2010 IEEE International Conference on Data Mining Workshops

uRule: A Rule-based Classification System for Uncertain Data

Biao Qin, Yuni Xia, Rakesh Sathyesh
Department of Computer Science
Indiana University - Purdue University Indianapolis, USA
{biaoqin,yxia,sathyesr}@cs.iupui.edu

Sunil Prabhakar
Department of Computer Science
Purdue University
sunil@cs.purdue.edu

Yicheng Tu
Department of Computer Science and Engineering
University of South Florida
ytu@cse.usf.edu

***Abstract*— Data uncertainty is common in real-world applications. Various reasons lead to data uncertainty, including imprecise measurements, network latency, outdated sources and sampling errors. These kinds of uncertainties have to be handled cautiously, or else the data mining results could be unreliable or wrong. In this demo,we will show uRule, a new rule-based classification and prediction system for uncertain data. This system uses new measures for generating, pruning and optimizing classification rules. These new measures are computed considering uncertain data intervals and probability distribution functions. Based on the new measures, the optimal splitting attributes and splitting values can be identified and used in classification rules. uRule can process uncertainty in both numerical and categorical data. It has satisfactory classification performance even when data is highly uncertain.**

I. Introduction

In many applications, data contains 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, and data staling. For example, in location based services, we assume moving objects are attached with locators and the location information is periodically updated and streamed to the control center. Based on the location information, various services can be provided including real time traffic-based routing, public transportation planning and accident prediction. In these applications, location data is typically inaccurate due to measurement error, locator energy constraint, network bandwidth constraint and network transmission latency. Similarly, in sensor network applications, measurement data such as temperature, humidity, pressure and moisture can be inaccurate.

Uncertainty can also arise in categorical data. For example, a tumor is typically classified as benign or malignant in cancer diagnosis and treatment. In practice, it is often extremely difficult to accurately classify a tumor at the initial diagnosis due to the experiment precision limitation. Lab results often have false positives and false negatives. Therefore, doctors may often diagnose tumors to be benign or malignant with certain probability or confidence [1].

Since data uncertainty is ubiquitous, it is important to develop data mining models for uncertain data. We study the problem of data classification with uncertainty and develop a rule-based classification algorithm for uncertain data. Rule-based data mining algorithms have a number of desirable properties. Rule sets are relatively easy for people to understand [2], and rule learning systems outperform decision tree learners on many problems [3], [4]. 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 [5], [6]. However, when data contains uncertainty - for example, when some numerical data is, instead of precise value, an interval with probability distribution function - these algorithms need to be extended to handle the uncertainty.

In the demo, we will show a new rule-based system for classifying and predicting both certain and uncertain data. We integrate the uncertain data model into the rule-based mining algorithm. We propose a new measure called probabilistic information gain for generating rules. We also extend the rule pruning measure for handling data uncertainty. We run uRule on uncertain datasets with both uniform and Gaussian distribution, and the results demonstrate its effectiveness.

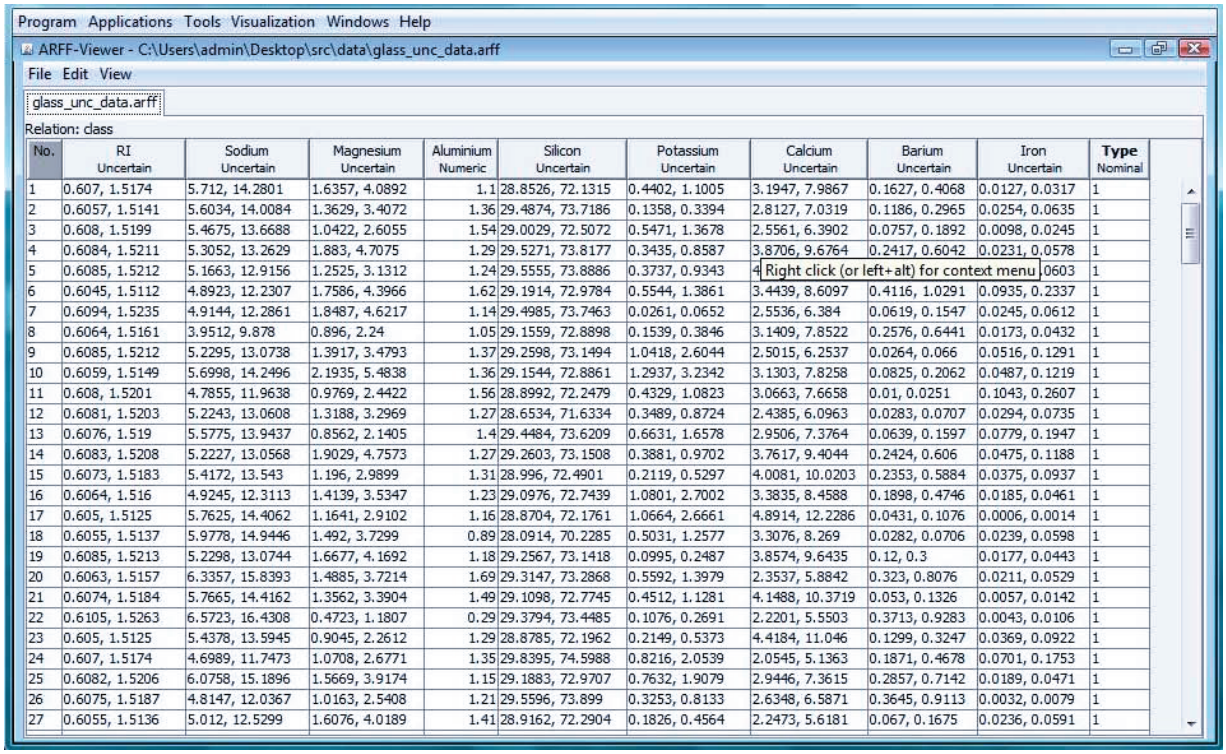
II. Data Uncertainty Model

TABLE I
Training Set for predicting borrowers who will default on loan payments

Rowid	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	110-120	No
2	No	Single	60-85	No
3	Yes	Married	110-145	No
4	No	Divorced	110-120	Yes
5	No	Married	50-80	No
6	Yes	Divorced	170-250	No
7	No	Single	85-100	Yes
8	No	Married	80-100	No
9	No	Single	120-145	Yes
10	No	Divorced	95-115	Yes
11	No	Divorced	80-95	No

We first discuss the uncertainty model for numerical and categorical attributes, which are the most common types of attributes encountered in data mining applications.

978-0-7695-4257-7/10 $26.00 © 2010 IEEE
DOI 10.1109/ICDMW.2010.73



No.	RI Uncertain	Sodium Uncertain	Magnesium Uncertain	Aluminium Numeric	Silicon Uncertain	Potassium Uncertain	Calcium Uncertain	Barium Uncertain	Iron Uncertain	Type Nominal
1	0.607, 1.5174	5.712, 14.2801	1.6357, 4.0892	1.1	28.8526, 72.1315	0.4402, 1.1005	3.1947, 7.9867	0.1627, 0.4068	0.0127, 0.0317	1
2	0.6057, 1.5141	5.6034, 14.0084	1.3629, 3.4072	1.36	29.4874, 73.7186	0.1358, 0.3394	2.8127, 7.0319	0.1186, 0.2965	0.0254, 0.0635	1
3	0.608, 1.5199	5.4675, 13.6688	1.0422, 2.6055	1.54	29.0029, 72.5072	0.5471, 1.3678	2.5561, 6.3902	0.0757, 0.1892	0.0098, 0.0245	1
4	0.6084, 1.5211	5.3052, 13.2629	1.883, 4.7075	1.29	29.5271, 73.8177	0.3435, 0.8587	3.8706, 9.6764	0.2417, 0.6042	0.0231, 0.0578	1
5	0.6085, 1.5212	5.1663, 12.9156	1.2525, 3.1312	1.24	29.5555, 73.8886	0.3737, 0.9343	4		0603	1
6	0.6045, 1.5112	4.8923, 12.2307	1.7586, 4.3966	1.62	29.1914, 72.9784	0.5544, 1.3861	3.4439, 8.6097	0.4116, 1.0291	0.0935, 0.2337	1
7	0.6094, 1.5235	4.9144, 12.2861	1.8487, 4.6217	1.14	29.4985, 73.7463	0.0261, 0.0652	2.5536, 6.384	0.0619, 0.1547	0.0245, 0.0612	1
8	0.6064, 1.5161	3.9512, 9.878	0.896, 2.24	1.05	29.1559, 72.8898	0.1539, 0.3846	3.1409, 7.8522	0.2576, 0.6441	0.0173, 0.0432	1
9	0.6085, 1.5212	5.2295, 13.0738	1.3917, 3.4793	1.37	29.2598, 73.1494	1.0418, 2.6044	2.5015, 6.2537	0.0264, 0.066	0.0516, 0.1291	1
10	0.6059, 1.5149	5.6998, 14.2496	2.1935, 5.4838	1.36	29.1544, 72.8861	1.2937, 3.2342	3.1303, 7.8258	0.0825, 0.2062	0.0487, 0.1219	1
11	0.608, 1.5201	4.7855, 11.9638	0.9769, 2.4422	1.56	28.8992, 72.2479	0.4329, 1.0823	3.0663, 7.6658	0.01, 0.0251	0.1043, 0.2607	1
12	0.6081, 1.5203	5.2243, 13.0608	1.3188, 3.2969	1.27	28.6534, 71.6334	0.3489, 0.8724	2.4385, 6.0963	0.0283, 0.0707	0.0294, 0.0735	1
13	0.6076, 1.519	5.5775, 13.9437	0.8562, 2.1405	1.4	29.4484, 73.6209	0.6631, 1.6578	2.9506, 7.3764	0.0639, 0.1597	0.0779, 0.1947	1
14	0.6083, 1.5208	5.2227, 13.0568	1.9029, 4.7573	1.27	29.2603, 73.1508	0.3881, 0.9702	3.7617, 9.4044	0.2424, 0.606	0.0475, 0.1188	1
15	0.6073, 1.5183	5.4172, 13.543	1.196, 2.9899	1.31	28.996, 72.4901	0.2119, 0.5297	4.0081, 10.0203	0.2353, 0.5884	0.0375, 0.0937	1
16	0.6064, 1.516	4.9245, 12.3113	1.4139, 3.5347	1.23	29.0976, 72.7439	1.0801, 2.7002	3.3835, 8.4588	0.1898, 0.4746	0.0185, 0.0461	1
17	0.605, 1.5125	5.7625, 14.4062	1.1641, 2.9102	1.16	28.8704, 72.1761	1.0664, 2.6661	4.8914, 12.2286	0.0431, 0.1076	0.0006, 0.0014	1
18	0.6055, 1.5137	5.9778, 14.9446	1.492, 3.7299	0.89	28.0914, 70.2285	0.5031, 1.2577	3.3076, 8.269	0.0282, 0.0706	0.0239, 0.0598	1
19	0.6085, 1.5213	5.2298, 13.0744	1.6677, 4.1692	1.18	29.2567, 73.1418	0.0995, 0.2487	3.8574, 9.6435	0.12, 0.3	0.0177, 0.0443	1
20	0.6063, 1.5157	6.3357, 15.8393	1.4885, 3.7214	1.69	29.3147, 73.2868	0.5592, 1.3979	2.3537, 5.8842	0.323, 0.8076	0.0211, 0.0529	1
21	0.6074, 1.5184	5.7665, 14.4162	1.3562, 3.3904	1.49	29.1098, 72.7745	0.4512, 1.1281	4.1488, 10.3719	0.053, 0.1326	0.0057, 0.0142	1
22	0.6105, 1.5263	6.5723, 16.4308	0.4723, 1.1807	0.29	29.3794, 73.4485	0.1076, 0.2691	2.2201, 5.5503	0.3713, 0.9283	0.0043, 0.0106	1
23	0.605, 1.5125	5.4378, 13.5945	0.9045, 2.2612	1.29	28.8785, 72.1962	0.2149, 0.5373	4.4184, 11.046	0.1299, 0.3247	0.0369, 0.0922	1
24	0.607, 1.5174	4.6989, 11.7473	1.0708, 2.6771	1.35	29.8395, 74.5988	0.8216, 2.0539	2.0545, 5.1363	0.1871, 0.4678	0.0701, 0.1753	1
25	0.6082, 1.5206	6.0758, 15.1896	1.5669, 3.9174	1.15	29.1883, 72.9707	0.7632, 1.9079	2.9446, 7.3615	0.2857, 0.7142	0.0189, 0.0471	1
26	0.6075, 1.5187	4.8147, 12.0367	1.0163, 2.5408	1.21	29.5596, 73.899	0.3253, 0.8133	2.6348, 6.5871	0.3645, 0.9113	0.0032, 0.0079	1
27	0.6055, 1.5136	5.012, 12.5299	1.6076, 4.0189	1.41	28.9162, 72.2904	0.1826, 0.4564	2.2473, 5.6181	0.067, 0.1675	0.0236, 0.0591	1

Fig. 1. Uncertain Data Viewer

When the value of a numerical type attribute is uncertain, the attribute is called an uncertain numerical attribute (UNA), denoted by A^{u_n}. The value of A^{u_n} is represented as a range or interval and the probability distribution function(pdf) over this range. Note that A^{u_n} is treated as a continuous random variable.

Table I shows an example of UNA. The data in this table are used to predict whether borrowers will default on loan payments. Among all the attributes, the Annual Income is a UNA, whose precise value is not available. We only know the range of the Annual Income of each person and the PDF $f(x)$ over that range. The probability distribution function of the UNA attribute Annual Income is assumed to be normal distribution.

Under the categorical uncertainty model, a dataset can have attributes that are allowed to take uncertain values. We call such an attribute an uncertain categorical attribute (UCA), denoted by A^{u_c}. Further, we use $A_i^{u_c}$ to denote the attribute value of the ith record in a database. The notion of UCA has been proposed in previous work such as [7].

$A_i^{u_c}$ takes values from a categorical domain Dom with cardinality $|Dom| = n$. For a certain dataset, the value of an attribute A is a single value d_k in Dom, $Pr(A = d_k) = 1$. In the case of an uncertain dataset, we record the information by a probability distribution over Dom instead of a single value. Let $Dom = \{d_1, d_2, \ldots, d_n\}$, then $A_j^{u_c}$ is the probability distribution $Pr(A_j^{u_c} = d_i)$ for all values of i in $\{1, \ldots, n\}$. Thus, $A_j^{u_c}$ can be represented by a probability vector $A_j^{u_c} = (p_1, p_2, \ldots, p_n)$ such that $\sum_{i=1}^{n} p_i = 1$.

TABLE II
A DATASET WITH AN UNCERTAIN CATEGORICAL ATTRIBUTE

Tuple	...	A_j	...	Class
1	...	(V_1:0.3; V_2:0.7)	...	1
2	...	(V_1:0.2; V_2:0.8)	...	0
3	...	(V_1:0.9; V_2:0.1)	...	1
4	...	(V_1:0.3; V_2:0.7)	...	1
5	...	(V_1:0.8; V_2:0.2)	...	1
6	...	(V_1:0.4; V_2:0.6)	...	1
7	...	(V_1:0.1; V_2:0.9)	...	0

Table II shows a dataset with a UCA attribute whose exact value is unobtainable. It may be either V_1 or V_2, each with associated probability. In practice, a dataset can have both uncertain numerical attributes and uncertain categorical attributes.

III. DEMONSTRATION

We implement uRule based on the open source data mining tool WEKA. We first extend the Arff Viewer in Weka so that it can display uncertain data in a proper tabular format. Figure 1 shows the use of the Viewer to see the uncertain intervals of the attributes of an uncertain numerical dataset.

A. Generating Classification Rules for Uncertain Data

To build a rule-based classifier, we need to extract a set of rules that show the relationships between the attributes

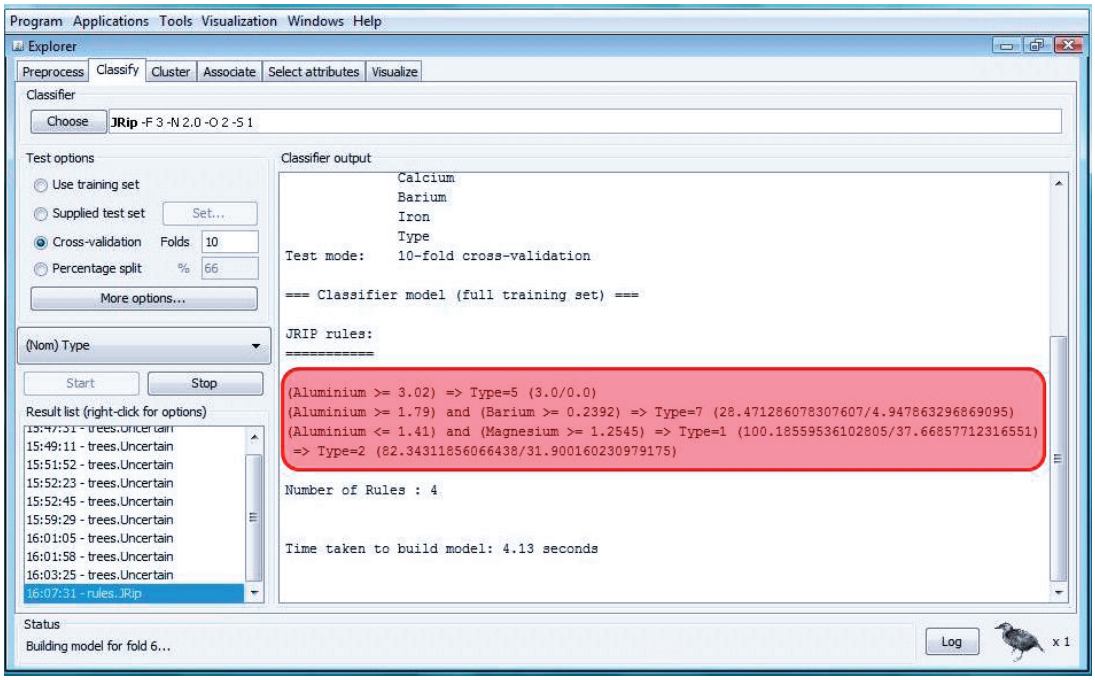


Fig. 2. Sample Classification Rules for Uncertain Data

of a dataset and the class label. Each classification rule is a form $R : Condition => y$. Here the $Condition$ is called the rule antecedent, which is a conjunction of the attribute test condition. y is called the rule consequent and it is the class label. A rule set can consist of multiple rules $R_S = \{R_1, R_2, ..., R_n\}$

A rule R covers an instance I_j if the attributes of the instance satisfy the condition of the rule. The **Coverage** of a rule is the number of instances that satisfy the antecedent of a rule. The **Accuracy** of a rule is the fraction of instances that satisfy both the antecedent and consequent of a rule, normalized by those satisfying the antecedent. Ideal rules should have both high coverage and high accuracy rates.

The uRule algorithm uses the sequential covering approach to extract rules from the data. The algorithm extracts the rules one class at a time for a data set. Let $(y_1, y_2, \ldots, y_n)$ be the ordered classes according to their frequencies, where y_1 is the least frequent class and y_n is the most frequent class. During the ith iteration, instances that belong to y_i are labeled as positive examples, while those that belong to other classes are labeled as negative examples. A rule is desirable if it covers most of the positive examples and none of the negative examples. Our uRule algorithm is based on the RIPPER algorithm [8], which was introduced by Cohen and considered to be one of the most commonly used rule-based algorithms in practice.

The rule learning procedure is the key function of the uRule algorithm. It generates the best rule for the current class, given the current set of uncertain training tuples. The rule learning procedure includes two phases: growing rules and pruning rules. After generating a rule, all the positive and negative examples covered by the rule are eliminated. The rule is then added into the rule set as long as it does not violate the stopping condition, which is based on the minimum description length (DL) principle. uRule also performs additional optimization steps to determine whether some of the existing rules in the rule set can be replaced by better alternative rules.

The basic strategy for growing rules is as follows:

1. It starts with an initial rule : {}-> y, where the left hand side is an empty set and the right hand side contains the target class. The rule has poor quality because it covers all the examples in the training set. New conjuncts will subsequently be added to improve the quality of the rule.

2. The probabilistic information gain is used as a measure to identify the best conjunct to be added into the rule antecedent. The algorithm selects the attribute and split point which has the highest probabilistic information gain and add them as an antecedent of the rule. The details of of computing probabilistic information gain for uncertain data can be found in our previous work [9] and will be shown in the demo.

3. If an instance is covered by the rule, A function splitUncertain() is invoked to find the part of the instance that is covered by the rule. Then, the part of the instance that is covered by the rule is removed from the dataset, and the rule growing process continues, until either all the data are covered or all the attributes have been used as antecedents.

Example: Refer to the dataset shown in Table I. Using uRule Algorithm, after rule growing and pruning, the following rule set is generated:
(annual_income $\geq$ 95) and (home_owner = No)=> class=Yes

(3.58/0.25)
$\{\} =>$ class=No (7.42/0.67)
The first is a regular rule, whose accuracy is around 93%, since it covers 3.58 positive instances and 0.25 negative instances. Please note that for uncertain data, a rule may partly cover instances, therefore, the number of positive and negative instances covered by a rule are no longer integers but real values. The second rule is a default rule. Like traditional rule-based classifier, uRule also generates a default rule when no more quality rule can be found. The default rule will be applied to instances which do not match any rule in the rule set. In the example, the default rule has an accuracy around 91%.

For the data shown in figure 1, The uncertain classification rules generated by uRule are shown in figure 2. The red shade area highlights all the uncertain classification rules.

B. Prediction with uRule

Once the rules are learned from a dataset, they can be used for predicting the class type of previously unseen data. Like a traditional rule classifier, each rule of uRule is in the form of "IF Conditions THEN Class = C_i". Because each instance I_i can be covered by several rules, a vector can be generated for each instance $(P(I_i, C) = (P(I_i, C_1), P(I_i, C_2), ..., P(I_i, C_n))^T)$, in which $P(I_i, C_j)$ denotes the probability for an instance to be in class C_j. We call this vector an Class Probability Vector (CPV).

As an uncertain data instance can be *partly* covered by a rule, we denote the degree an instance I covered by a rule R_j by $P(I, R_j)$. When $P(I, R_j)$ = 1, R_j fully covers instance I_i; when $P(I_i, R_j)$ = 0, R_j does not cover I_i; and when $0 < P(I_i, R_j) < 1$, R_j partially covers I_i.

An uncertain instance may be covered or partially covered by more than one rule. We allow a test instance to trigger all relevant rules. We use $w(I_i, R_k)$ to denote the weight of an instance I_i covered by the kth rule R_k. The weight of an instance I_i covered by different rules is as follows:

$$\begin{aligned} w(I_i, R_1) &= I_i.w * P(I_i, R_1) \\ w(I_i, R_2) &= (I_i.w - w(I_i, R_1)) * P(I_i, R_2) \\ &\cdots \\ w(I_i, R_n) &= (I_i.w - \sum_{k=1}^{n-1} w(I_i, R_k)) * P(I_i, R_n) \end{aligned}$$

Suppose an instance I_i is covered by m rules, then it class probability vector(CPV) $CPV(I_i, C)$ is computed as follows:

$$CPV(I_i, C) = \sum_{k=1}^{m} P(R_k, C) * w(I_i, R_k)$$

Where $P(R_k, C)$ is a vector $P(R_k, C) = (P(R_k, C_1), P(R_k, C_2), \ldots, P(R_k, C_n))^T$ and denotes the class distribution of the instances covered by rule R_k. $P(R_k, C_i)$ is computed as the fraction of the probabilistic cardinality of instances in class C_i covered by the rule over the overall probabilistic cardinality of instances covered by the rule.

After we compute the CPV for instance I_i, the instance will be predicted to be of the class which has the largest probability in the class probability vector. This prediction procedure is different from a traditional rule based classifier. When predicting the class type for an instance, a traditional rule based classifier such as RIPPER usually predicts with the first rule in the rule set that covers the instance. This is different for uncertain data. As an uncertain data instance can be fully or partially covered by multiple rules, the first rule in the rule set may not be the rule that best covers it. uRule uses all the relevant rules to compute the probability for the instance to be in each class and predicts the instance to be the class with the highest probability.

Example, suppose we have a test instance {No, Married, 90-110}, when applying the rules shown in Example 2 on the test instance, the class probability vector is

$$\begin{aligned} P(I, C) &= P(R_1, C) * w(I, R_1) + P(R_2, C) * w(I, R_2) \\ &= \begin{pmatrix} 0.93017 \\ 0.06983 \end{pmatrix} \times 0.75 + \begin{pmatrix} 0.0903 \\ 0.9097 \end{pmatrix} \times 0.25 \\ &= \begin{pmatrix} 0.6976 \\ 0.0524 \end{pmatrix} + \begin{pmatrix} 0.0226 \\ 0.2274 \end{pmatrix} \\ &= \begin{pmatrix} 0.7202 \\ 0.2798 \end{pmatrix} \end{aligned}$$

This shows that the instance is in class "Yes" with probability 0.7202 and in class "No" with probability 0.2798. Thus the test instance will be predicted to be in class "Yes".

During the demonstration, the audience will see how uRule generates uncertain classification rules in real time based on uncertain training datasets and how the uncertain classification rules are used for prediction. We will demonstrate uRule on uncertain datasets with both uniform and Gaussian distribution, and show that uRule has satisfactory classification and prediction accuracy on highly uncertain data.

REFERENCES

[1] G. Bodner, M. F. H. Schocke, F. Rachbauer, K. Seppi, S. Peer, A. Fierlinger, T. Sununu, and W. R. Jaschke, "Differentiation of malignant and benign musculoskeletal tumors: Combined color and power doppler us and spectral wave analysis," vol. 223, pp. 410–416, 2002.

[2] J. Catlett, "Megainduction: A test flight," in *ML*, 1991, pp. 596–599.

[3] G. Pagallo and D. Haussler, "Boolean feature discovery in empirical learning," *Machine Learning*, vol. 5, pp. 71–99, 1990.

[4] S. M. Weiss and N. Indurkhya, "Reduced complexity rule induction," in *IJCAI*, 1991, pp. 678–684.

[5] J. R. Quinlan, "Learning logical definitions from relations," *Machine Learning*, vol. 5, pp. 239–266, 1990.

[6] J. R. Quinlan and R. M. Cameron-Jones, "Induction of logic programs: Foil and related systems," *New Generation Comput.*, vol. 13, no. 3&4, pp. 287–312, 1995.

[7] S. Singh, C. Mayfield, S. Prabhakar, R. Shah, and S. Hambrusch, "Indexing categorical data with uncertainty," in *ICDE*, 2007, pp. 616–625.

[8] W. W. Cohen, "Fast effective rule induction," in *Proc. of the 12th Intl. Conf. on Machine Learning*, 1995, pp. 115–123.

[9] B. Qin, Y. Xia, S. Prabhakar, and Y. Tu, "A rule-based classification algorithm for uncertain data," in *the IEEE workshop on Management and Mining of Uncertain Data(MOUND), in conjunction with ICDE*, 2009.