sum_l C(GR^q \mid AR^l) \cdot \mathrm{P}(AR^l), \tag{5}$$

where l is the index over all possible accurate recommendations, q is the index over all possible generated recommendations, $\mathrm{P}(AR^l)$ is the same probability for AR^l as that obtained in Step 2 of the ST method, and $C(GR^q \mid AR^l)$ is the misclassification cost and can be found from the misclassification cost matrix.

Figure 9 shows the value-based split tree for the credit risk assessment example. Similar to ST, VST produces a tree structure that includes a True Tree branch and a valued-based liar tree branch. Once again, the branch traversed at the time of consultation depends on whether the probability of a user being a liar is above or below the given threshold.

3.2. Value-Based Consolidated Tree Method (VCT)

Analogous to the extension from ST to VST, we now extend the accuracy-based CT method to produce a *value-based consolidated tree* (VCT) that minimizes the expected total misclassification cost. Similar to the difference between ST and VST, the procedures for CT and VCT differ only in the step that selects the CT (VCT) recommendations. Specifically, in Step 4 of the VCT method, once the sum of the LT and TT path probabilities associated with every recommendation are obtained, we use formula (5) to obtain the expected total misclassification cost $C(GR^q)$ associated with each recommendation GR^q, and then select the recommendation with the lowest total misclassification cost as the *VCT recommendation*. Figure 10 illustrates the VCT for the credit risk assessment example. Again, VST always requires the verification of the true value of the VA, whereas VCT needs it only for some tree branches. Furthermore, the differences between Figures 6 and 9 or those between Figures 7 and 10 show that the most accurate recommendation is not always the one that minimizes the misclassification cost, implying that pursuing a higher accuracy does not always lead to a lower cost. This confirms that the value-based methods are indeed necessary, especially when the misclassification costs demonstrate a high level of asymmetry.

Figure 9. Value-Based Split Tree

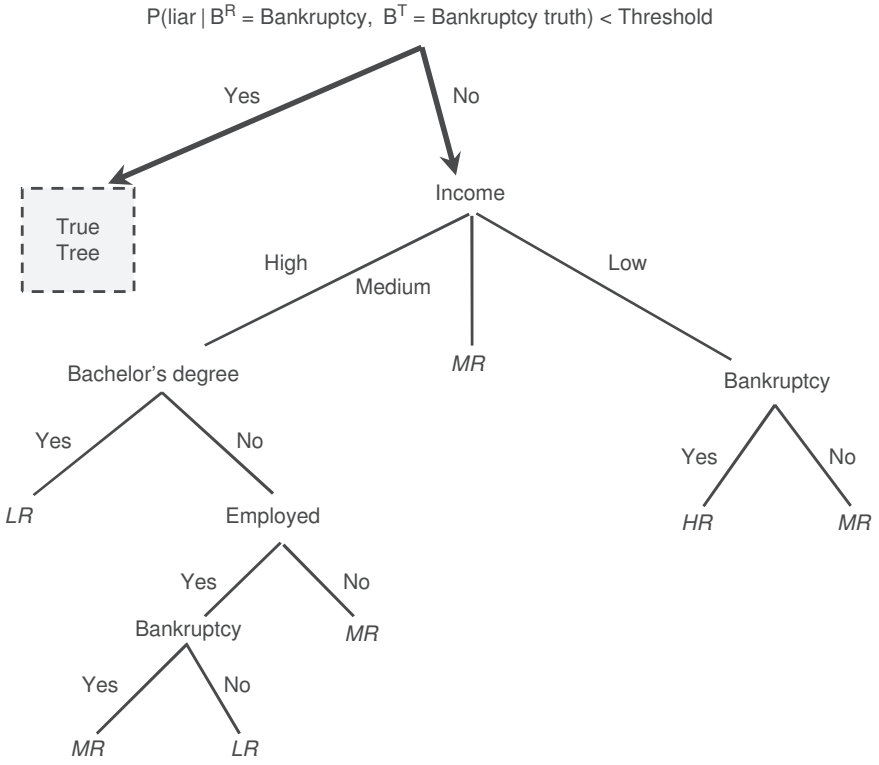
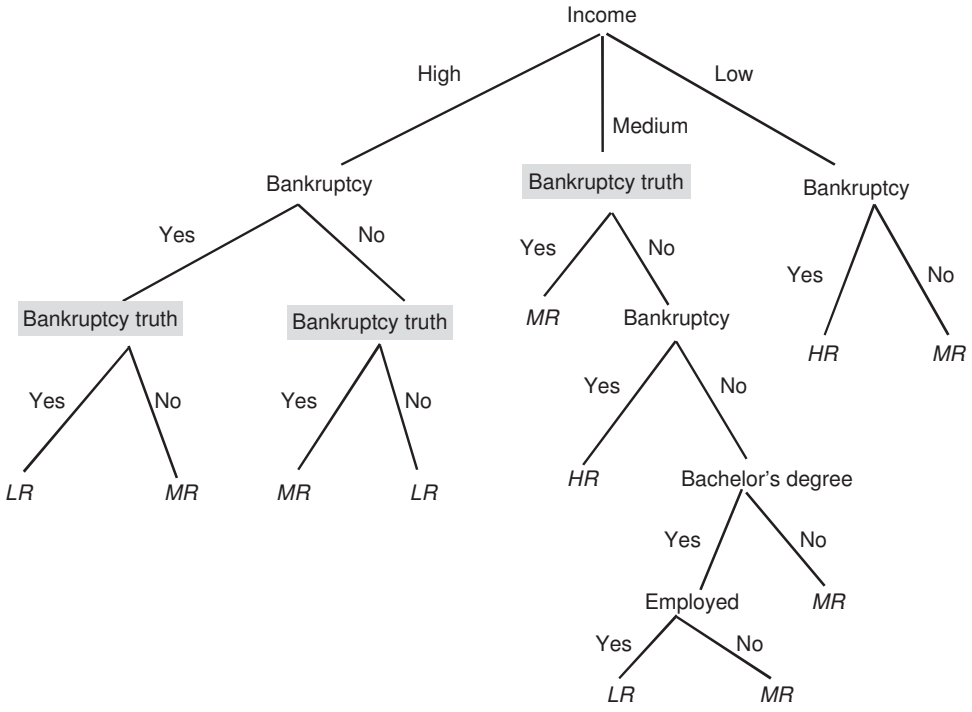


Figure 10. Value-Based Consolidated Tree



4. Experiments for Performance Evaluation

We conduct a series of experiments to evaluate the performances of the proposed methods against existing methods. The first set of experiments use a payment default risk data set provided by a telecom service company in China. The data set contains 6,783 customer service/default risk records and is used to assess the default risk level of a customer, i.e., either low risk (LR) or high risk (HR). The data set contains eight predictor attributes: *residence status* (village/city), *customer type* (family/individual), *identity type* (business license/ID card), *service status* (active/inactive/suspend), *credit score* (low/medium/high), and *intended payment method* (buyout/cash/withhold/collection) are categorical attributes, and *age* and *service duration* (i.e., the continuous time duration of the current service) are numerical attributes. We adopt the discretization algorithm developed by Fayyad and Irani (1993) to discretize the two continuous attributes, as it was used in a previous credit-risk analysis study (Baesens et al. 2003). Specifically, *age* is assigned a value of "younger" if a customer's age is ≤40, and "elder" otherwise. Similarly, *service duration* takes a value of "short" if the duration of service is ≤4 years, and "long" otherwise. Because this data set is highly reliable, we treat the tree generated from the data set, as shown in Figure 11, as the true tree in our subsequent experiments.

4.1. Experiment Procedure

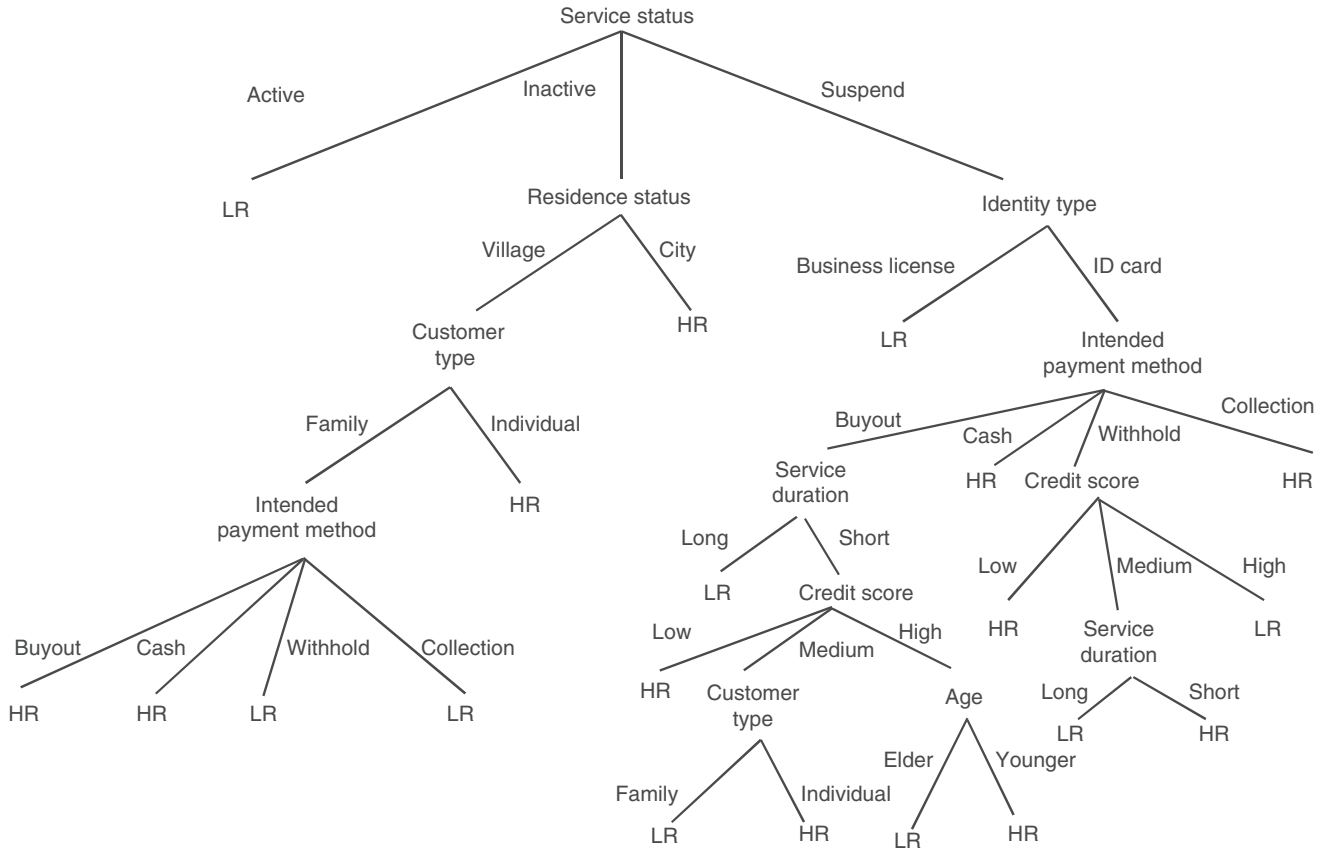
We first describe the basic experiment procedure for a fixed True Tree and a fixed set of parameter values, followed by the extended procedure that varies some of the parameter values.

4.1.1. The Basic Procedure. The experiment procedure for a given true tree is as follows:

Step 1. *Obtain the underlying parameter values*: We first generate the true parameters for the underlying user population, including the percentage of liars, the liars' distortion matrices, and the marginal distributions of the attributes for liars and truth-tellers. We then generate a random sample of users based on these parameter values. The sample of users is subsequently used to obtain the estimated distributions and distortion matrices. In this step, we also simulate a misclassification cost matrix and the values in the matrix are again randomly generated.

Step 2. *Generate KM Tree*: The KM tree is constructed for comparison with the methods proposed in the present study. Since the KM method does not differentiate liars from truth-tellers, we construct the KM

Figure 11. True Tree for Telecom Payment Default Risk Assessment



tree using the marginal distributions and distortion matrixes for the entire population, similar to those illustrated in Figures 3(c) and 4(c).

Step 3. *Generate ST, CT, VST, and VCT Trees*: Select an attribute as the verifiable attribute (VA). Follow the steps described in Sections 2 and 3 to build the ST, CT, VST, and VCT trees.

Step 4. *Simulate User's True and Reported Input Vectors*: Based on the marginal distribution of each attribute, we randomly generate a true input vector for each simulated user. This input vector is then fed into the True Tree to obtain the true recommendation, which is used to decide the accuracy of other recommendations. Then, based on the percentage of liars in the population, we randomly determine whether the user is a liar or not. If the user is determined to be a truth-teller, her reported input vector is the same as her true input vector. In case the user is a liar, her reported input values are distorted based on the assumed distortion matrices for liars.

Step 5. *Calculate Accuracy and Misclassification Cost*: Feed each user's reported input vector into the true tree, KM tree, ST tree, CT tree, VST tree, and VCT tree to obtain their respective recommendations. Compare the recommendation obtained by each method with the true one. If they are the same, the recommendation is considered correct; otherwise it is incorrect. For each incorrect recommendation, the corresponding misclassification cost is calculated based on the simulated misclassification cost matrix.

Step 6. *Compare Accuracy and Misclassification Cost*: Repeat Steps 4 and 5 10,000 times to simulate 10,000 users. Record the total number of correct recommendations and the total misclassification cost associated with each method. Compare the accuracies, i.e., the percentages of correct recommendations, and the misclassification costs of different methods.

4.1.2. The Extended Procedure. The basic experiment procedure uses a fixed set of population parameter values, including the liar percentage and distortion matrices. To evaluate the performance of the different methods under varying severity of lying, we control two important parameters, i.e., liar percentage and distortion level for liars in an additional set of experiment runs. The distortion level for an attribute measures the probability that the attribute's value will be distorted by liars, and is calculated based on the marginal distribution and distortion matrix for liars. The distortion levels are kept the same for all attributes in our experiments.

In these experiment runs, we try 11 different liar percentages varying from 0% to 100% and nine different levels of distortion from 0.1 to 0.9. In addition, we randomly choose one attribute as the verifiable attribute (VA) during the extended procedure. For a given combination of liar percentage and distortion level, we perform the basic procedure shown in Section 4.1.1 to compare the performances of the different methods.

4.2. Results

We run the extended procedure based on the true tree built for the telecom payment default risk example, as shown in Figure 11. Based on the data collected from the experiments, we calculate the average accuracy and the total misclassification cost of each method, as summarized in Table 1. We can see that the CT method achieves an average accuracy of 84.18%, the highest among all compared methods. ST ranks second, followed by TT , VCT, KM, and VST. Compared with TT and KM, CT improves the accuracy by 4.91% and 5.31%, respectively. Regarding the misclassification cost, VCT achieves the lowest total cost at $51,395 for 10,000 simulated users. VST ranks second, and then CT, ST, TT, and KM. Compared with TT and KM, VCT delivers a per-user cost saving of $2.52 and $2.79, respectively. The results show that proposed methods can lead to substantial financial benefits for organizations.

To better understand how the performances of the methods are affected by the severity of users' lying behaviors, we plot their performances against two controlled parameters, i.e., the liar percentage and the distortion level.

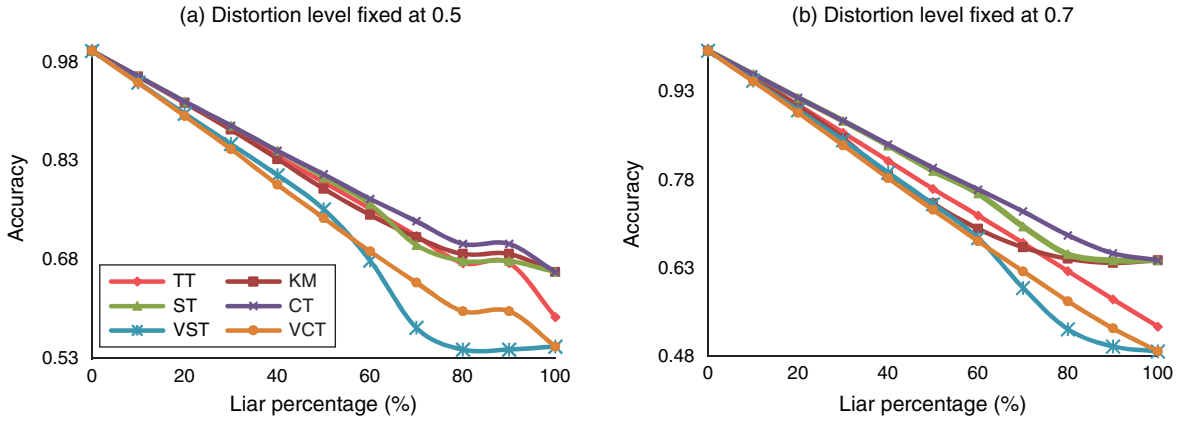
4.2.1. Liar Percentage. Figures 12(a) and 12(b) show the impact of liar percentage on accuracy under two distortion levels (50% and 70%). We find that CT always performs better than ST, KM, TT, VCT, and VST, and ST is the second best in most cases. In both figures, when the liar percentage is low, the performance differences among the different methods are small. Their performance differences start to widen as the liar percentage increases. As the liar percentage approaches 100%, however, the performance differences between CT, ST, and KM again become narrower. When the liar percentage is 100%, the three methods have exactly the same accuracy. This is because both ST and CT rely completely on the liar tree, which is the same as KM tree under this condition. Similarly, VCT and VST achieve the same accuracy when liar percentage is 100% because they both provide the same recommendations as those from the value-based liar tree.

A surprising result observed from the two figures is that the True Tree can outperform the KM tree when

Table 1. Overall Performance Comparison

Method	TT	KM	ST	CT	VST	VCT
Accuracy (%)	79.27	78.87	83.31	84.18	78.22	78.70
Misclassification cost	76,609	79,271	69,820	66,338	53,589	51,395

Figure 12. (Color online) Impact of Liar Percentage on Accuracy



the liar percentage is not very high. The advantage of true tree over KM is more significant when distortion level is between 30% and 70% (more visible in Figure 12(b)). This is a result not observed in the study that develops the KM method (Jiang et al. 2005). We will further examine this interesting phenomenon in Section 4.3.

Figures 13(a) and 13(b) show the impact of liar percentage on the total misclassification cost at two distortion levels (0.5 and 0.7, respectively). We can see that the value-based methods lead to significantly lower misclassification costs than do other methods. Between the two value-based methods, VCT always performs better than, or as well as, VST. The difference is smaller when the liar percentage is close to 100% or 0%. Regarding the other methods, we find that the cost-based performance ordering closely follows the accuracy ordering. For instance, CT has a relatively lower misclassification cost compared to ST, KM, and TT, and KM can lead to a higher misclassification cost than TT unless the liar percentage is very high. In both cases, the lower (higher) misclassification cost is the result of a higher (lower) accuracy of recommendations.

4.2.2. Distortion Level. We next examine the impact of input distortion on the performances of the different methods. Figures 14(a) and 14(b) show its impact on accuracy under two liar percentages (50% and 70%, respectively). We can see that CT still performs best among all compared methods. When distortion level is very low (e.g., 0.1) or very high (e.g., 0.9), the performance difference between CT and ST narrows. We also see in Figure 14(a) that the TT has a clear advantage over KM. At a higher distortion level, KM may lead to a worse accuracy than all other methods.

Further, by comparing Figure 14 with Figure 12, we observe that the accuracies of the four methods proposed in this research (i.e., CT, ST, VCT, and VST) generally decrease with the liar percentage, while they first decrease and then increase as the distortion level changes from very low (e.g., 0.1) to very high (e.g., 0.9). This is because when the distortion level is close to 1.0, there is less certainty about the true attribute values. For instance, given a binary attribute with a distortion level of 0.9, just choosing the value opposite to the reported one can lead to a 90% accuracy. The four proposed methods are sufficiently intelligent

Figure 13. (Color online) Impact of Liar Percentage on Misclassification Cost

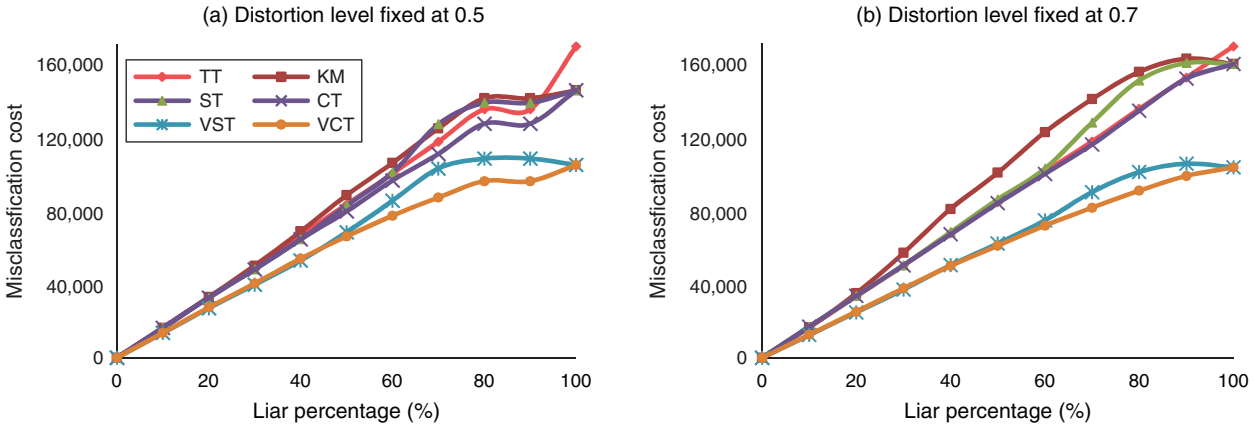
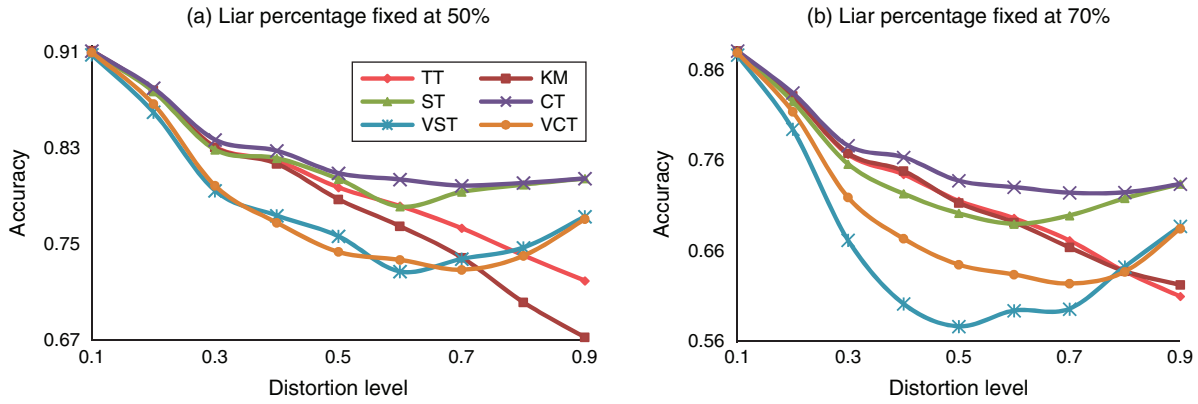


Figure 14. (Color online) Impact of Distortion Level on Accuracy



to incorporate this factor, hence their accuracies can improve as the distortion level approaches 1.0.

The impact of the distortion level on the misclassification costs of the compared methods is shown in Figures 15(a) and 15(b), corresponding to two fixed liar percentages at 50% and 70%, respectively. Again, VCT always has the lowest misclassification cost and VST ranks second. The misclassification costs of the accuracy-based methods again correlate with the accuracies of their recommendations, with CT consistently performing better than ST. The KM method performs the worst when the distortion level is high.

4.3. Why TT May Outperform KM

As pointed out in Section 4.2, KM is frequently outperformed by the True Tree (TT), a result contrasting the finding by Jiang et al. (2005), which shows that KM consistently outperforms the true tree. To understand why KM sometimes may perform worse than TT, we need to examine the composition of the user population that includes both liars and truth-tellers. Note that KM does not differentiate liars from truth-tellers. Instead, the KM method assumes that all users may lie and the probability of lying about each attribute is the same across users.

In reality, there are users who tend to lie and users who rarely or never lie. The two groups of users in a population are illustrated in Figure 16. The first group accounts for 20% of the population, and they lie 100% of the time. The second group accounts for 80% of the population, and all of them are truth-tellers. When the two groups are mixed together, the liar percentage for the entire population is 20%. If the True Tree is adopted for this particular population, although the recommendations for the liars may have a low accuracy, the recommendations for all truth-tellers are guaranteed to be correct. Because the truth-tellers account for 80% of the population, the average accuracy for true tree will be at least 80%. On the other hand, when the KM tree is adopted, since it assumes that everyone is a liar, the recommendations for truth-tellers may not be correct. Even the recommendations for liars may not have a high accuracy rate because the distortion matrices used by the KM, which are calculated for the entire population, are very different from those for the liars. As a result, the accuracy of KM may suffer. Therefore, the true tree can outperform the KM tree under this scenario.

Based on the user composition illustrated in Figure 16, we can imagine that when the liar percentage is

Figure 15. (Color online) Impact of Distortion Level on Misclassification Cost

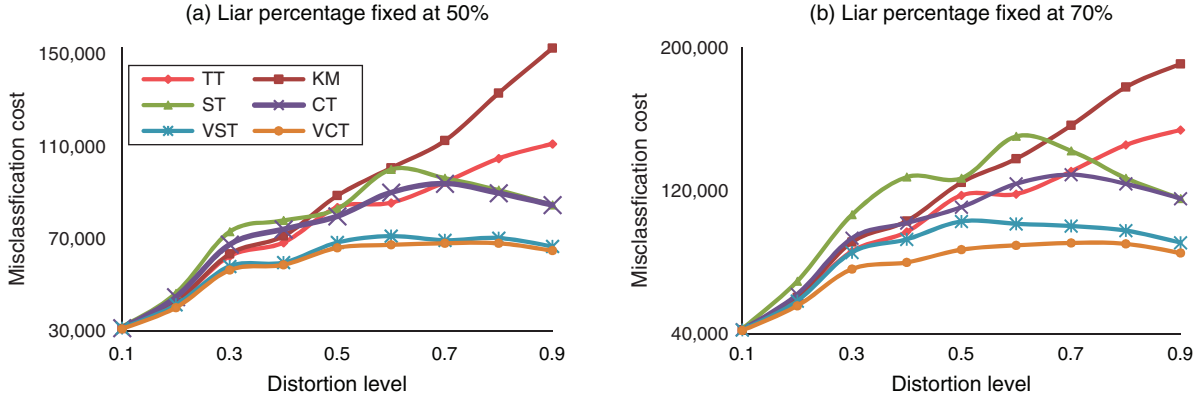
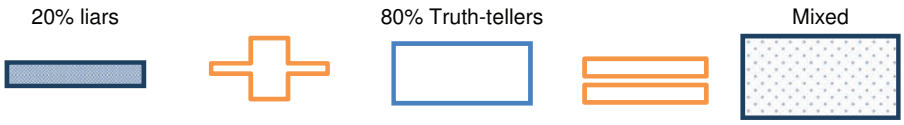


Figure 16. (Color online) Mixing Liars and Truth-Tellers



high, the distortion matrices for the entire population, which are used by KM, will become similar to those for liars. Then, the performance of KM should improve. This is confirmed by Figure 12, which shows that when the liar percentage is above 70%, the KM tree clearly outperforms the true tree. It even matches CT when the liar percentage is close to 100%.

From the experiment results and the above analysis, we conclude that the KM method is recommended only if the user population is relatively homogeneous and the majority of them tend to lie. When liars only account for a relatively small percentage of the population, the methods proposed in this study, CT and VCT in particular, should be used.

4.4. Robustness Tests

Additional simulated experiments are conducted to evaluate the robustness of the proposed methods under various conditions.

4.4.1. Tests on Different VAs. We run additional tests to assess whether the selection of VAs affects the performances of the proposed methods. With a fixed true tree, we change the VA in each run. We find that there is no significant change in performances as the selected VA changes, and the findings presented in Section 4.2 remain qualitatively valid.

4.4.2. Tests on Multiple VAs. An intuitive extension of the proposed methods is to utilize multiple VAs to estimate the probability of a user being a liar. In additional experiments, we try two and three VAs for the ST and CT methods. For expositional convenience, we label the Double-VA extensions of ST and CT by DST and DST, and their Triple-VA extensions by TST and TCT, respectively. The average accuracies for ST, CT, DST, DCT, TST, and TCT are found to be 83.31%, 84.18%, 84.58%, 85.12%, 85.17%, and 85.52%, respectively. Figure 17 depicts how the performances of the different methods change with the two controlled parameters. It shows that including more VAs slightly increases the accuracy, and the CT-based methods consistently outperform the ST-based methods. Our t-tests confirm that the difference between ST-based and CT-based methods is significant regardless of the number of VAs used.

4.4.3. Tests on Distortion Matrices. We are also interested in evaluating whether the characteristics of the distortion matrices affect the performances of the compared methods. For this purpose, we adopt the Kullback–Leibler divergence (Kullback and Leibler 1951) to measure the "amount" of difference between two distortion matrices. The formula for this measure is

$$D_{KL}(P\|Q) = \sum_i P(i) \ln \frac{P(i)}{Q(i)}, \tag{6}$$

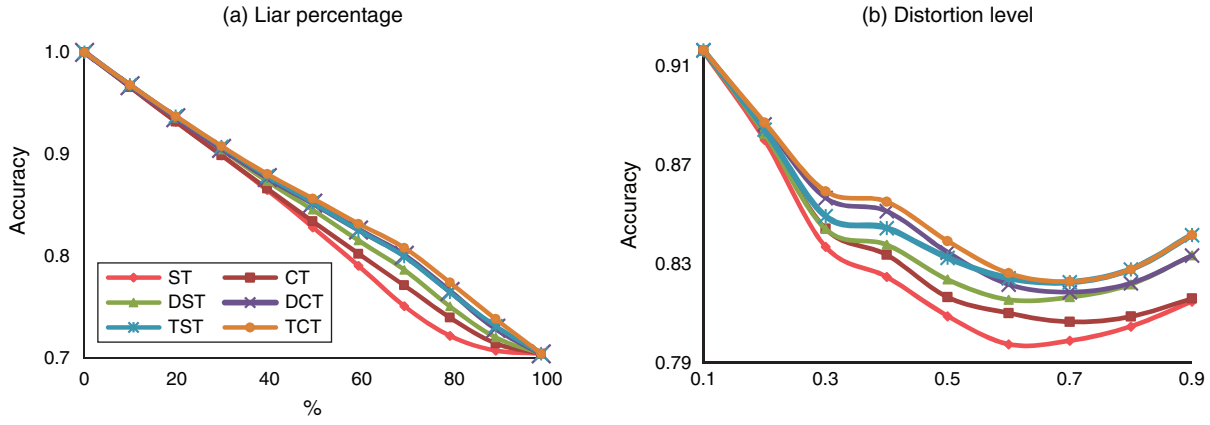
where P, Q are two matrixes and $D_{KL}(P\|Q)$ is the K-L divergence of Q from P. We select identity matrix P as the benchmark matrix. For each distortion matrix, we calculate its K-L divergence from the identity matrix. Then we examine the relationships between the K-L divergence of each distortion matrix and the corresponding performance metrics, including both accuracy and misclassification cost. The results do not suggest a significant correlation between the Kullback–Leibler divergence and the performances of the different methods.

4.4.4. Additional Tests on Simulated Trees. We simulate another 100 True decision tables based on the credit risk assessment example shown in Figure 1(a). In each table, the recommendation corresponding to each input vector is randomly generated. We also simulate 33 true decision tables for the telecom payment default risk example by varying the marginal distributions of the input vectors. The recommendation to each input vector is the same as that from the True Tree shown in Figure 11. For each of the simulated True Trees, we reevaluate the performance of the different methods by trying 11 liar percentages and nine distortion levels. The results are found to be consistent with the findings reported in Section 4.2.

4.4.5. Tests on Random Errors. The goal of this study is to address users' intentional input distortion. In reality, however, incorrect input values could also result from random errors. For instance, a male user, who does not benefit from lying about his gender, could accidently select "Female" from a drop-down list. As a result, the proposed methods may incorrectly classify him as a liar and produce an incorrect recommendation. Because it is typically impossible to separate random errors from intentional lying, instead of extending the methods to handle random errors, we are interested in evaluating the robustness of proposed methods in the presence of random errors.

With possible random errors considered, we again conduct experiments based on both the credit risk assessment and the telecom payment default risk assessment examples. In these experiments, the procedures used to construct the ST, CT, VST, and VCT

Figure 17. (Color online) Multiple Verified Attributes Affect Accuracy



trees remain unchanged. The difference lies in how the "reported" attribute values for each simulated user are generated. Specifically, given a simulated "true" vector, if the user is determined to be a liar, we generate an *intermediate* vector based on the distortion matrices for liars; if she is a truth-teller, the intermediate vector is the "true" vector. We then add random errors to the intermediate vector based on a separate set of *random error matrices* to generate the "reported" vector. We vary the percentage of users producing random errors from 0% to 10%. Because of the random errors, the average accuracies for TT, KM, ST, CT, VST, and VCT drop slightly to 77.93%, 77.31%, 80.59%, 81.48%, 76.89%, and 77.64%, respectively. It is clear that CT and ST remain the two top performers. Therefore, the presence of random errors does not seem to affect the advantage of the proposed methods in relation to benchmark methods.

Based on the prior robustness tests, we conclude that the four proposed methods are superior to existing methods. Among them, the consolidated tree methods (i.e., CT and VCT) are better than the split tree methods (i.e., ST and VST). Therefore, depending on whether the goal is to maximize accuracy or minimize misclassification cost, either CT or VCT should be the method of choice in addressing the challenge of input distortion for rule-based expert systems.

5. Selection of Verifiable Attributes

In this section, we discuss several practical issues related to the selection of verifiable attributes (VAs) to support the four methods proposed in this study (i.e., ST, CT, VST, VCT).

5.1. Determining the Best VA(s)

Verifying the true values of a VA is not cost-free. Some attributes are less costly to verify than others. For instance, verifying a user's bankruptcy status could be much less costly than verifying her income. Therefore, the attribute(s) that leads to the highest *net* benefit, which equals the expected reduction in misclassification cost minus the cost of verification, should be selected as the VA(s) for the proposed methods.

The cost of verification for each selected VA(s) equals the cost of one verification times the probability that the VA(s) will need to be verified at the time of consultation (recall that under CT or VCT, sometimes it is not necessary to verify the true values of VAs). The probability can be obtained by summing up the probabilities associated with all branches of a tree (e.g., a VCT tree) that require the true values of the VA(s). In the credit risk assessment example, suppose the cost of verifying a user's *bankruptcy* status is \$20, and that of verifying *Education* is \$45. When *bankruptcy* is selected as the VA, the probability that it needs to be verified in the VCT tree is 0.35. When *education* is selected as the VA, the probability that it has to be verified is 0.41. Then, the expected costs of verification for the two attributes are, respectively,

$$C_V(\text{Bankruptcy}) = 20 \cdot 0.35 = 7.00 \quad \text{and}$$
$$C_V(\text{Education}) = 45 \cdot 0.41 = 18.45.$$

The misclassification cost associated with each selected VA(s) can be assessed based on either simulated or real performance evaluation. In either case, we need to obtain a *mis-recommendation matrix* for each selected VA(s), which summarizes the probabilities that objects with a given true classification are misclassified into other classifications. For instance, such a matrix can record the percentages of "low risk" customers that are misclassified as "high risk" and "medium risk" by a proposed method. We can use such a mis-recommendation matrix, along with the misclassification cost matrix and the marginal distribution of true classifications, to determine the expected *cost of mis-recommendations* corresponding to each selected VA(s).

To illustrate, suppose the mis-recommendation matrices corresponding to the two VA options are shown in Figures 18(a) and 18(b), and the marginal

Figure 18(a). Mis-Recommendation Matrix Corresponding to Bankruptcy

	HR (%)	MR (%)	LR (%)
HR	85	14	1
MR	23	71	6
LR	39	18	43

Figure 18(b). Mis-Recommendation Matrix Corresponding to Education

	HR (%)	MR (%)	LR (%)
HR	87	9	4
MR	22	73	5
LR	31	6	63

probabilities associated with "HR," "MR," and "LR," are 0.1, 0.3, 0.6, respectively. Using the values in Figures 18(a) and 18(b), the misrepresentation costs are calculated as

$$\begin{aligned} &C_{MR}(Bankruptcy)\\ &\quad=(45\cdot 14\%+100\cdot 1\%)\cdot 0.1+(10\cdot 23\%+50\cdot 6\%)\cdot 0.3\\ &\qquad+(20\cdot 39\%+28\cdot 18\%)\cdot 0.6=10.024,\ \text{and}\\ &C_{MR}(Education)\\ &\quad=(45\cdot 9\%+100\cdot 4\%)\cdot 0.1+(10\cdot 22\%+50\cdot 5\%)\cdot 0.3\\ &\qquad+(20\cdot 31\%+28\cdot 6\%)\cdot 0.6=6.943. \end{aligned}$$

Therefore, the total costs associated with the two VA options are

$$\begin{aligned} C_T(Bankruptcy) &= C_V(Bankruptcy)+C_{MR}(Bankruptcy)\\ &= 17.024 \end{aligned}$$

and

$$\begin{aligned} C_T(Education) &= C_V(Education)+C_{MR}(Education)\\ &= 25.393. \end{aligned}$$

Thus, *bankruptcy* is a better option than *education* as VA. Similarly, the costs associated with other VA(s) can be computed. The VA(s) with the lowest total cost should be selected.

5.2. Selecting an External VA

In some cases, it may be difficult or costly to verify the attributes included in the True Tree. As an alternative, we could use an *external attribute*, which is not included in the True Tree, to estimate the probability that a user is a liar. For instance, in the credit risk assessment example, a user's *Citizenship* status (yes, no) is an external attribute. To use it as a VA, we obtain its distortion matrices and marginal distributions, as illustrated in Figures 19(a) and 19(b). Once a user's reported and true VA values are known, we can calculate the probability that a user is a liar.

Figure 19(a). Distortion Matrices for the External Verified Attribute

Citizenship (C) for liars

	Yes	No
Yes	0.39	0.61
No	0.17	0.83

Citizenship (C) for truth-teller

	Yes	No
Yes	1.00	0.00
No	0.00	1.00

Figure 19(b). Marginal Distribution for the External Verified Attribute

Citizenship (C) for liars

Yes	0.52
No	0.48

Citizenship (C) for truth-teller

Yes	0.6
No	0.4

Note that the structure of split trees is not much affected by using an external or internal VA since the VA is always first verified. For Consolidated Trees, since the reported and true values for an external VA are treated as two additional attributes in the decision table, the tree structure changes. To illustrate, using values shown in Figures 2, 4(a), 4(b), 4(c), 19(a), and 19(b), we obtain a consolidated tree for the credit risk assessment example, as shown in Figure 20. Compared to Figure 7, this tree contains more nodes and layers. Note that in certain branches, we do not need to verify the value of the external VA, which is consistent with CT trees built using internals VAs.

With *citizenship* added as an external VA, we rebuild the 100 simulated decision trees (for credit risk assessment) described in the Section 4.4.4 and rerun the previous performance evaluations. We also reevaluate the performances of the methods after adding an external VA to the decision trees built from the telecom payment default risk example. The results are consistent between the two sets of experiments. Figure 21 shows how the performances of the compared methods change with the liar percentage and distortion level for the telecom payment default risk example. The average accuracies for TT, KM, ST, CT, VST, and VCT are found to be 80.32%, 79.63%, 83.31%, 84.91%, 78.98%, and 79.45%, respectively. The performance ordering is consistent with those obtained when internal VAs are used, which indicates that the adoption of external VAs does not affect the effectiveness of the proposed methods. This fact provides us more options when selecting the VAs for the proposed methods. It is even possible to adopt a combination of internal and external attributes. These options can help lower the barrier for implementing the proposed methods and potentially reduce the verification cost.

5.3. Dealing with Strategic Agents

Once the proposed methods are deployed, it is possible that after repeated tries, a strategic agent can find out which attribute is used as a VA. There are possible countermeasures to deal with such a strategic behavior. The first option is to change the VA from time to

Figure 20. Consolidated Tree with External Verified Attribute

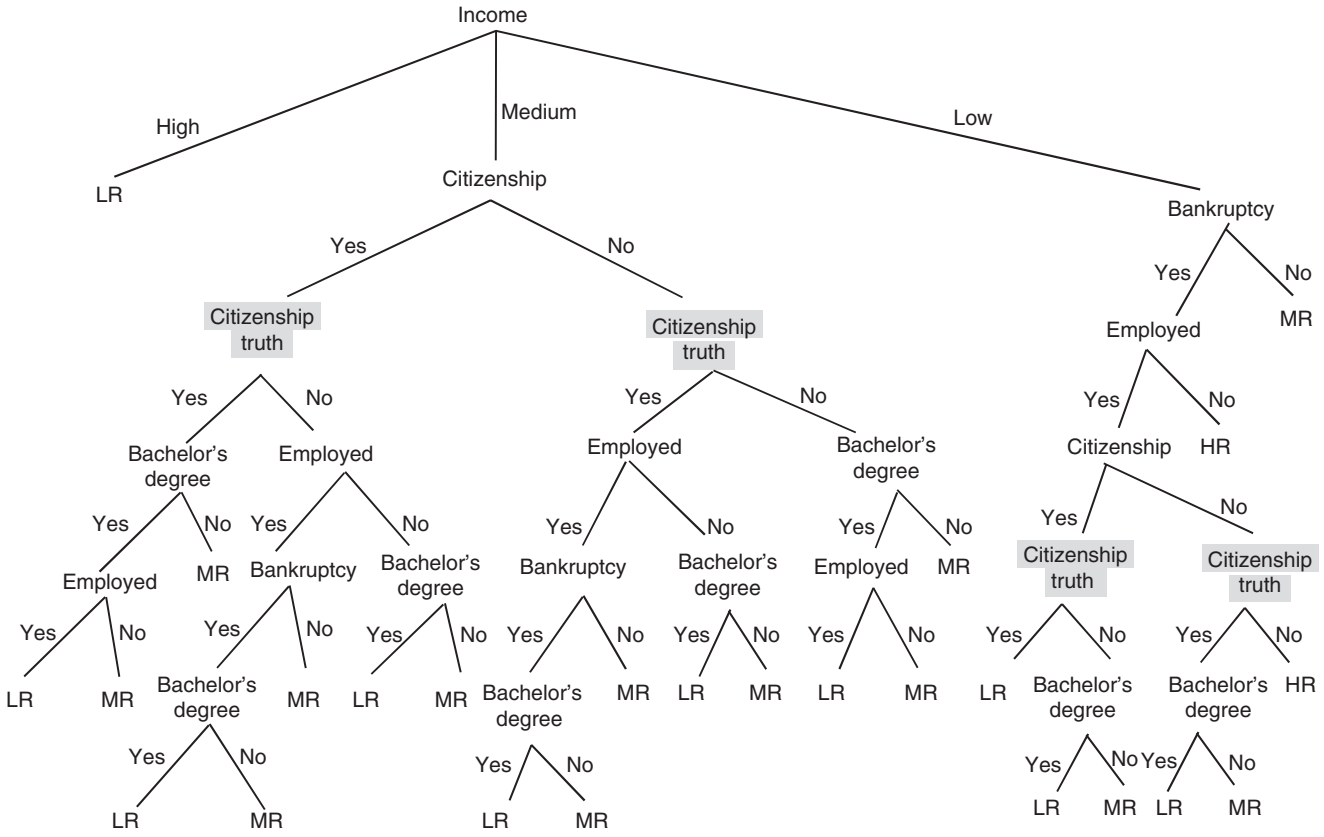
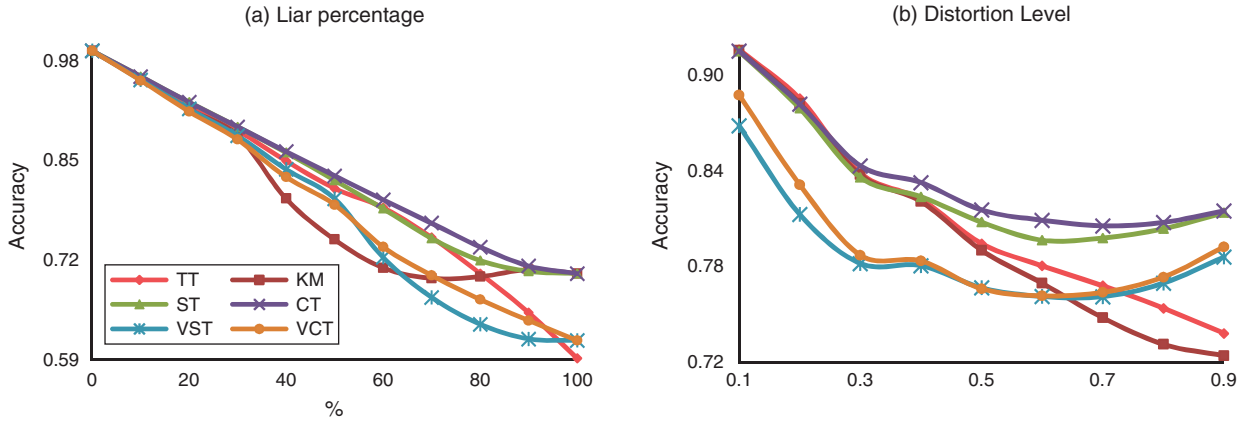


Figure 21. (Color online) External Verified Attributes Affect Accuracy



time. The second option is to randomize the selection of VAs at runtime. We have built and tested such variation methods and found that the previous findings regarding performance ordering remain valid.

6. Extension with Multiple User Groups

So far, the proposed methods consider only two groups of users, i.e., liars and truth-tellers. Under certain settings, such a two-group classification may not be sufficient to capture the heterogeneity of potential users. For instance, in a given user population, some never lie, some lie occasionally, while others lie frequently. Users could also exhibit different levels of strategic lying behaviors. For instance, some may only distort the attributes that are more likely to affect the outcome, while others may be more casual in attribute selection. Upon further examination, we conclude that as long as the different groups of users can be identified from a sample (possibly with the help of behavioral experts or through deeper analysis of historical data),

the proposed methods can be extended to deal with a more heterogeneous user population. We next demonstrate how this can be done using a three-group case including frequent liars, occasional liars, and truth-tellers.

Under the three-group case, we need to estimate the probability of a user being a frequent liar and that of a user being an occasional liar. Such probabilities can again be calculated based on the Bayes' theorem. The difference from the two-group case is that we need to have one more set of distortion matrices and marginal distributions, which can be obtained based on data collected from the three groups of users during sampling.

We first develop the accuracy-based three-group methods. The Split Tree has three branches, i.e., *frequent liar tree* (FLT), *occasional liar tree* (OLT), and true tree. The construction of the FLT and OLT branches is similar to that of the liar tree, with the exception that the frequent and occasional liars' distortion matrices and marginal distributions are used, instead of those for all liars. At the time of system consultation, the branch with the highest probability for a given user will be traversed. The construction of the three-group consolidated tree is also similar to that of the two-group one, with the exception that the three-group method combines the FLT, OLT, and TT path probabilities to determine the CT recommendation. Analogous to the extension from the two-group accuracy-based methods to the two-group value-based methods, we also extend three-group accuracy-based methods to the three-group value-based methods.

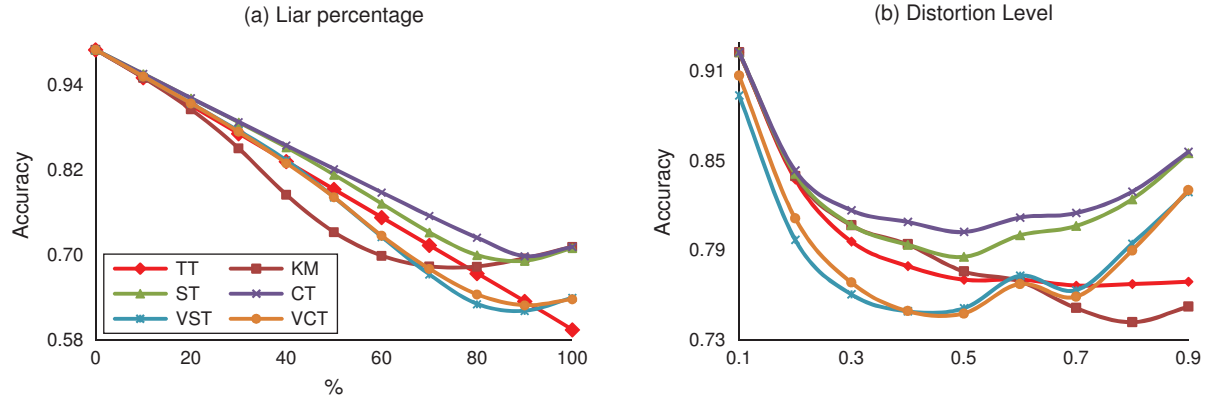
With the three-group methods developed, we rerun the previous experiments based on the simulated decision trees for credit risk assessment and the decision trees for telecom payment default risk assessment. The results are again consistent with those obtained from the experiments for the two-group methods. Figure 22 shows how the accuracy of the different methods changes with the liar percentage and distortion level for the telecom payment default risk example. We can see that CT and ST consistently rank the first and second in terms of accuracy. In additional experiments, we use nine different ratios (from 1 to 9) of the number of occasional liars to the number of frequent liars. In each run, we randomly select one attribute in the true tree as the VA. The average accuracies for TT, KM, ST, CT, VST, and VCT are found to be 79.80%, 79.51%, 82.66%, 83.44%, 79.04%, and 79.26%, respectively. The ordering of accuracy is exactly the same as that shown in Table 1. We also evaluate the performance of the three-group methods measured by the total misclassification cost. The results are again consistent with the findings reported in Section 4.2 for the two-group case. We thus conclude that the proposed methods remain superior even if a more heterogeneous user population is taken into account.

7. Practical Applications of the Proposed Methods

Although we use credit risk assessment as the motivating example of this study, the proposed methods should be applicable in other domains as well, provided that the following conditions are met: (1) the decision rules used to construct an expert system are provided by experts or learned from reliable historical data, and the data fed into the expert systems has much lower reliability than that assumed by the system; (2) users providing inputs to the expert system are heterogeneous in their propensity to lie about the inputs; (3) each user group's distortion matrices and marginal distributions can be separately estimated from historical or sampling data; and (4) misclassification costs are obtainable if the VCT method is adopted. Note that the above conditions also summarize the key data required in order to implement the proposed methods.

Based on the theoretical analyses and experimental results presented in the previous sections, we summarize a few practical guidelines on the implementation of the proposed methods. First, if misclassification costs

Figure 22. (Color online) Accuracy of Three-Group Methods



are asymmetric, the cost-minimizing VCT method is preferable. In case such costs are approximately symmetric, then the accuracy-maximizing CT method is a good alternative. Second, selecting the appropriate VA(s) is the most critical step in a successful implementation of the proposed methods. In general, the selected VA(s) should maximize the net benefit, which equals the expected reduction in misclassification cost minus the cost of verification. The selection of VA(s) should not be limited to a single VA, or only the internal VAs. In the presence of strategic agents, randomizing the selection of VAs in real time should be considered.

8. Conclusion

Despite the prevalence of input distortion by users of rule-based expert systems, research on how to effectively address such challenges has been limited. The methods proposed in this study explicitly differentiate liars from truth-tellers and treat them differently when their information is fed into a redesigned rule-based expert system. Two of the proposed methods (i.e., split tree [ST] and consolidated tree [CT]) attempt to improve the accuracy of recommendations, and the other two (i.e., value-based split tree [VST] and value-based consolidated tree [VCT]) aim to minimize the expected misclassification cost resulting from incorrect recommendations. Experimental results show that the proposed methods can lead to significantly better accuracy or lower cost than existing methods. Between the two pairs of proposed methods, we find that CT consistently outperforms ST in improving the accuracy of recommendations, and VCT always performs better than VST in reducing the expected misclassification cost.

The methodologies and findings of this study have important financial implications. Although the proposed methods require the verification of user-provided attribute values, the cost of such validation can be controlled by selecting the attributes that are relatively easy to verify. As a result, we expect the benefit of adopting the proposed methods to exceed the cost under most real-world applications. Given the wide application of rule-based expert systems in various problem domains, the proposed methods can potentially lead to significant financial saving for organizations.

One limitation of the proposed methods is that they are computationally intensive. In a future study, one could simplify the methods to reduce their complexity. For instance, when computing the CT table, it is possible that the accuracy may not degrade much even if we consider only a small subset of the possible true vectors given a reported vector. The computation time can be significantly reduced if a subset of vectors can be identified and used in constructing the trees.

Endnote

[1] An example of such data is available at http://en.wikipedia.org/wiki/Credit_card_interest.

References

Abbasi A, Albrecht CC, Vance A, Hansen JV (2012) MetaFraud: A meta-learning framework for detecting financial fraud. *MIS Quart.* 36(4):1293–1327.

Abbasi A, Zhang Z, Zimbra D, Chen H, Nunamaker JF (2010) Detecting fake websites: The contribution of statistical learning theory. *MIS Quart.* 34(3):435–461.

Baesens B, Setiono R, Mues C, Vanthienen J (2003) Using neural network rule extraction and decision tables for credit-risk evaluation. *Management Sci.* 49(3):312–329.

Boylu F, Aytug H, Koehler GJ (2010) Induction over strategic agents. *Inform. Systems Res.* 21(1):170–189.

Buller DB, Burgoon JK (1996) Interpersonal deception theory. *Comm. Theory* 6(3):203–242.

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

Duan Y, Edwards JS, Xu MX (2005) Web-based expert systems: Benefits and challenges. *Inform. Management* 42(6):799–811.

Fayyad UM, Irani KB (1993) Multi-interval discretization of continuous-valued attributes for classification learning. *Proc. 13th Internat. Joint Conf. Artificial Intelligence, Chambéry, France*, 1022–1029.

George JF, Biros DP, Adkins MJ, Burgoon JK, Nunameker JF (2004) Testing various modes of computer-based training for deception detection. *Internat. Conf. Intell. Security Inform.* (Springer, Berlin), 411–417.

Hoffman DL, Novak TP, Peralta M (1999) Building consumer trust online. *Comm. ACM* 42(4):80–85.

Jiang Z, Mookerjee VS, Sarkar S (2005) Lying on the Web: Implications for expert systems redesign. *Inform. Systems Res.* 16(2): 131–148.

Keefer A (2015) Penalties for a falsified credit application. Accessed February 9, 2017, http://www.livestrong.com/article/117652-penalties-falsified-credit-application.

Kullback S, Leibler RA (1951) On information and sufficiency. *Ann. Math. Statistics* 22(1):79–86.

Liao H (2005) Expert system methodologies and applications—A decade review from 1995 to 2004. *Expert Systems Appl.* 28(1): 93–103.

Metzger MJ (2004) Privacy, trust, and disclosure: Exploring barriers to electronic commerce. *J. Comput.-Mediated Comm.* 9(4).

Power DJ (2000) Web-based and model-driven decision support systems: Concepts and issues. *Proc. Americas Conf. Inform. Systems, Long Beach, CA*, 352–355.

Rubin D (2004) *Multiple Imputations for Nonresponse in Surveys* (Wiley, New York).

Zhou L, Zhang D (2008) Following linguistic footprints: Automatic deception detection in online communication. *Comm. ACM* 51(9):19–22.

Zhou L, Shi Y, Zhang D (2008) A statistical language modeling approach to online deception detection. *IEEE Trans. Knowledge Data Engrg.* 20(8):1077–1081.

Zhou L, Twitchell D, Qin T, Burgoon J, Nunamaker J (2003) An exploratory study into deception detection in text-based computer-mediated communication. *Proc. 36th Hawaii Internat. Conf. Systems Sciences, Big Island, HI.*

Zhu X, Wu X, Chen Q (2003) Eliminating class noise in large data sets. *Proc. Internat. Conf. Machine Learning, Washington, DC*, 920–927.