

Rule induction for uncertain data

Biao Qin · Yuni Xia · Sunil Prabhakar

Received: 23 October 2009 / Revised: 11 July 2010 / Accepted: 17 July 2010
© Springer-Verlag London Limited 2010

Abstract Data uncertainty are common in real-world applications and it can be caused by many factors such as imprecise measurements, network latency, outdated sources and sampling errors. When mining knowledge from these applications, data uncertainty need to be handled with caution. Otherwise, unreliable or even wrong mining results would be obtained. In this paper, we propose a rule induction algorithm, called uRule, to learn rules from uncertain data. The key problem in learning rules is to efficiently identify the optimal cut points from training data. For uncertain numerical data, we propose an optimization mechanism which merges adjacent bins that have equal classifying class distribution and prove its soundness. For the uncertain categorical data, we also propose a new method to select cut points based on possible world semantics. We then present the uRule algorithm in detail. Our experimental results show that the uRule algorithm can generate rules from uncertain numerical data with potentially higher accuracies, and the proposed optimization method is effective in the cut point selection for both certain and uncertain numerical data. Furthermore, uRule has quite stable performance when mining uncertain categorical data.

Keywords Rule · Uncertain data · Cut point · Possible world semantics

B. Qin (✉)
Department of Computer Science, Renmin University of China, Beijing, China
e-mail: biaoqin.cs@hotmail.com

B. Qin
Key Labs of Data Engineering and Knowledge Engineering, MOE, Beijing, China

Y. Xia
Department of Computer and Information Science, Indiana University-Purdue University,
Indianapolis, IN, USA
e-mail: yxia@cs.iupui.edu

S. Prabhakar
Department of Computer Science, Purdue University, West Lafayette, IN, USA
e-mail: sunil@cs.purdue.edu

1 Introduction

In many applications, data contain inherent uncertainty. A number of factors contribute to the uncertainty, such as the random nature of the physical data generation and collection process, measurement and decision errors, unreliable data transmission and data staling. For example, in location-based services, moving objects of interest are attached with locators, and this location information is periodically updated and streamed to the control center. Based on this location information, many location-based services can be provided including real-time traffic-based routing, public transportation planning and accident prediction. In these types of applications, location data are typically inaccurate due to locator energy and precision constraint, network bandwidth constraint and latency. Similarly, uncertainty is a fundamental property of sensor networks, software engineering, web design, e-commerce, robotics and many others areas [31].

Uncertainty can also arise in categorical data. For example, a tumor is typically classified as benign or malignant in cancer diagnosis treatment. In practice, it is often very difficult to accurately classify a tumor at the initial diagnosis due to the experiment precision limitation. Inevitably, the lab results are sometimes false positive or false negative. Therefore, doctors may often diagnose tumors to be benign or malignant with certain probability or confidence [6]. Another example of uncertain categorical data can be found in protein databases. An important feature of protein is whether it is ordered, that is, whether it has a secondary structure. Since it is extremely expensive and time-consuming to determine the existence of secondary structure by experiments, this type of information is typically obtained by literature mining—examining the experiments, or the features of similar or closely related protein databases. However, the literature mining results tend to be highly uncertain, and a protein may be taken as ordered (or unordered) with certain confidence [33].

Since data uncertainty are ubiquitous, it is important to develop data mining algorithms for uncertain data sets. When mining knowledge from these applications, data uncertainty need to be handled with caution. Otherwise, unreliable or even wrong mining results would be obtained. This paper focuses on rule-based classification algorithms, which have a number of desirable properties. Rule sets are relatively easy for people to understand, and rule learning systems outperform decision tree learners on many problems [36]. Rule sets have a natural and familiar first-order version, namely Prolog predicates, and techniques for learning propositional rule sets can often be extended to the first-order case [30]. RIPPER [11] is considered to be one of the most commonly used rule-based algorithms for mining certain data sets in practice. However, when data contain uncertainty—for example, when some numerical data are, instead of precise value, an interval with probability distribution function in that interval [9] and some categorical data are *x-tuple* [37]—these algorithms can not process the uncertain data properly. Based on RIPPER, we propose a new rule induction algorithm, called uRule, to learn rules from uncertain data. The main contributions of this paper are as follows:

- We incorporate the data uncertainty model into the rule induction process and propose the uRule algorithm, which can effectively learn rules from uncertain data. We also extend the rule-based prediction process considering rule partial coverage.
- For numerical and uncertain numerical attributes, we optimize the cut point selection by merging adjacent bins that have equal proportion of classifying class distribution. We prove that this method is sound and efficient.
- For uncertain categorical attributes, we propose a new method to select cut points based on possible world semantics.

- 63 – We perform extensive experiments on uRule. Experiments show that by exploiting uncer-
 64 tainty, uRule can generate rules from uncertain numerical data with potentially higher
 65 accuracies, and uRule is also stable for mining uncertain categorical data.

66 This paper is organized as follows. In the next section, we will introduce basic concepts
 67 of rule-based classifiers. Section 3 discusses the optimized process of selecting cut point for
 68 both certain and uncertain numerical attributes. Section 4 describes the cut point selection
 69 for uncertain categorical attributes. Section 5 illustrates uRule in details. The experimental
 70 results are shown in Sect. 6. Section 7 discusses related work on data mining with uncertainty
 71 and Sect. 8 concludes the paper.

72 2 Rule-based classification background

73 We will first briefly introduce the concept of rule classifier. Rule-based classifiers generate a
 74 classification model represented as a set of IF-THEN rules. An IF-THEN rule is an expres-
 75 sion of the form IF condition THEN conclusion, for example, “IF Home Owner = No AND
 76 Annul Income < 30,000 THEN Defaulted Borrower = Yes”. The IF-part (or left-hand side)
 77 of a rule is known as the rule antecedent or precondition. The THEN-part (or right-hand side)
 78 is the rule consequent. In the rule antecedent, the condition consists of one or more attribute
 79 tests (such as Home Owner = No and Annul Income < 30,000) that are logically ANDed.
 80 The rule’s consequent contains a class prediction. In this case, we are predicting whether a
 81 customer will default a loan. If the condition in a rule antecedent holds true for a given tuple,
 82 we say that the rule antecedent is satisfied and that the rule covers the tuple. A rule set can
 83 consist of multiple rules $r_s = \{r_1, r_2, \dots, r_n\}$. A rule r covers a tuple T_j if the attributes of
 84 the tuple satisfy the condition of the rule. The *coverage* of a rule is the number of tuples that
 85 satisfies the antecedent of a rule. The *accuracy* of a rule is the fraction of tuples that satisfy
 86 both the antecedent and consequent of a rule. Ideal rules should have both high coverage and
 87 high accurate rates.

88 uRule employs the sequential covering algorithm, which is commonly used in rule-based
 89 classifier for generate rules from data sets. Sequential covering algorithm extracts the rules
 90 one class at a time for data sets. Let (C_1, C_2, \dots, C_m) be the ordered classes according to
 91 their frequencies, where C_1 is the least frequent class and C_m is the most frequent class. Here,
 92 frequency means the number of tuples a class contains. During the k th iteration, tuples that
 93 belong to C_k are labeled as positive tuples, while those that belong to other classes are labeled
 94 as negative tuples. The sequential covering method is used to generate rules that discrimi-
 95 nate between the positive and negative tuples. After a rule is extracted, the algorithm must
 96 eliminate all the positive and negative tuples covered by the rule. This process is repeated
 97 until only C_m is left.

98 Like traditional rule-based classifier, uRule employs a general-to-specific strategy to grow
 99 a rule. We start with an empty rule and then gradually keep appending attribute tests to it.
 100 We append by adding the attribute test as a logical conjunct to the existing condition of the
 101 rule antecedent. For example, for the default loan data below, we start with the most general
 102 rule, that is, the condition of the rule antecedent is empty. The rule is IF THEN default
 103 Loan = No. We then consider each possible attribute test that may be added to the rule. These
 104 can be derived from the parameter Att-vals, which contains a list of attributes with their
 105 associated values. For example, for an attribute-value pair (att; val), we can consider attribute
 106 tests such as att = val, att < val, att \geq val and so on. Typically, the training data will contain
 107 many attributes, each of which may have several possible values. Finding an optimal rule set

108 becomes computationally explosive. uRule adopts a greedy depth-first strategy. Each time it
 109 is faced with adding a new attribute test (conjunct) to the current rule. Based on the training
 110 samples, it picks the one that most improves the rule quality. It extends FOIL's information
 111 gain measure [29] to choose a conjunct to be added into the rule antecedent.

112 The FOIL's information gain is defined as: $\text{Gain}(r_0, r_1) := t \times \left(\log \left(\frac{p_1}{p_1+n_1} \right) - \right.$
 113 $\left. \log \left(\frac{p_0}{p_0+n_0} \right) \right)$. Here, r_0 denotes a rule before adding a new conjunct and r_1 is an extension
 114 of r_0 after adding the new conjunct. p_0 and n_0 denote the number of positive tuples and
 115 negative tuples covered by r_0 , respectively; p_1 and n_1 denote the number of positive tuples
 116 and negative tuples covered by r_1 , respectively. t is the number of positive tuples, covered by
 117 r_0 as well as by r_1 . The best conjunct that brings the highest FOIL's information gain measure
 118 will be added into the rule antecedent. The new rule is then pruned based on its performance
 119 on the validation set. uRule also performs additional optimization steps to determine whether
 120 some of the existing rules in the rule set can be replaced by better alternative rules. In practice,
 121 the FOIL's information gain is implemented differently in various data mining tools, for
 122 example, WEKA [38] uses a variation of the FOIL's information gain in which t is replaced
 123 with p_1 .

124 3 Cut point selection in uncertain numerical data

125 While the rules are being generated from a certain training data set, the goal is to determine
 126 the cut point that best divides the data sets. There are many metrics that can be used to
 127 find optimal cut points. uRule determines the optimal cut points based on the extended FOIL
 128 information gain measures. We first give the uncertain numerical data model and then discuss
 129 the optimal cut point selection. In this paper, we focus on the uncertainty in attributes and
 130 assume the class type is certain.

131 3.1 A model for uncertain numerical data

132 When the value of a numerical attribute is uncertain, the attribute is called an uncertain
 133 numerical attribute (UNA) [9]. Further, we use A_j to denote the j th instance of A . The value
 134 of A is represented as a range or interval and the probability distribution function (PDF) over
 135 this range [35]. Note that A is treated as a continuous random variable. The PDF $f(x)$ can be
 136 related to an attribute if all instances have the same distribution, or related to each instance
 137 if each instance has different distributions.

138 Table 1 shows an example of UNA. The data in this table are used to predict whether
 139 borrowers will default on loan payments. Among all the attributes, the Annual Income is a
 140 UNA, whose precise value is not available. We only know the range of the Annual Income
 141 of each person and the PDF $f(x)$ over that range. The probability distribution function of
 142 the UNA attribute Annual Income is assumed to be normal distribution and PDF $f(x)$
 143 is not shown in Table 1. For an uncertain numerical attribute value $[A_j.a, A_j.b]$, μ_j can be
 144 estimated as $(a + b)/2$ assuming the error is unbiased; and σ_j can be estimated as $(b - a)/6$
 145 because for a normal random variable, the probability that a value falls within the mean plus
 146 minus 3 standard deviation is more than 99% [18].

147 **Definition 1** An uncertain interval of A_j , denoted by $A_j.U$, is an interval $[A_j.a, A_j.b]$
 148 where $A_j.b \geq A_j.a$. Random variable A_j only allows to have values inside the interval, that
 149 is, $P(A_j \in U) = 1$.

Table 1 Training set for predicting borrowers who will default on loan payments

Rowid	Home owner	Marital status	Annual income	Defaulted borrower
1	No	Single	50–60	No
2	No	Single	10–25	Yes
3	Yes	Married	50–85	No
4	Yes	Divorced	50–60	No
5	No	Married	10–30	Yes
6	Yes	Divorced	110–150	No
7	Yes	Single	25–40	No
8	Yes	Married	20–40	No
9	No	Single	60–85	No
10	No	Divorced	40–55	No
11	No	Divorced	20–35	Yes
12	Yes	Married	20–30	Yes

150 3.2 Measures for selecting the best cut point

151 In papers [14, 15], Elomaa and Rousu proposed a framework to accelerate the cut point selection.
 152 We extend it to handle uncertain numerical data. As described earlier, the value of an
 153 uncertain numeric attribute is an interval with associated PDF. Each uncertain interval has a
 154 maximal value and a minimal value, which are called critical points. We can order all critical
 155 points of an uncertain numeric attribute in an ascending sort with duplicate elimination. An
 156 interval between two adjacent critical points for the attribute under consideration is called
 157 a bin. If the interval between two critical points converges to a point, the bin has the same
 158 meaning as given in [14, 15]. So, the definition of bin in this paper is an extension of the
 159 original definition. Suppose there are N critical points after eliminating duplicates, then this
 160 UNA can be divided into $N + 1$ bins. Since the leftmost and rightmost bins do not contain
 161 any tuple at all, a partition definitely will not occur within them. We need only consider the
 162 remaining $N - 1$ bins.

163 Assume a data set D can be divided into N bins by UNA A . One bin may overlap many
 164 instances of the UNA A . For example, the bin $[20, 25)$ of the Annual Income attribute overlaps
 165 4 tuples, namely T_2, T_5, T_8 and T_{12} . Among them, T_8 is in the class No, while T_2, T_5 and
 166 T_{12} are in class Yes. When a tuple T_i with UNA overlaps a bin $[a, b)$, the probability of the
 167 attribute instance that actually falls in that bin is $P(T_i \in [a, b))$. For example, the probability
 168 that T_2 falls in the bin $[20, 25)$ is $P(T_2 \in [20, 25))$. Because its PDF is a normal distribution,
 169 the probability is $P(T_2 \in [20, 25)) = \phi((20 - 17.5)/2.5) - \phi((25 - 17.5)/2.5) = 0.1573$.
 170 Based on the probability of each individual tuple falling in a bin $[a, b)$, we can compute the
 171 cardinality of a data set falling in that bin.

172 **Definition 2** The Bin Cardinality of the data set over a bin $Bi = [a, b)$ is the sum of the
 173 probabilities of each tuple whose corresponding UNA falls in $[a, b)$. That is, $BC(Bi) =$
 174 $\sum_{j=1}^n P(A_j \in [a, b))$.

175 $BC(Bi)$ is the expected number of instances of A occurring in Bi . Naturally, this may be
 176 greater than 1. Refer to the data set shown in Table 1, the bin cardinality of the bin $[20, 25)$
 177 of the Annual Income is the sum of the probabilities of tuples with Annual Income falling in

Table 2 Grow data for rule induction (This table is used as toy example in the text)

Rowid	Home owner	Marital status	Annual income	Defaulted borrower
2	No	Single	10–25	Yes
4	Yes	Divorced	50–60	No
5	No	Married	10–30	Yes
6	Yes	Divorced	110–150	No
7	Yes	Single	25–40	No
8	Yes	Married	20–40	No
10	No	Divorced	40–55	No
12	Yes	Married	20–30	Yes

Table 3 Bins and their CDVs (This tables is also used as toy example in the text)

	[10, 20)	[20, 25)	[25, 30)	[30, 40)	[40, 50)
CDV(Bi, C)	(1.3386, 0)	(1.0905, 0.0654)	(0.5641, 0.5905)	(0, 1.3386)	(0, 0.84)
	[50, 55)	[55, 60)	[60, 110)	[110, 150)	
CDV(Bi, C)	(0, 0.656)	(0, 0.4986)	(0, 0)	(0, 0.9973)	

178 [20, 25). T_2, T_5, T_8 and T_{12} overlap bin [20, 25), and the probability for T_2 with Annual Income
 179 in [20, 25) is $P(T_2 \in [20, 25)) = 0.1573$. Similarly, $P(T_5 \in [20, 25)) = 0.4332$, $P(T_8 \in$
 180 $[20, 25)) = 0.0655$ and $P(T_{12} \in [20, 25)) = 0.5$. Therefore, the bin cardinality of this data
 181 set over bin [20, 25) is 1.156.

182 **Definition 3** The bin cardinality for class C_k of the data set over a bin $Bi = [a, b)$ is
 183 the sum of the probability of each tuple T_j in class C_k whose corresponding UNA falls in
 184 $[a, b)$. That is, $BC(Bi, C_k) = \sum_{k=1}^n P(A_j \in [a, b) \wedge C_{T_j} = C_k)$, where C_{T_j} denotes the
 185 class label of T_j .

186 **Definition 4** The class distributions of each bin can be denoted by a vector as follows:

$$187 \quad CDV(Bi, C) = (BC(Bi, C_1), BC(Bi, C_2), \dots, BC(Bi, C_m))$$

188 and we call it a Class Distribution Vector (CDV) of the bin Bi .

189 Let us continue on the previous example. The bin cardinality for class Default Borrower =
 190 No over the bin [20, 25) of Annual Income is the sum of the probabilities of tuples who
 191 are not a Default Borrower with Annual Income falling in [20, 25). Among T_2, T_5, T_8 and
 192 T_{12} who overlap [20,25), only T_8 is in class No. Therefore, the bin cardinality for Default
 193 Borrower = No over bin [20, 25) is 0.0655. Similarly, the bin cardinality for Default Bor-
 194 rower = Yes over bin [20, 25) is 1.0905. So $CDV(Bi \in [20, 25), C) = (1.0905, 0.0655)$.

195 *Example 1* We perform a 3-fold stratified holdout, taking out 2/3 the data by random sam-
 196 pling. We use those data to build rules and use the remaining 1/3 data for pruning. The grow
 197 data for Table 1 are shown in Table 2. The CDVs for each bin are shown in Table 3.

198 If the CDV of a bin $Bi = [a, b)$ is $(0, 0, \dots, 0)$, then no instance falls in that bin. Thus,
 199 the bin can be combined with its adjacent bins. For example, the data set in Table 2 has 9 bins

for the Annual Income attribute. There is a bin $Bi = [60, 110)$ with $CDV(Bi, C) = (0, 0)$. Then, this bin can be combined with bin $Bi = [55, 60)$ and a new bin $Bi = [55, 110)$ is formed.

For numerical data, every point is covered by exactly one rule. A bin is also covered by exactly one rule for uncertain numerical data. However, an uncertain tuple may be covered either fully or partially, by more than one rule. For example, a rule “if Annul Income < 30 then ...”partially covers T_7, T_8 and T_{11} . The following definitions are for the cut point, cut direction and the degree of an instance covered by a rule.

Definition 5 Suppose a rule involves a cut point value v and a direction x (either $<$ or \geq) for an uncertain numerical attribute A . Let the interval of the uncertain numerical attribute instance is $[a, b]$. If the instance is covered by the rule, then it must be one of the following scenarios:

- $a < v < b$. That means the cut point is between the uncertain interval of the instance. In this case, this instance will be partially covered by this rule.
- the cut direction x is $<$ and $b \leq v$. This indicates the cut point v is larger than the interval and the antecedent is of the form attribute $< v$. Therefore, the instance is fully covered by this rule.
- the cut direction x is \geq and $a \geq v$. This indicates the cut point v is smaller than the interval and the antecedent is of the form attribute $\geq v$. Therefore, the instance is also fully covered by this rule.

We call such a threshold value, v , a cut point and the direction, x , a cut direction.

Suppose the rule $r_i : A_1 \rightarrow +$ covers p_0 positive bins and n_0 negative bins, then the bin cardinality of positive bins is $BC(p_0) = \sum_{j=1}^{|p_0|} BC(Bi_j)$ and the bin cardinality of negative bins is $BC(n_0) = \sum_{j=1}^{|n_0|} BC(Bi_j)$. Suppose after adding a new conjunct A_2 , the extended rule $r_j : A_1 \wedge A_2 \rightarrow +$ now covers p_1 positive bins with bin cardinality $BC(p_1) = \sum_{j=1}^{|p_1|} BC(Bi_j)$ and n_1 negative bins with bin cardinality $BC(n_1) = \sum_{j=1}^{|n_1|} BC(Bi_j)$. Then, the extended FOIL information gain of adding the conjunct A_2 is defined as following:

Definition 6 Assume that r_j is a rule by adding a new conjunct to r_i , r_i covers p_0 positive bins and n_0 negative bins, and r_j covers p_1 positive bins and n_1 negative bins. Let $y_1 = \log_2 \frac{BC(p_1)}{BC(p_1)+BC(n_1)}$ and $y_0 = \log_2 \frac{BC(p_0)}{BC(p_0)+BC(n_0)}$, then the *extended FOIL’s Information Gain* for a data set D is

$$\text{Gain}(r_i, r_j) = BC(p_1) \times (y_1 - y_0). \tag{1}$$

Since the measure is proportional to $BC(p_1)$ and $BC(p_1)/(BC(p_1) + BC(n_1))$, it prefers rules that have both a high support count (coverage) and accuracy. The extended FOIL’s information gain will be used in the process of rule generation. It can determine the best cut point for an uncertain numerical attribute, and it can be applied to certain numerical attributes as well.

Theorem 1 *The traditional FOIL’s information gain [11] is actually a special case of the extended FOIL’s information gain.*

239 *Proof* Because $BC(p) = p$ and $BC(n) = n$ in a certain data set, the equations are as follows.

$$\begin{aligned}
 240 \quad y_1 &= \log_2 \frac{BC(p_1)}{BC(p_1) + BC(n_1)} \\
 241 \quad &= \log_2 \frac{p_1}{p_1 + n_1} \\
 242 \quad y_0 &= \log_2 \frac{BC(p_0)}{BC(p_0) + BC(n_0)} \\
 243 \quad &= \log_2 \frac{p_0}{p_0 + n_0} \\
 244 \quad \text{Gain}(r_i, r_j) &= BC(p_1) \times (y_1 - y_0) \\
 245 \quad &= p_1 \times \left(\log_2 \frac{p_1}{p_1 + n_1} - \log_2 \frac{p_0}{p_0 + n_0} \right).
 \end{aligned}$$

246 This proves the theorem.

247 However, it takes quadratic time in the number of potential cut points in the numerical
 248 range to find the best cut point [15]. In next subsection, we discuss how to improve the
 249 efficiency of finding the best cut point.

250 3.3 Cut points selection optimization

251 To improve efficiency, many techniques have been proposed to reduce the number of candi-
 252 dinate cut points [14–16]. Fayyad and Irani [16] introduced boundary points in analysis the
 253 binarization technique for the average class entropy and information gain. Elomaa and Rousu
 254 [14, 15] have given the definition of block and segment similar as following:

255 **Definition 7** If only one element of $CDV(Bi, C)$ is not equal to 0, we call such bins as class
 256 uniform bins. In order to construct blocks, merge together adjacent class uniform bins with
 257 the same class label. In order to construct segments, merge together adjacent bins with an
 258 equal relative class distribution.

259 Elomaa and Rousu [15] have proved that most common evaluation functions minimize
 260 on segment borders. Considering all classes at the same time is the common point of those
 261 functions. Because rule-based classification methods deal with one class at a time, the cut
 262 point selection can be optimized in a special way. In order to prove the theorem on optimizing
 263 the function $\text{Gain}(r_0, r_1)$, we give the following definition:

264 **Definition 8** Let (C_1, C_2, \dots, C_m) be the ordered classes according to their frequencies.
 265 Sequential covering algorithm extracts rules one class at a time from data sets. If C_k
 266 is selected to learn rules from the data set, we call C_k the classifying class, then the classifying
 267 class distribution of $CDV(Bi, C)$ is as follows:

$$268 \quad \frac{BC(Bi, C_k)}{BC(Bi, C_1) + \dots + BC(Bi, C_k) + \dots + BC(Bi, C_m)}.$$

269 The page is obtained by combining adjacent bins with an equal classifying class distribution.
 270 The boundary point between two adjacent pages is called page border.

271 According to Definitions 7 and 8, we know that both adjacent bins with an equal relative
 272 class distribution and adjacent class uniform bins also satisfy this condition; hence, uniform
 273 blocks and segments are special cases of pages. The following theorem shows that maximal
 274 $\text{Gain}(r_i, r_j)$ cannot occur within pages.

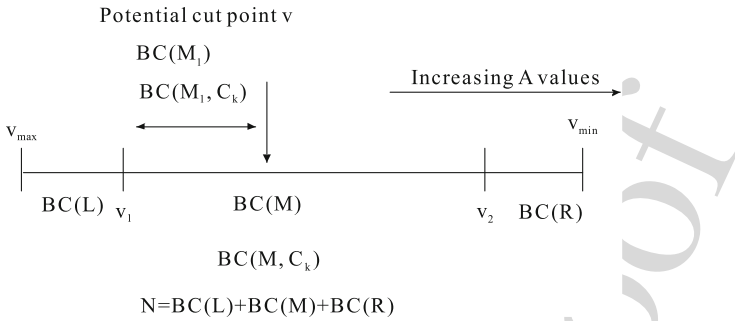


Fig. 1 Modeling a possible partition at v

275 **Theorem 2** *Optimal partitions based on the extended FOIL's information gain only occur*
 276 *on page borders.*

277 *Proof* In Fig. 1, v_{\min} and v_{\max} denote the minimum and maximum values of attribute A ,
 278 respectively. The bins are formed according to the values of attribute A of the tuples in the
 279 training data set. There is a page M with minimum value v_1 and maximum value v_2 . Assume
 280 that the classifying class is C_k and the classifying class distribution is $w_k \in [0, 1]$. We shall
 281 proceed to show that the optimal cut point v can not occur within M . Let $|M|$ denote the
 282 number of bins in page M . We will prove the theorem in the following two cases:

- 283 Case 1. If $|M| = 1$, then v can not occur within page M , which is obvious.
- 284 Case 2. If $|M| \geq 2$, we prove as follows.

285 Assume that v occurs somewhere within page M . Assume that M_1 bins in page M have
 286 an A -value less than v , $0 \leq M_1 \leq |M|$. Suppose there be L bins with A -values $< v_1$,
 287 and R bins with A -values $> v_2$, where $0 \leq BC(L), BC(R) \leq N - BC(M)$. Note that
 288 $BC(M) + BC(L) + BC(R) = N$ by definition (see Fig. 1). In order to simplify notation, let
 289 $BC(L) = l$, $BC(L, C_k) = l_k$, $BC(M_1) = x$ and $BC(M_1, C_k) = w_k x$. The crucial part of
 290 the theorem is that $BC(M_1, C_k) = w_k x$ and w_k does not depend on v , as long as v is inside
 291 the page. From Eq. (1), we know $\text{Gain}(r, r') = BC(p_1) \times (y_1 - y_0)$. So, we have

$$\begin{aligned}
 \text{Gain}(r_i, r_j) &= (BC(L, C_k) + BC(M_1, C_k)) \times \left(\log \frac{BC(L, C_k) + BC(M_1, C_k)}{BC(L) + BC(M_1)} - y_0 \right) \\
 &= (l_k + w_k x) \times \left(\log \frac{l_k + w_k x}{l + x} - y_0 \right). \tag{2}
 \end{aligned}$$

294 There are two subcases as following:

295 Subcase 1. $BC(L) = l = 0$. So $BC(L, C_k) = l_k = 0$ and Eq. (2) becomes

$$\begin{aligned}
 \text{Gain}(r_i, r_j) &= (l_k + w_k x) \times \left(\log \frac{l_k + w_k x}{l + x} - y_0 \right) \\
 &= w_k (\log w_k - y_0) x.
 \end{aligned}$$

298 If $w_k = 0$ or $y_0 = \log w_k$, $\text{Gain}(r_i, r_j) = 0$. In this situation, a split will not occur
 299 between v_1 and v_2 since 0 is the minimum value of extend FOIL's information gain function.
 300 If $w_k \neq 0$ and $y_0 \neq \log w_k$, the extreme value of $\text{Gain}(r_i, r_j)$ must be at the extremes of the
 301 interval.

Table 4 Pages and their CDVs

	[10, 20)	[20, 25)	[25, 30)	[30, 85)
CDV(P_a, C)	(1.3386, 0)	(1.0905, 0.0654)	(0.5641, 0.5905)	(0, 4.3305)

Subcase 2. $BC(L) = l > 0$. Taking the first derivative of Eq. (2) with respect to x gives

$$\frac{d}{dx} \text{Gain}(r_i, r_j) = w_k \log \frac{l_k + w_k x}{l + x} - w_k y_0 + w_k - \frac{l_k + w_k x}{l + x}.$$

Taking the second derivation with respect to x gives

$$\begin{aligned} \frac{d^2}{dx^2} \text{Gain}(r_i, r_j) &= w_k \left(\frac{w_k}{l_k + w_k x} - \frac{1}{l + x} \right) - \frac{w_k(l + x) - (l_k + w_k x)}{(l + x)^2} \\ &= \frac{w_k^2}{l_k + w_k x} - \frac{2w_k(l + x) - (l_k + w_k x)}{(l + x)^2} \\ &= \frac{(w_k(l + x) - (l_k + w_k x))^2}{(l_k + w_k x)(l + x)^2} \\ &= \frac{(w_k l - l_k)^2}{(l_k + w_k x)(l + x)^2}. \end{aligned}$$

If $w_k = \frac{l_k}{l}$, it reduces to Subcase 1. Otherwise, $\frac{d^2}{dx^2} \text{Gain}(r_i, r_j) > 0$. Thus, $\text{Gain}(r_i, r_j)$ is concave upwards and its maximum must be at one of the extremes of the interval. This proves the theorem.

Based on Theorem 2, for the bins shown in Table 3, we need only evaluate three candidate cut points at Annual Income = 20, 25 and 30. Thus, we can reduce the original set of 8 bins down to 4 pages as depicted in Table 4, where $\text{CDV}(P_a, C)$ denotes the class distribution of each page. Then only the class distribution vector of each page needs to be known in order to compute the extended FOIL's information gain defined on page borders.

4 Cut point selection in uncertain categorical data

The x -tuple model has been proposed in database systems such as [37] to model data uncertainty. In this section, we will discuss the method of cut point selection for uncertain categorical data.

4.1 A model for uncertain categorical data

According to x -tuple model [37], an uncertain data set D consists of a number of x -tuples. Each x -tuple T_j includes a number of *items* as its alternatives which are associated with probabilities, representing a discrete probability distribution of these alternatives being selected. Independence is assumed among the x -tuples. The probability of an item t is denoted as $p(t)$. Thus, an x -tuple T_j is a set of a bounded number of items, subject to the constraint that $\sum_{t \in T_j} p(t) = 1$. Following all earlier work on probabilistic databases, we view an uncertain data set as defining a probability distribution over a collection of possible worlds. Implicitly, a probabilistic data set encodes a large number of deterministic data sets, which

Table 5 Training set for predicting patient for 5 years survival

Rowid	Sex	Symptom	Tumor	Class
1	M	a	(Benign, 0.3:Malignant, 0.7)	1
2	M	b	(Benign, 0.2:Malignant, 0.8)	0
3	M	c	(Benign, 0.9:Malignant, 0.1)	1
4	F	b	(Benign, 0.3:Malignant, 0.7)	1
5	F	a	(Benign, 0.8:Malignant, 0.2)	1
6	F	a	(Benign, 0.4:Malignant, 0.6)	1
7	F	a	(Benign, 0.1:Malignant, 0.9)	0
8	M	c	(Benign, 0.4:Malignant, 0.6)	0
9	M	b	(Benign, 0.1:Malignant, 0.9)	0
10	F	b	(Benign, 0.2:Malignant, 0.8)	0
11	M	a	(Benign, 0.4:Malignant, 0.6)	0

are called possible worlds. Each possible world W occurs with the probability $P[W] = \prod_{T_j \cap W = t} P(t) \times \prod_{T_j \cap W = \emptyset} (1 - \sum_{t \in T_j} P(t))$.

There is a special x -tuple model called uncertain categorical attributes (UCA), in which each uncertain categorical attribute instance is distributed among all of its domain [32]. So, each uncertain attribute instance has the maximal number of alternatives. We also use A_j to denote the j th instance of uncertain categorical attribute A . A_j takes values from the categorical domain Dom with cardinality $|Dom| = m$. Within a certain relation, the value of an attribute instance A is a single value v_k in Dom , $P(A_j = v_k) = 1$. In the case of an uncertain relation, we record the information by a probability distribution over Dom instead of a single value.

Definition 9 Given a categorical domain $Dom = \{v_1, \dots, v_m\}$, an uncertain categorical attribute instance A_j is characterized by a probability distribution over its Dom , $P(A_j = v_k) = p_{jk}$ and $\sum_{k=1}^m p_{jk} = 1$ for $1 \leq k \leq m$.

Assume n is the number of instances and m is the number of candidate values for an attribute. Under the definition of UCA, we can record a UCA of a data set using an $n \times m$ matrix, which we call a categorical probability matrix, as shown in the following matrix.

$$\begin{pmatrix} p_{11} & p_{12} & p_{13} & \dots & p_{1m} \\ p_{21} & p_{22} & p_{23} & \dots & p_{2m} \\ \dots & \dots & \dots & \dots & \dots \\ p_{n1} & p_{n2} & p_{n3} & \dots & p_{nm} \end{pmatrix}.$$

Accurate data can be treated as a special case of uncertain data. When using the above matrix to represent uncertain categorical data, any data in this matrix can be a non-negative real number between 0 and 1; when using the above matrix to represent categorical data, each element is either 0 or 1, and there is exactly one element per row to be 1.

Table 5 shows an example of UCA. This data set is used for a medical diagnosis and it contains information for cancer patients with a tumor. The type of tumor for each patient is a UCA attribute, whose exact value is unobtainable. It may be either benign or malignant, each with associated probability. The class type of each instance is either 0 or 1, representing whether the patient survived 5 years after the diagnosis.

356 4.2 Cut point selection in possible worlds

357 An uncertain categorical attribute A is characterized by probability distribution over all possible
 358 values $v_k (1 \leq k \leq m)$. Each v_k for $1 \leq k \leq m$ is a candidate cut point. That is, a
 359 conjunct like “ $A = v_k$ ” can be added into the antecedent of a rule. We use a bin to denote the
 360 value of a categorical attribute or an uncertain categorical attribute. Then, the bin cardinality
 361 can be used for categorical attributes and uncertain categorical attributes. Similar to uncertain
 362 numerical attributes, we also define $P(A_j, r_i)$ and $P(T_j, r_i)$ for uncertain categorical
 363 attributes. If the cut point is v_k , then $P(A_j, r_i) = p_{jk}$. If r_i covers m attributes of T_j , then
 364 $P(T_j, r_i) = \prod_{l=1}^m P(A_l, r_i)$. Chui et al. [10] proposed the method to calculate the bin cardinality
 365 from the possible world semantics. For the sake of completeness, we give the proof
 366 as follows.

367 **Theorem 3** *The bin cardinality of the data set over a set of attribute values X is the sum*
 368 *of the probability of each tuple T_j which has the property $X \subseteq T_j$. That is, $BC(X) =$*
 369 $\sum_{j=1}^{|D|} \sum_{X \subseteq T_j} P(W_i)$.

370 *Proof* Let $N(X, W_i)$ be the number of X in possible world W_i and $N_j(X, W_i)$ be the indi-
 371 cator function whether X covers T_j w.r.t. possible world W_i . If $X \subseteq T_j$, $N_j(X, W_i) = 1$;
 372 otherwise, $N_j(X, W_i) = 0$. Then we have

$$\begin{aligned}
 373 \quad BC(X) &= \sum_{i=1}^{|W|} P(W_i) \times N(X, W_i) \\
 374 &= \sum_{i=1}^{|W|} P(W_i) \times \sum_{j=1}^{|D|} N_j(X, W_i) \\
 375 &= \sum_{j=1}^{|D|} \sum_{i=1}^{|W|} P(W_i) \times N_j(X, W_i) \\
 376 &= \sum_{j=1}^{|D|} \sum_{X \subseteq T_j} P(W_i).
 \end{aligned}$$

377 This proves the theorem.

378 In Theorem 3, if X is equal to v_k , then $BC(v_k) = \sum_{j=1}^{|D|} P(v_k \in T_j) = \sum_{j=1}^{|D|} p_{jk}$, where
 379 $BC(v_k)$ denotes the bin cardinality of the data set over an attribute value v_k . If X is equal to
 380 $\{v_k, C_i\}$, then $BC(v_k, C_i) = \sum_{j=1}^{|D|} P(v_k \in T_j \wedge C_{T_j} = C_i)$, where $BC(v_k, C_i)$ denotes the
 381 bin cardinality for class C_i of the data set over an attribute value v_k .

382 Refer to the data set shown in Table 5, the bin cardinality of Tumor = Benign is the
 383 overall probability of each tuple whose tumor attribute is Benign, which is 4.1, and the bin
 384 cardinality of Tumor = Malignant is 6.9. Similar to uncertain numerical attributes, the class
 385 distribution of an uncertain categorical attribute over each value can also be denoted by a
 386 vector $CDV(v_k, C) = (BC(v_k, C_1), BC(v_k, C_2), \dots, BC(v_k, C_m))$, and we call it the class
 387 distribution vector (CDV) over the attribute value v_k . For example, the bin cardinality for
 388 class 0 over Tumor = Benign is the overall probabilities of tuples in class 0 whose Tumor
 389 attribute is Benign, which is 1.4; and the bin cardinality for class 1 over Tumor = Benign is
 390 4.6. Thus, $CDV(\text{Benign}, C) = (1.4, 4.6)$.

391 Based on the CDV for each value, we can compute the extended information gain if data
 392 are split on that value, and the value with the highest extended information gain is the optimal
 393 cut point. The criteria for splitting based on UCA is very similar to UNA. The differences
 394 are as follows:

- 395 – For a UNA, the page borders are candidate cut points. For a UCA, all possible values of
- 396 an uncertain categorical attribute are candidate cut points.
- 397 – For a UNA A , we need to identify a cut point v and the cut direction, which can be either
- 398 $v_k < v$ or $v_k \geq v$. Whereas for a UCA A , we need only find the cut point v_k , which is
- 399 one value of attribute A and the cut criteria can only be equality $A = v_k$.

400 5 The uRule algorithm and its prediction

401 This section will present the uRule algorithm in detail and also the prediction process for
 402 uncertain data using rules.

403 5.1 The uRule algorithm

404 A naive way to deal with the uncertain numerical data are to replace each PDF with its
 405 expected value, thus effectively converting the data tuples to point-valued tuples. This reduces
 406 the problem back to that for point-valued data, and hence traditional rule-based algorithm
 407 such as RIPPER can be used. This method, although simple and straight-forward, can cause
 408 valuable information loss. Below, we discuss the uRule algorithm, which is designed based
 409 on the framework of RIPPER, for rule learning from uncertain data. The differences between
 410 uRule and RIPPER are as follows:

- 411 – uRule can work on both certain and uncertain data. It uses the extended FOIL's informa-
 412 tion gain as the measure to find the best antecedents and generate rules.
- 413 – In RIPPER, a tuple is either full covered or not covered by a rule. In uRule, a tuple can
 414 be partially covered by a rule; therefore, we must provide a mechanism for handling this
 415 partial coverage situation.

416 Similar as RIPPER, we perform a 3-fold stratified holdout, taking out 2/3 the data by
 417 random sampling. We use those data to build rules and use the remaining 1/3 data for prun-
 418 ing. Function uGrow() discusses the process of generating rules from uncertain data sets, and
 419 we use it as an example to illustrate the overall design. It is shown in Algorithm 1, whose
 420 basic strategy is as follows:

- 421 – It begins with an empty set at the left-hand side and the target class at the right-hand side.
- 422 – If an attribute is numerical or uncertain numerical, Theorem 2 is used to speed up the
 423 cut point selection. If an attribute is categorical or uncertain categorical, Theorem 3 is
 424 used to select the cut point by one scan over the uncertain data. The algorithm selects
 425 the cut point which has the highest extended FOIL's information gain and adds it into the
 426 antecedent of the rule (steps 3–4).
- 427 – If the selected attribute is numerical or uncertain numerical, then we find the cut point
 428 v and the cut direction (either $<$ or \geq). (steps 5–7). If a tuple is covered by the rule,
 429 Function uSplit() is invoked (steps 8–13). uSplit() returns part of the tuple that is covered
 430 by the rule, and later that part of the tuple will be removed from the data set which is used
 431 for subsequent rule growing.
- 432 – If the attribute is categorical or uncertain categorical, the cut point v is a value of the
 433 categorical attribute. Function uSplit() is also invoked to return part of the tuple that is

Input: Instances Data

```

1 growData := Data and coverData :=  $\emptyset$ ;
2 while growData.size() > 0  $\wedge$  AttributeUnused > 0 do
3   select the attribute  $A$  with the highest extended FOIL's information gain;
4   Ant := Ant + RuleAnt( $A$ );
5   if  $A$  is a numeric or uncertain numeric attribute then
6     get the cut direction  $x$ ;
7     binary split the data from the cut point  $v$ ;
8     while  $T_j \in Data$  do
9       if covers( $T_j$ ) then
10        inst := uSplit( $T_j, A, x, v$ );
11        coverData := coverData + inst;
12      end
13    end
14  else
15    get the cut point  $v$ ;
16    while  $T_j \in Data$  do
17      if covers( $T_j$ ) then
18        inst := uSplit( $T_j, A, 0, v$ );
19        coverData := coverData + inst;
20      end
21    end
22  end
23 growData := growData - coverData;
24 end

```

Algorithm 1: uGrow()

434 covered by the rule (steps 16–21). Again, this part of the tuple will then be removed from
 435 the training data set which is used for subsequent rule growing.

436 Function uSplit() decides the part of the tuple that is covered by the rule and computes
 437 the corresponding weight. It is shown in Algorithm 2 with the following procedure:

- 438 – If A is certain, then return T_j .
 439 – If A is uncertain numerical, then it must be either one of the two cases: Case 1. If it is
 440 completely covered by the antecedent, then return T_j . Case 2. If it is partly covered by
 441 the antecedent, then calculate the new weight after the split (steps 2–6).
 442 – If A is uncertain categorical, then for every value of $A(i)$ which is not covered by the
 443 antecedent, set the weight to be 0 (steps 8–10).
 444 – If there is a split on any attribute, we recompute the weight of the instance according to
 445 the degree of the coverage (steps 12–17).

446 *Example 2* Based on the data set shown in Table 1, uRule learns the following rule set:
 447 (Annual_Income \leq 25) \Rightarrow class = Yes (2.653/0.065)
 448 \Rightarrow class = No (9.347/1.412)

449 This shows that for the first rule, there are 2.653 tuples covered by the rule, that is, 2.653
 450 tuples satisfy the condition of the rule and out of them 0.065 tuple is false positive. The
 451 second rule is a default rule, which means that for tuples that are not covered by any other
 452 rule, it will be predicted to be in class No. Based on the training data set in Table 1, 9.347
 453 tuples are covered by this default rule, out of which 1.412 tuples are false positives.

```

Input: (tuple  $T_j$ , attribute  $A$ , cut direction  $y$ , cut point  $v$ )
1 if  $A$  is certain then return  $T_j$ ;
2 else if  $A$  is uncertain numerical then
3   if  $((y = 1) \wedge (T_j.A.min \geq v)) \vee ((y = 0) \wedge (T_j.A.max \leq v))$  then
4     return  $T_j$ ;
5   if  $y = 1$  then  $T_j.A.w := \phi(T_j.A.max) - \phi(v)$ ;
6   else  $T_j.A.w := \phi(v) - \phi(T_j.A.min)$ ;
7 else
8   for  $i = 0; i < A.numValues(); i ++$  do
9     if  $i! = v$  then  $T_j.A(i).w := 0$ ;
10  end
11 end
12  $T_j.w := 1$ ;
13 for  $i = 0; i < numAttrs(); i ++$  do
14   if  $A$  is uncertain numerical then  $T_j.w := T_j.w \times T.A.w$ ;
15   else if  $A$  is uncertain categorical then  $T_j.w := T_j.w \times \sum_{i=1}^m T_j.A(i).w$ ;
16   else  $T_j.w := T_j.w \times 1$ ;
17 end
18 return  $T_j$ ;
    
```

Algorithm 2: uSplit()

454 5.2 Prediction with uRule

455 Once the rules are learned from a data set, they can be used to predict class types of previously
 456 unseen data. Like a traditional rule classifier, each rule generated by uRule is in the form of “IF
 457 Conditions THEN Class = C_k ”. Because each T_j may be covered by more than one rules, we
 458 can compute a vector for each T_j ($CDV(T_j, C) = (P(T_j, C_1), P(T_j, C_2), \dots, P(T_j, C_m))$),
 459 in which $P(T_j, C_k)$ denotes the probability for a tuple to be in class C_k . The following defini-
 460 tion is about how to calculate the probability of a rule covering several attributes of an
 461 instance.

462 **Definition 10** We denote the degree of an instance A_j covered by a rule r_i by $P(A_j, r_i)$ and
 463 the degree of T_j covered by a rule r_i by $P(T_j, r_i)$. $P(A_j, r_i) = \frac{P(A_j \in [v, b])}{A_j.w}$ or $P(A_j, r_i) =$
 464 $\frac{P(A_j \in [a, v])}{A_j.w}$ according to the cut direction. When $P(A_j, r_i) = 1$, r_i fully covers A_j ; when
 465 $0 < P(A_j, r_i) < 1$, r_i partially covers A_j ; and when $P(A_j, r_i) = 0$, r_i does not cover A_j .
 466 If r_i covers m attributes of T_j , then $P(T_j, r_i) = \prod_{l=1}^m P(A_l, r_i)$.

467 Here, $A_j.w$ is the weight of instance A_j . The weight starts with 1. Each time a rule r_i is
 468 generated, the weight of the part of instance A_j that is covered by the rule is computed as
 469 $P(A_j, r_i)$, and the weight of the other part becomes $A_j.w - P(A_j, r_i)$. Suppose T_j is covered
 470 by m rules, then $CDV(T_j, C)$ is computed as follows:

471
$$CDV(T_j, C) = \sum_{i=1}^m P(r_i, C) \times P(T_j, r_i).$$

472 Here, $P(r_i, C)$ is a vector $P(r_i, C) = (P(r_i, C_1), P(r_i, C_2), \dots, P(r_i, C_m))$ and it
 473 denotes the class distribution of the tuples covered by rule r_i . $P(r_i, C_k) = \frac{BC(r_i, C_k)}{BC(r_i)}$, where
 474 $BC(r_i)$ denotes the bin cardinality covered by rule r_i and $BC(r_i, C_k)$ denotes the bin car-
 475 dinality covered by rule r_i and belonging to class C_k . After we compute $CDV(T_j, C)$, the

476 tuple will be predicted to be of class C_k which has the largest probability among the vector
477 $CDV(T_j, C)$.

478 *Example 3* Suppose we have a test tuple {No, Married, 20–28, ?}, based on the uncertain
479 training data shown in Table 1, we need predict whether this customer will default on loan.
480 The prediction procedure is as follows:

481 When applying the uRule algorithm on data in Table 1, the rule set is generated as shown in
482 Example 2. The cut point of the uncertain numerical attribute Annual Income is 25.0. Because
483 its PDF is a normal distribution in [20, 28], $P(A \in [20, 25]) = \phi((25 - 24)/1.3) - \phi((20 -$
484 $24)/1.3) = 0.772$ and $P(A \in [25, 28]) = \phi((28 - 24)/1.3) - \phi((25 - 24)/1.3) = 0.228$.
485 Then, we apply these rules on the test tuple and compute its class distribution vector as
486 follows:

$$\begin{aligned}
 487 \quad CDV(T_j, C) &= P(r_1, C) \times P(T_j, r_1) + P(r_2, C) \times P(T_j, r_2) \\
 488 \quad &= \begin{pmatrix} (2.653 - 0.065)/2.653 \\ 0.065/2.653 \end{pmatrix} \times 0.772 + \begin{pmatrix} (9.347 - 1.412)/9.347 \\ 1.412/9.347 \end{pmatrix} \times 0.228 \\
 489 \quad &= \begin{pmatrix} 0.9753 \\ 0.0247 \end{pmatrix} \times 0.772 + \begin{pmatrix} 0.1511 \\ 0.8489 \end{pmatrix} \times 0.228 \\
 490 \quad &= \begin{pmatrix} 0.753 \\ 0.019 \end{pmatrix} + \begin{pmatrix} 0.0344 \\ 0.1935 \end{pmatrix} \\
 491 \quad &= \begin{pmatrix} 0.7874 \\ 0.2126 \end{pmatrix}.
 \end{aligned}$$

492 This means the test tuple is in the “Yes” class with probability 78.74% and in the “No” class
493 with probability 21.26%. Thus, the tuple will be predicted to be in class “Yes”.

494 6 Experiments

495 We have implemented uRule as described in Sect. 5 to classify uncertain data sets. When
496 uRule is applied on certain data, it works as a traditional RIPPER classifier, which has
497 been implemented in Weka [38]. To make numerical attributes uncertain, we convert each
498 numerical value to an uncertain interval with normal distribution as in [5, 13] and [24–26].
499 The uncertain interval is generated around the original value, which is the center point of
500 the interval. In [34], the uncertain interval is relative to the domain size of each attribute.
501 We modify their source code to handle uncertain interval around the original value. In this
502 section, the data set with 10% uncertainty is denoted by U10. For a certain value x , its
503 $U10 = [x - |x| \times 0.05, x + |x| \times 0.05]$. For example, if an original value is 20, then its
504 U10 has an interval [19, 21]. Other notations have the similar meaning. We use U0 to denote
505 accurate or certain data sets.

506 In this paper, the baseline scheme adopts the DTU algorithm, which is an extension of
507 J4.8 [38], a java implementation of the traditional C4.5 decision tree classifier. If DTU is
508 applied on certain data, it works the same as J4.8. The UDT algorithm [34] adopt the same
509 technique as DTU algorithm [25] in splitting tuples into subsets when the domain of its PDF
510 spans across the cut point.

511 In the following experiments, we use ten times ten-fold cross-validation. For each ten-fold
512 cross-validation, data are split into 10 approximately equal partitions; each one is used in turn
513 for testing, while the rest is used for training, that is, 9/10 of data are used for training and 1/10
514 for test. We employed corrected paired t-test to test whether uRule has a significantly better

Table 6 Characteristic figures of the ten numerical data sets

Data sets	Attrs	m	n	\bar{V}	\bar{B}	\bar{G}	\bar{P}
<i>E.coli</i>	7/7	8	336	51.9	44.1	43.7	4.1
Heartc	13/13	5	303	30.5	27.4	27.2	11.6
Iris	4/4	3	150	30.8	15	14.5	7.5
Page block	10/10	5	5,473	909.2	338.9	337.1	40.1
Satellite	36/36	6	4,435	76.3	62.6	62.4	42.8
Segment	19/19	7	2,310	628.3	404.1	400.6	140.7
Shuttle	9/9	7	58,000	123.2	85.3	85	11.9
Waveform	21/21	3	5,000	714	653.2	636.9	510.3
Wine	13/13	3	178	98.1	55.6	55.3	35.2
Yeast	8/8	10	1,484	51.5	47.9	47.8	7.1

515 prediction accuracy than DTU. The experiments are executed on a PC with an Intel Pentium
 516 IV 3.2GHZ CPU and 4.0GB main memory. In this section, we present the experimental
 517 results of the proposed uRule algorithm. We studied the uRule classifier accuracy, the effect
 518 of the optimization technique and classifier construction time over uncertain data sets.

519 6.1 Performance on uncertain numerical data

520 6.1.1 The effect of optimization

521 We first study the effect of the optimization technique describe in Sect. 3.3. As test domains,
 522 we use 10 data sets, which are described in Table 6. All data sets can be downloaded from the
 523 UCI repository [2]. In Table 6, column m is the number of classes, n is the total number of
 524 tuples in the domain, ATTRS gives the number of numerical attributes out of the total number
 525 of attributes, \bar{V} is the average number of bins for an attribute, \bar{B} is the average number of
 526 blocks for an attribute, \bar{G} is the average number of segments for an attribute and \bar{P} is the
 527 average number of pages for an attribute. If a data set only has two classes, it has the same
 528 number of segment borders and page borders. Thus, we only select the data sets, which have
 529 more than two classes as in [14, 15], which discusses the approaches to optimize the cut
 530 point selection for certain data sets. For these 10 “real-world” data sets shown in Table 6, our
 531 algorithm has the similar properties as the algorithms shown in [14, 15].

- 532 – It holds with only few exceptions that $\bar{B} < \bar{V} \ll n$.
- 533 – On the average, approximately 50% of bin borders are boundary points. However, the
 534 differences between domains are huge. The average number of segments borders per
 535 attribute is only marginally smaller than that of the boundary points.

536 Our algorithm also has the following specific property comparing with the algorithms
 537 shown in [14, 15].

- 538 – The average number of page borders per attribute is much smaller than that of the seg-
 539 ment borders. By combining bins into pages, the reduction in the number of points that
 540 needs to be examined comes from three main sources: combination of class uniform bins,
 541 combination of mixed bins with an equal relative class distribution and combination of
 542 mixed bins with an equal classifying class distribution. In most situations, the last one is
 543 an important source. The reason for this is obvious: uRule begins with the least frequent

Table 7 Characteristic figures of the ten data sets with all numerical attributes uncertain

Data sets	U1				U5			
	\bar{V}	\bar{B}	\bar{G}	\bar{P}	\bar{V}	\bar{B}	\bar{G}	\bar{P}
<i>E.coli</i>	103.1	77.1	43.7	4	102.1	88.7	86.7	10.6
Heartc	60.8	53.7	45.5	17.8	60.3	58.5	55.6	39.0
Iris	61.5	25.8	14.5	7.5	61	30.5	29.3	14.5
Page block	1,797	962.8	886.8	130.1	1,794	1,528	1,513	353.1
Satellite	152.6	122.7	75.7	49.9	151.4	129.9	128.7	93.4
Segment	1,175.3	905.5	843.3	451.8	1,172.5	964.3	950.8	458.1
Shuttle	245.9	146.4	94	12.1	244.7	171.2	149.7	23.7
Waveform	1,419	1,315	1,145	935.6	1,414	1,382	1,347	1,161
Wine	196.1	114	103	66.5	195.2	152.9	152	108.2
Yeast	102.3	87.3	47.8	7.1	101.4	95	92	18.1
Data sets	U10				U20			
<i>E.coli</i>	101.1	91.6	90.7	17.4	98.9	91.9	91.3	28.4
Heartc	59.8	58.8	56.6	43.5	59.1	58.3	56.6	50.0
Iris	60.3	34	33.5	16.5	59.3	37	36.3	20.3
Page block	1,781	1,666	1,661	505.1	1,755	1,717	1,715	697.3
Satellite	150.7	133.6	133	103.7	147.9	134.3	134	110.7
Segment	1,171.3	979.7	973.6	897.1	1,166.5	992.4	989.5	557
Shuttle	242.8	187.8	176.2	34.6	239.8	206.2	200.8	48.9
Waveform	1,396	1,379	1,360	1,192	1,365	1,354	1,346	1,200
Wine	194.2	161.8	161.5	123.5	192.4	168.2	168.2	137.7
Yeast	100.4	96.5	95.3	26.8	98.4	95.9	95.3	35.3

544 class C_1 . The number of tuples belongs to class C_1 is minimal. So, those tuples are only
 545 distributed among small number of attribute values. Thus most bins can be merged.

546 The results of 10 uncertain data sets after optimization are shown in Table 6. Besides
 547 the above three properties of certain data sets, their respective \bar{V} , \bar{B} , \bar{G} and \bar{P} are much
 548 larger than those of certain data sets. The reason is that each uncertain numerical attribute
 549 instance has two critical values. From Tables 6 and 7, we know that optimization based on
 550 Theorem 2 is effective in reducing the cut point candidates and accelerates the selection for
 551 both numerical data and uncertain numerical data.

552 6.1.2 Accuracy

553 As prediction accuracy is by far the most important measure for a classification method,
 554 we studied the prediction accuracy of uRule. We studied the prediction accuracy of uRule
 555 in cases when only one single attribute is uncertain and when all numerical attributes are
 556 uncertain.

557 1. Single attribute uncertain

558 To choose which attribute to be uncertain, we use the information gain attribute selection
 559 method, which is a commonly used feature selection method. It chooses the attribute with
 560 the highest information gain, which is the most discriminating attribute. This important

Table 8 Accuracy of uRule vs. DTU over data sets with one numerical attribute uncertain

Data sets	uRule					
	U0	Best	U1	U5	U10	U20
<i>E.coli</i>	81.93	81.78	81.5	81.49	81.71	81.78
Heartc	54.4	54.4	54.16	54.24	54.07	54.39
Iris	94.5	94.17	93.5	92.5	93.67	94.17
Page block	96.99	97.15	97.08	97.15	97.02	97.06
Satellite	85.38	85.89	85.37	85.89	85.76	85.87
Segment	95.65	95.41	95.32*	95.02*	95.41*	95.36*
Shuttle	99.97	99.97	99.97	99.96	99.97	99.97
Waveform	79.74 _v	79.91	79.18 _v	79.91 _v	79.65 _v	79.54 _v
Wine	93.37	92.71	92.12	92.71	92.12	91.73
Yeast	58.19	58.1	57.5	58	57.92	58.1
Data sets	DTU					
<i>E.coli</i>	83.28	82.54	83.35	82.54	82.46	83.42
Heartc	53.66	55.22	55.22	55.22	55.22	54.89
Iris	94.83	94.33	93.67	94.17	93.83	94.33
Page block	96.99	97.13	97.06	97.04	97.07	97.13
Satellite	85.96	86.07	85.8	86.07	85.58	85.53
Segment	96.82	96.77	96.77	96.73	96.77	96.61
Shuttle	99.97	99.98	99.97	99.97	99.98	99.98
Waveform	76.52	76.55	76.55	76.26	76.21	76.27
Wine	93.82	93.54	93.54	91.99	92.27	93.1
Yeast	56.31	57.06	57.06	56.82	56.67	57.01

attribute may appear in multiple rules, and the rule classifier will almost certainly choose it as the first attribute to split on.

UDT [34] only handles uncertain data sets with all attributes uncertain; therefore, in this experiment, we only compare the accuracies of uRule and DTU. The results of uRule and DTU over uncertain numerical data sets and their respective certain parts (U0) are shown in Table 8, in which each cell is the average accuracy. The annotation *v* or * indicates that a specific result is statistically better (*v*) or worse (*) than the baseline scheme (DTU) at the significant level specified (currently 0.05). If there is no annotation in a cell, the specific result of uRule is statistically the same as that of DTU.

Comparing the second and the third columns of Table 8, the average accuracies between the certain data sets (denoted by U0) and their respective uncertain parts are small. Comparing uRule with DTU, we find the accuracy of uRule is statistically the same or vary close to DTU for most data sets. Both DTU and uRule are suitable for mining data sets with one numerical attribute uncertain.

2. All numerical attributes uncertain

In this experiments, all numerical attributes are uncertain. The results are shown in Table 9. Comparing the second and the third columns of Table 9, we find that uRule can potentially generate more accurate rules from uncertain data sets than from their corresponding certain parts in most cases. For instance, for the wine data set, the average

Table 9 Accuracy of uRule vs. DTU and UDT over data sets with all numerical attributes uncertain

Data sets	uRule					
	U0	Best	U1	U5	U10	U20
<i>E.coli</i>	81.93	81.99	80.96	81.77	81.64	81.99
Heartc	54.4	54.14	53.98	53.81	54.14	54.06
Iris	94.5	95.33	93.17	92.17	93.33	95.33
Page block	96.99	97.25	97.17	97.1	97.25	97.23
Satellite	85.38	85.61	85.39	85.61	85.43	85.37
Segment	95.65	95.7	95.25	95.7*	94.92*	94.61*
Shuttle	99.97	99.84	99.27*	99.84	99.58*	99.56*
Waveform	79.74 _v	79.74	79.74 _v	78.87	79.34 _v	79.27 _v
Wine	93.37	95.65	94.49	93.56	95.65	94.24
Yeast	58.19	58.59	58.41	58.09	58.3	58.59
Data sets	DTU					
<i>E.coli</i>	83.28	81.27	80.59	79.84	79.54	81.27
Heartc	53.66	56.61	56.44	56.61	56.52	57.44
Iris	94.83	95	93.67	94	94.67	95
Page block	96.99	97.24	97.13	97.24	97.2	97.2
Satellite	85.96	87.32	87.19	87.32	87.31	87.18
Segment	96.82	96.76	96.5	96.76	96.46	96.17
Shuttle	99.97	99.98	99.95	99.95	99.98	99.98
Waveform	76.52	77.93	77.61	77.47	77.63	77.93
Wine	93.82	95.49	93.39	93.82	94.63	95.49
Yeast	56.31	59.03	58.76	58.67	59.03	58.02
Data sets	UDT					
<i>E.coli</i>	71.13	74.91	72.9	70.83	73.49	74.91
Iris	94.67	96.67	94.67	95.33	96.67	96.67
Page block	96.27	96.44	96.44	96.24	96.16	96.24
Satellite	81.82	83.76	81.73	83.76	82.57	74.72
Waveform	76.12	75.96	75.96	75.82	75.4	75.06
Wine	88.89	93.89	89.38	91.6	93.89	89.97

580 accuracy is improved from 93.37 to 95.65%. DTU can build more accurate decision trees
581 from uncertain data sets than from their corresponding certain parts in most cases. For
582 instance, for the heartc data set, the average accuracy is improved from 53.66 to 56.61%.
583 We also compare the accuracy of DTU and uRule with UDT. Out of the ten data sets,
584 UDT can only work on six of them (due to the fact that UDT can not handle uncer-
585 tainty interval such as [0, 0]). As shown in Table 9, UDT also potentially generates more
586 accurate rules from uncertain data sets than from their corresponding certain parts. For
587 instance, for the wine data set, the average accuracy is improved from 88.89 to 93.89%.
588 Comparing uRule with DTU, we find the results of uRule are statistically the same as
589 those of DTU on most data sets and both algorithms have their respective advantages
590 when mining different data sets. Because UDT is not based on J4.8, UDT and DTU have
591 different accuracies when classifying both certain and uncertain data sets. The experi-
592 ments show that DTU classifier is slightly more accuracies than UDT on most data sets.

Table 10 Classifier construction time (second) of uncertain numerical data sets

Data sets	uRule				
	U0	U1	U5	U10	U20
<i>E.coli</i>	0.07	0.36	0.43	0.63	0.87
Iris	0.05	0.06	0.07	0.08	0.08
Heartc	0.13	0.34	0.72	0.83	1.07
Page block	1.87	14.85	24.57	34.48	38.37
Satellite	8.12	32.12	32.27	64.32	120.1
Segment	1.41	11.78	14.28	44.13	48.37
Shuttle	26.5	42.36	64.14	78.66	127.1
Waveform	6.51	41.92	44.15	45.41	132.8
Wine	0.02	0.13	0.27	0.31	0.48
Yeast	0.53	1.62	1.89	3.1	3.61
Data sets	DTU				
<i>E.coli</i>	0.26	0.08	0.1	0.13	0.1
Iris	0.02	0.03	0.05	0.05	0.03
Heartc	0.03	0.08	0.06	0.09	0.13
Page block	0.41	3.02	6.01	8.1	13.21
Satellite	0.95	2.06	3.09	3.12	4.93
Segment	0.2	1.42	2.83	4.23	6.31
Shuttle	5.94	3.92	4.96	7.21	11.2
Waveform	0.78	4.9	6.66	8.67	8.58
Wine	0.03	0.06	0.11	0.16	0.2
Yeast	0.14	0.17	0.2	0.22	0.25
Data sets	UDT				
<i>E.coli</i>	0.1	0.2	0.2	0.2	0.3
Iris	0.02	0.03	0.05	0.05	0.03
Page block	0.41	1.72	2.32	4.3	4.7
Satellite	0.95	1.58	3.36	4.78	6.02
Waveform	1.5	1.86	2.22	3.16	4.7
Wine	0.03	0.06	0.18	0.18	0.44

593 6.1.3 Computation time

594 We investigate the time it takes to construct a classifier. When only one attribute is uncertain,
 595 the time difference between generating rules from uncertain data sets and their corresponding
 596 certain parts is little. So, we only show the results when all numerical attributes are uncertain.
 597 Table 10 depicts the absolute run time in second when all tuples of a data set are used to
 598 generate the rules by uRule and to build a decision tree by DTU and UDT. When data are
 599 certain, the time it takes to construct a classifier is determined by the data set size in terms
 600 of both the number of instances and dimensionalities. For example, Shuttle is the largest one
 601 among all data sets; therefore, it takes longest to build a classifier for both uRule and DTU.

602 From Table 10, we observe that it takes longer to construct a classifier from an uncertain
 603 data set than from its corresponding certain data set in most situations. The reason is that
 604 each uncertain attribute contains more candidate cut points and consequently incurs more

Table 11 Characteristic figures of the ten categorical data sets

Data set	Attrs	m	n
Acute	6/8	2	120
Contracep	8/9	3	1,473
Derm	33/34	6	366
Hayes	4/5	3	132
Mushroom	22/22	2	8,124
Promote	57/57	2	106
Soybean	35/35	19	307
Teacher	4/5	3	151
Voting	16/16	2	435
Zoo	15/16	7	101

605 computation. Moreover, for each uncertain attribute instance, its interval may span across a
 606 cut point. This will result in tuple split and bin cardinality computation, which do not exist
 607 in building a classifier over a certain data set. We further observe that whether the data sets
 608 are certain or uncertain, it takes more time to construct an uRule classifier than to construct
 609 a DTU classifier and a UDT classifier. This means it takes longer to construct an uRule clas-
 610 sifier than to construct a DTU classifier and a UDT classifier even if they are applied over
 611 certain data sets. This is because uRule takes more time to prune than DTU and UDT. It is
 612 also shown that the UDT classifier is slightly faster than DTU on a few data sets due to its
 613 simplified implementation.

614 6.2 Categorical data sets

615 As test domains, we use 10 data sets from the UCI repository [2]. The data sets are described
 616 in Table 11. To make categorical attributes uncertain, we convert each categorical value into
 617 a probability vector as shown in [5, 13] and [24–26]. For example, a categorical attribute A
 618 may have m possible values v_k for $1 \leq k \leq m$. For tuple T_j , we convert its value A_j into a
 619 probability vector $\mathbf{P} = (p_{j1}, p_{j2}, \dots, p_{jm})$, while p_{jk} is the probability of A_j to be equal
 620 to v_k , that is, $P(A_j = v_k) = p_{jk}$. For example, when we introduce 10% uncertainty, this
 621 attribute will take the original value with 90% probability, and 10% probability to take any
 622 of the other values. Suppose in the original accurate data set $A_j = v_1$, then we will assign
 623 $p_{j1} = 90\%$, and assign p_{jk} for $2 \leq k \leq m$ to ensure $\sum_{k=2}^m p_{jk} = 10\%$. Because the
 624 UDT classifier can not work on uncertain categorical data sets, we only compare the uRule
 625 classifier with the DTU classifier in this subsection.

626 6.2.1 Accuracy

627 Similar as before, we studied the prediction accuracy of uRule in cases when only one single
 628 categorical attribute is uncertain and when multi-categorical attributes are uncertain.

629 1. Single attribute uncertain

630 To choose which attribute to be uncertain, we also use the information gain attribute
 631 selection method. We choose the most important attribute with the highest information
 632 gain value. The experimental results are shown in Table 12. For both algorithms, we find
 633 the average accuracies between the certain data sets (denoted by U0) and their respective

Table 12 Accuracy of uRule vs. DTU over data sets with one categorical attribute uncertain

Data sets	uRule					
	U0	Best	U1	U5	U10	U20
Acute	100	100	100	100	100	100
Contracep	52.54	54.70v	53.89v	54.70v	53.70v	53.20v
Derm	87.65*	88.46*	88.46*	87.86*	87.7*	88.39*
Hayes	77.46*	76.42	76.42	73.82	70.62	63.04*
Mushroom	100	100	100	100	100	100
Promote	81.27	78.85	78.85	78.78	75.91	75.5
Soybean	85.4	84.88	84.81	84.74	84.88	83.96
Teacher	39.26*	34.65*	34.15*	34.65*	33.87*	33.84*
Voting	95.75	95.63	95.63	94.16	94.05	90.68
Zoo	89.91	90.21	90.21	90.01	89.91	89.91
Data sets	DTU					
Acute	100	100	100	100	100	100
Contracep	51.57	52.54	52.54	52.36	51.93	49.88
Derm	94.1	94.14	94.14	93.46	93.46	93.46
Hayes	71.7	74.64	72.76	71.62	72.93	74.64
Mushroom	100	100	100	100	100	100
Promote	79.04	77.45	77.45	75.82	76.39	76.84
Soybean	84.03	86.65	86.65	86.65	86.57	86.48
Teacher	50.86	52.32	52.32	41.23	41.56	41.73
Voting	96.62	96.14	96.14	94.83	94.89	91.32
Zoo	92.61	92.84	92.84	91.36	91.36	91.36

634 uncertain parts are within 5% in most situations. For uRule, there is only a few exceptions,
 635 for example, the U10 and U20 of the promote data set decrease more than 5%. For DTU,
 636 there is also a few exceptions; for example, the U5, U10 and U20 of the teacher data set
 637 decrease more than 5%. The accuracy of uRule and DTU vary on different data sets. On
 638 some data sets, uRule takes a lead, while on the others, DTU is better. In general, both
 639 algorithms work well on these uncertain data sets and they can generate good-quality
 640 classifier even when the uncertainty is high.

641 2. Multiple categorical attributes uncertain

642 In this experiment, we try to make multiple categorical attributes uncertain. Since some
 643 of the data sets contain more than 30 or even 50 attributes, we process the data sets as
 644 following: if the number of attributes in a data set is less than or equal to 6, we make all
 645 of its attributes uncertain; otherwise, we make its first 6 attributes uncertain according to
 646 information gain selection method.

647 The experimental results are shown in Table 13. We find both algorithms have better
 648 accuracies over certain data sets than their corresponding uncertain parts in most cases.
 649 We also observe that uRule is more stable than DTU on uncertain data sets. For example,
 650 the accuracies of DTU over U5 of the soybean and zoo data sets drop to 0, while the
 651 performance of uRule is much more stable. This experiment shows that uRule is more
 652 suitable for mining uncertain categorical data than DTU.

Table 13 Accuracy of uRule vs. DTU over data sets with multi-categorical attributes uncertain

Data sets	uRule					
	U0	Best	U1	U5	U10	U20
Acute	100	99.83v	99.5v	99.83v	99v	91.67
Contracep	52.54	54.7v	53.89v	54.7v	53.7v	53.2v
Derm	87.65*	86.6*	82.4v	83.26*	86.6*	85.57*
Hayes	77.46v	60.52v	60.52v	60.16v	60.34v	55v
Mushroom	100	99.97v	99.97v	99.14v	98.79	98.72
Promote	81.27	71.45v	71.42v	71.45v	69.44*	68.4*
Soybean	85.4	78.77	72.48v	78.06v	78.77	78.44*
Teacher	39.26*	34.56*	34.56*	33.92*	33.63*	34.44*
Voting	95.75	90.72*	90.72*	88.64*	88.68*	89.28v
Zoo	89.91	86.13v	79.15	86.13v	80.58v	78.22
Data sets	DTU					
Acute	100	91.67	27.5	27.5	26.67	91.67
Contracep	51.57	53.78	51.34	53.78	53.32	48.22
Derm	94.1	94.75	46.73	94.75	94.75	94.75
Hayes	71.7	43.97	43.97	43.97	43.97	43.78
Mushroom	100	98.67	98.52	98.52	98.52	98.67
Promote	79.04	72	67.93	69.89	70.36	72
Soybean	84.03	84.92	19.64	0	75.99	84.92
Teacher	50.86	50.16	49.83	49.68	50.16	47.82
Voting	96.62	94.48	94.48	94.48	94.48	5.21
Zoo	92.61	80.18	81.68	0	25.25	80.18

6.2.2 Computation time

We also compare the classifier construction time for both uRule and DTU on uncertain categorical data. Table 14 depicts the absolute run time in second for uRule to build a whole rule set and for DTU to build a decision tree. We only show the results when multiple categorical attributes uncertain. We observe that it takes longer to construct a classifier for an uncertain data set than its corresponding certain one in most situations. For the uRule algorithm, the reason is that each uncertain attribute instance of a tuple may be either covered or partly covered by a rule and incur more computation for $BC(p_0)$ and $BC(n_0)$. Thus for a rule set, it takes longer to grow and prune. For the DTU algorithm, the reason is that each uncertain attribute instance of a tuple may be partitioned into different subtrees and incur more computation.

With the similar reason as in the uncertain numerical data sets, we further observe that no matter the data sets are certain or uncertain, it takes longer to construct an uRule classifier than to construct a DTU classifier, due to the complexity in rule pruning.

7 Related work

Uncertain data, also called symbolic data [5, 13], have been studied for many years. Much work focuses on clustering uncertain data and the key issue is how to compute the distance

Table 14 Classifier construction time (second) of data sets with multi categorical attributes uncertain

Data sets	uRule				
	U0	U1	U5	U10	U20
Acute	0.01	0.06	0.07	0.1	0.09
Contra	0.2	10.64	8.1	8.35	6.85
Derm	0.12	7.88	7.45	4.9	6.83
Hayes	0.01	0.18	0.11	0.12	0.09
Mushroom	6.7	170.6	170.4	180.1	180.2
Promote	0.01	0.71	0.81	0.89	0.03
Soybean	0.12	8.69	7.62	7.62	8.42
Teacher	0.04	1.56	2.01	1.38	1.79
Voting	0.04	1.22	1.34	1.75	0.73
Zoo	0.02	0.26	0.27	0.19	0.17
Data sets	DTU				
Acute	0.02	0.03	0.03	0.03	0.03
Contra	0.1	0.2	0.27	0.28	0.32
Derm	0.03	0.1	0.04	0.05	0.04
Hayes	0.05	0.02	0.03	0.01	0.02
Mushroom	0.06	1.37	1.3	1.39	3.87
Promote	0.03	0.05	0.03	0.03	0.05
Soybean	0.05	0.08	0.25	0.05	0.04
Teacher	0.02	0.05	0.04	0.05	0.06
Voting	0.03	0.02	0.03	0.04	0.07
Zoo	0.48	0.03	0.03	0.03	0.03

670 between uncertain objects. Different solutions have been proposed in literature [7,8,12,19,
671 21] There is also some research on identifying frequent item sets and association mining
672 from uncertain data sets. The support of itemsets and confidence of association rules have be
673 extended to integrate the existential probability of transactions and items, as in [1,3,10,40].

674 Classification is a well-studied area in data mining. Many classification algorithms have
675 been proposed in the literature, such as C4.5 [28], SWC4.4 and NBC4.4 [17], RIPPER [11],
676 MOGGP [23] and SVM classification trees [22]. In spite of the numerous classification algo-
677 rithms, building classification based on uncertain data has remained a great challenge. There
678 is some previous work performed on classifying uncertain data in various applications [4].
679 Most of them try to solve specific classification tasks instead of developing a general algo-
680 rithm for classifying uncertain data. Early work has also been performed on developing data
681 classification methods when data contain missing or noisy values [20,27]. In C4.5 [28], these
682 are handled by using fractional tuples.

683 Recently, Tsang et al. [34] and Qin et al. [24] independently developed decision tree algo-
684 rithms for uncertain data. Their common point is that both adopt the technique of fractional
685 tuple for splitting tuples into subsets when the domain of its PDF spans across the cut point.
686 The difference between them is that Qin's decision tree can handle both numerical and cate-
687 gorical data sets, while Tsang's decision trees can only deal with numerical data sets and they
688 focus on optimal strategies. To the best of our knowledge, this is the first attempt to develop

689 a rule-based classifier from uncertain data. A preliminary version of this paper appears in
690 [25]. A demo paper about rule-based classifier for uncertain data appears in [26].

691 Building a decision tree on tuples with numerical, point-valued data are computationally
692 demanding [14] and finding the best cut point is especially expensive. To improve efficiency,
693 many techniques have been proposed to reduce the number of candidate cut points [14–16].
694 Mining rules from tuples with numerical, point-valued data are also computationally inten-
695 sive. Since processing PDF is even more costly [34], our work offers a method to optimize
696 cut point selection for learning rules from both numerical and uncertain numerical data sets
697 and thus makes the process more efficient.

698 8 Conclusions and future work

699 In this paper, we propose a new rule-based classification algorithm for mining data with
700 uncertainty. Uncertain data occur in many emerging applications, including sensor databases,
701 spatial-temporal databases and medical or biology information systems. We integrate the
702 uncertain data model into the rule learning process. We generalize the definition of boundary
703 points to improve the efficiency of evaluating numerical attributes and uncertain numerical
704 attributes for cut point selection. We also propose a method to select cut points in possi-
705 ble worlds for the uncertain categorical data. We extend the traditional rule generation and
706 prediction process considering data with uncertain interval and x -tuple.

707 In many applications, it is highly expensive or even impossible to obtain accurate data sets;
708 however, it is much easier and cheaper to collect the uncertain ones. uRule is designed for
709 such scenarios. Experiments show that exploiting data uncertainty enables uRule to generate
710 rule sets with potentially higher accuracies on most uncertain numerical data sets. And the
711 uRule classifier is also more stable for mining uncertain categorical data sets. Thus, uRule
712 is a promising approach of mining uncertain data sets. This algorithm follows the new para-
713 digm of directly mining uncertain data sets. The avenues of future work include developing
714 uncertain data mining techniques for various applications, including sequential pattern min-
715 ing, association mining, spatial data mining and web mining, where data can be commonly
716 uncertain.

717 **Acknowledgments** The authors would like to thank the anonymous reviewers for their suggestions and Ben
718 Kao and Smith Tsang for the source code of their UDT algorithm.

719 References

- 720 1. Aggarwal C, Li Y, Wang J, Wang J (2009) Frequent pattern mining with uncertain Data. In: Proceedings
721 of SIGKDD'09 pp 29–38
- 722 2. <http://archive.ics.uci.edu/ml/datasets.html>
- 723 3. Bernecker T, Kriegel H, Renz M, Verhein F, Zfle A (2009) Probabilistic frequent itemset mining in
724 uncertain databases. In: Proceedings of SIGKDD'09 pp 119–128
- 725 4. Bi J, Zhang T (2004) Support vector classification with input data uncertainty. Adv Neural Inf Process
726 Syst 17:161–168
- 727 5. Bock HH, Diday E (2000) Analysis of symbolic data, exploratory methods for extracting statistical infor-
728 mation from complex data. Springer, Berlin
- 729 6. Bodner G, Schocke M, Rachbauer F, Seppi K, Peer S, Fierlinger A, Sununu T, Jaschke WR (2002) Dif-
730 ferentiation of malignant and benign musculoskeletal tumors: combined color and power doppler US and
731 spectral wave analysis. Radiology 223:410–416
- 732 7. Carvalho F, Brito P, Bock HH (2006) Dynamic clustering for interval data based on L2 distance. Comput
733 Stat 21(2):231–250

- 734 8. Chavent M, Carvalho F, Lechevallier Y, Verde R (2006) New clustering methods for interval data. *Comput*
735 *Stat* 21(2):211–229
- 736 9. Cheng R, Kalashnikov D, Prabhakar S (2003) Evaluating probabilistic queries over imprecise data. In:
737 *Proceedings of the ACM SIGMOD international conference on management of data* pp 551–562
- 738 10. Chui C, Kao B, Hung E (2007) Mining frequent itemsets from uncertain data. In: *Proceedings of the*
739 *PAKDD'07* pp 47–58
- 740 11. Cohen WW (1995) Fast effective rule induction. In: *Proceedings of the 12th international conference on*
741 *machine learning* pp 115–123
- 742 12. Cormode G, McGregor A (2008) Approximation algorithm for clustering uncertain data. In: *Proceedings*
743 *of the PODS 2008* pp 191–199
- 744 13. Diday E, Fraiture MN (2008) *Symbolic data analysis and the sodas software*. Wiley, London
- 745 14. Elomaa T, Rousu J (1999) General and efficient multisplitting of numerical attributes. *Mach Learn*
746 36(3):201–244
- 747 15. Elomaa T, Rousu J (2004) Efficient multisplitting revisited: optimapreserving elimination of partition
748 candidates. *Data Min Knowl Discov* 8(2):97–126
- 749 16. Fayyad UM, Irani KB (1992) On the handling of continuous-valued attributes in decision tree generation.
750 *Mach Learn* 8:87–102
- 751 17. Jiang L, Li C, Cai Z (2009) Learning decision tree for ranking. *Knowl Inf Syst* 20(1):123–135
- 752 18. Johnson NL, Kotz S, Balakrishnan N (1994) *Continuous univariate distributions*, 2nd edn. Wiley,
753 London
- 754 19. Kriegel H, Pfeifle M (2005) Density-based clustering of uncertain data. In: *Proceedings of the KDD'05*
755 *pp 672–677*
- 756 20. Lobo O, Numao M (1999) Ordered estimation of missing values. In: *Proceedings of PAKDD'99*
757 *pp 499–503*
- 758 21. Ngai WK, Kao B, Chui CK, Cheng R, Chau M, Yip KY (2006) Efficient clustering of uncertain data, In:
759 *Proceedings of ICDM'06*. pp 436–445
- 760 22. Pang S, Kasabov N (2009) Encoding and decoding the knowledge of association rules over SVM classifica-
761 tion trees. *Knowl Inf Syst* 19(1):79–105
- 762 23. Pappa G, Freitas A (2009) Evolving rule induction algorithms with multi-objective grammar-based
763 genetic programming. *Knowl Inf Syst* 19(3):283–309
- 764 24. Qin B, Xia Y, Li F (2009) DTU: a decision tree for uncertain data. In: *Proceedings of PAKDD'09*
765 *pp 4–15*
- 766 25. Qin B, Xia Y, Prbhakar S (2009) A rule-based classification algorithm for uncertain data. In: *Proceedings*
767 *of the workshop on management and mining Of uncertain data (MOUND)*
- 768 26. Qin B, Xia Y, Sathyesh R, Prabhakar S, Tu Y (2009) uRule: a rule-based classification system for uncertain
769 data. In: *Proceedings of ICDM'09 (Demo)*
- 770 27. Quinlan JR (1990) *Probabilistic decision trees in machine learning: an artificial intelligence approach*.
771 Morgan Kaufmann, San Francisco
- 772 28. Quinlan JR (1993) *C.45: programs for machine learning*. Morgan Kaufman, San Francisco
- 773 29. Quinlan JR (1995) MDL and categorical theories (Continued). In: *Proceedings of international conference*
774 *on machine Learning* pp 464–470
- 775 30. Quinlan JR, Cameron-Jones RM (1995) Induction of logic programs: FOIL and related systems. *New*
776 *Gener Comput* 13(3):287–312
- 777 31. Resconi G, Kovalerchuk B (2009) Agents' model of uncertainty. *Knowl Inf Syst* 18(2):213–229
- 778 32. Singh S, Mayfield C, Prabhakar S, Shah R, Hambrusch S (2007) Indexing categorical data with uncer-
779 tainty. In: *Proceedings of ICDE'07* pp 616–625
- 780 33. Tong W, Wei Y, Murga LF, Ondrechen MJ, Williams RJ (2009) Partial order optimum likelihood (POOL):
781 maximum likelihood prediction of protein active site residues using 3D structure and sequence Properties.
782 *PLoS Comput Biol* 5(1):410–416
- 783 34. Tsang S, Kao B, Yip KY, Ho WS, Lee SD (2010) Decision trees for uncertain data. *IEEE Trans Knowl*
784 *Eng*
- 785 35. Wei G, Wang H, Lin R (2010) Application of correlation coefficient to interval-valued intuitionistic fuzzy
786 multiple attribute decision-making with incomplete weight information. *Knowl Inf Syst*
- 787 36. Weiss SM, Indurkha N (1991) Reduced complexity rule induction. In: *Proceedings of IJCAI'91*
788 *pp 678–684*
- 789 37. Widom J (2005) Trio: a system for integrated management of data, accuracy, and lineage. In: *Proceedings*
790 *of ICDR'05* pp 262–276
- 791 38. Witten IH, Frank E (2005) *Data mining: practical machine learning tools and techniques*, 2nd edn.
792 Morgan Kaufman, San Francisco