Tối ưu quá trình học cây quyết định cho bài toán phân lớp theo cách tiếp cận khoảng mờ lớn nhất

Lê Văn Tường Lân[1], Nguyễn Mậu Hân[1], Nguyễn Công Hào[2]
[1]Khoa Công nghệ Thông tin, Trường Đại học Khoa học, Đại học Huế
[2]Trung tâm Công nghệ Thông tin, Đại học Huế
E-mail: lvtlan@yahoo.com, nmhan2009@gmail.com, nchao@hueuni.edu.vn
Tác giả liên hệ: Lê Văn Tường Lân
Ngày nhận: 07/07/2017, ngày sửa chữa: 20/09/2017, ngày duyệt đăng: 09/10/2017

Tóm tắt: Hiện nay, khai phá dữ liệu rõ không thể giải quyết tất cả các yêu cầu đặt ra và bài toán phân lớp cây quyết định mờ tất yếu đóng góp một vai trò quan trọng của bài toán khai phá dữ liệu mờ. Tuy vậy, việc học cây quyết định dựa vào định lượng ngữ nghĩa theo điểm vẫn còn một số hạn chế như vẫn xuất hiện nhiều sai số trong quá trình xử lý, cây kết quả thu được không thật sự linh hoạt. Bằng cách thức đối sánh theo khoảng mờ, cây thu được đã giảm thiểu sai số và linh hoạt trong dự đoán tuy nhiên số nút trên cây tăng nhanh nên không đơn giản với người dùng và mất thời gian duyệt cây khi dự đoán. Trong bài báo này, chúng tôi đề xuất khái niệm khoảng mờ lớn nhất để xây dựng phương pháp học quy nạp cây quyết định mờ HAC4.5*, nhằm thu được cây quyết định mờ đạt được tối thiểu về số nút trên cây nhưng có khả năng dự đoán cao.

Từ khóa: *Khai phá dữ liệu, cây quyết định, cây quyết định mờ, khoảng mờ lớn nhất, HAC4.5*.*

Title: **Fuzzy Decision Tree Learning for Classification based on Maximum Fuzziness Intervals**

Abstract: Nowadays, data mining based on precise logic cannot satisfy all requirements, so classification based on fuzzy decision trees is important due to the fuzziness nature of data mining. However, decision tree learning based on the quantitative methods of the hedge algebra still has some restrictions because of errors involved and the resulted tree not truly flexible. Using fuzziness interval matching, the resulted tree has reduced the number of errors and increase flexibility in prediction. However, the number of nodes in the resulted tree increases rapidly, causing difficulty to use and more time to produce prediction results. In this paper, a concept of maximum fuzziness interval is proposed in order to build an inductive learning method called "HAC4.5* fuzzy decision tree", resulting a fuzzy decision tree with the minimum number of nodes and highly accurate prediction.

Keywords: *Data mining, decision tree, fuzzy decision tree, maximum fuzziness intervals, HAC4.5*.*

I. ĐẶT VẤN ĐỀ

Thế giới thực thì vô hạn trong khi ngôn ngữ của chúng ta lại hữu hạn, vì thế sẽ xuất hiện những cụm từ không chính xác hoặc mơ hồ. Trong thế giới thực, các kho dữ liệu nghiệp vụ được lưu trữ mờ là tất yếu, vì thế bài toán phân lớp cây quyết định rõ không thể giải quyết tất cả các yêu cầu đặt ra và bài toán phân lớp dữ liệu cây quyết định mờ là vấn đề tất yếu của khai phá dữ liệu [1–12].

Cho một tập các mẫu dữ liệu $V = U \times Y$, trong đó $U = \{u_i | i = 1, \ldots, N\}$ là tập dữ liệu, $Y = \{y_1, \ldots, y_n\}$ là tập các nhãn của các lớp, $u_i \in D$ là dữ liệu thứ i với $D = A_1 \times \cdots \times A_m$ là tích Đề-các của các miền của m thuộc tính tương ứng, n là số lớp và N là số mẫu dữ liệu, để ý rằng $U \subset D$. Mỗi dữ liệu $u_i \in U$ thuộc một lớp $y_i \in Y$ tương ứng tạo thành từng cặp $(u_i, y_i) \in V$. Giải bài toán phân lớp dựa trên mô hình cây quyết định chính là xây dựng một cây quyết định để phân lớp, ký hiệu S, là một ánh xạ từ tập dữ liệu vào tập nhãn:

$$S : D \rightarrow Y. \quad (1)$$

Cây quyết định biểu diễn cho tri thức về bài toán, nó không chỉ phản ánh đúng với tập dữ liệu mẫu mà còn phải có khả năng dự đoán và cung cấp giúp cho người dùng phán đoán, ra quyết định đối với đối tượng trong tương lai mà nhãn lớp của nó chưa được xác định từ tập dữ liệu chưa biết. Với bài toán phân lớp cây quyết định được nêu ở (1),

nếu tồn tại một thuộc tính mờ A_j trong số các thuộc tính của D thì ta nói (1) là bài toán phân lớp cây quyết định mờ.

Như vậy, mô hình cây quyết định S phải đạt các mục tiêu như hiệu quả phân lớp cao, tức là sai số phân lớp cho các dữ liệu ít nhất có thể, cây có ít nút nhưng có khả năng dự đoán cao, không xảy ra tình trạng quá khớp. Mục tiêu về hiệu quả phân lớp nhằm đáp ứng tính đúng đắn đối với tập dữ liệu mẫu được cho của bài toán, còn mục tiêu sau được đưa ra với mong muốn mô hình cây quyết định nhận được phải đơn giản và dễ hiểu đối với người dùng.

Gọi $f_h(S)$ là hàm đánh giá tính hiệu quả của quá trình phân lớp, $f_n(S)$ là hàm đánh giá tính đơn giản của cây và dễ hiểu đối với người dùng, mục tiêu của bài toán là

$$f_h(S) \to \max \text{ và } f_n(S) \to \min. \quad (2)$$

Hai mục tiêu ở (2) khó có thể đạt được đồng thời. Khi số nút của cây giảm đồng nghĩa với lượng tri thức về bài toán giảm thì nguy cơ phân lớp sai tăng lên, nhưng khi có quá nhiều nút cũng có thể gây ra sự quá khớp thông tin trong quá trình phân lớp. Bên cạnh đó, sự phân chia tại mỗi nút ảnh hưởng đến tính phổ quát hay cá thể tại nút đó. Nếu sự phân chia tại một nút là nhỏ sẽ làm tăng tính phổ quát và ngược lại nếu sự phân chia lớn sẽ làm tăng tính cá thể của nút đó. Tính phổ quát của nút trên cây sẽ làm tăng khả năng dự đoán nhưng nguy cơ gây sai số lớn, trong khi tính cá thể giảm khả năng dự đoán nhưng lại tăng tính đúng đắn nhưng nó cũng là nguyên nhân của tình trạng quá khớp trên cây. Các phương pháp giải quyết bài toán mô hình cây quyết định đều phải thỏa hiệp giữa các mục tiêu này để đạt được kết quả cuối cùng.

Hiện có một số cách tiếp cận như sau. Trước hết là cách tiếp cận giá trị ngôn ngữ cho tập mờ để xây dựng cây quyết định mờ. Các tác giả trong [13–15] đã xác định các giá trị ngôn ngữ cho tập dữ liệu mờ và xây dựng cây quyết định ngôn ngữ (LDT: Linguistic Decision Tree) bằng cách sử dụng tư tưởng của thuật toán ID3 của cây quyết định rõ cho các nút ứng với các thuộc tính ngôn ngữ (LID3).

Tuy nhiên, cách làm này sẽ làm phát sinh cây đa phân, có sự phân chia theo chiều ngang lớn tại các nút ngôn ngữ khi tập giá trị ngôn ngữ của thuộc tính mờ lớn (Hình 1), nên dễ dẫn đến tình trạng quá khớp. Thêm vào đó, tại nút này, ta không thể sử dụng cách phân chia nhị phân của thuật toán C4.5 vì không có thứ tự giữa các giá trị ngôn ngữ. Hơn nữa, với các giá trị rõ trong miền thuộc tính mờ của tập dữ liệu huấn luyện, một đoạn con các giá trị rõ sẽ được ánh xạ bằng một giá trị ngôn ngữ nên có sự sai lệch lớn.

Cách tiếp cận mờ thứ hai là theo đại số gia tử (ĐSGT) do N. C. Ho và W. Wechler khởi xướng từ 1990. Cách này có nhiều ưu điểm vì theo cách tiếp cận này, mỗi giá trị ngôn ngữ của một biến ngôn ngữ nằm trong một cấu trúc đại số nên ta có thể đối sánh giữa các giá trị ngôn ngữ.

Hình 1. Điểm phân chia đa phân theo giá trị ngôn ngữ tại thuộc tính mờ.

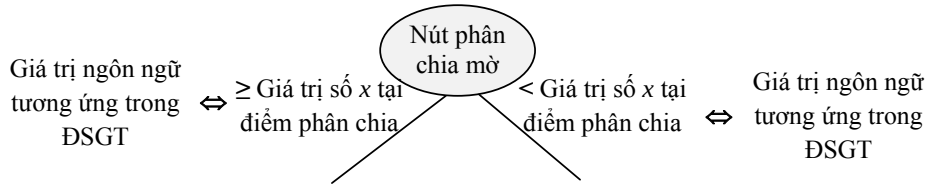
Theo cách tiếp cận ĐSGT, chúng ta có thể thuần nhất các trường mà dữ liệu của nó bao gồm cả dữ liệu rõ và mờ. Các tác giả trong [16–20] đã chỉ ra phương pháp định lượng ngữ nghĩa theo điểm nhằm thuần nhất dữ liệu về các giá trị số hay giá trị ngôn ngữ và cách thức truy vấn dữ liệu trên thuộc tính này. Từ đó, chúng ta có thể học phân lớp trên tập mẫu thuần nhất đó.

Bài toán xây dựng cây quyết định mờ lúc này có thể sử dụng các thuật toán xây dựng cây quyết định rõ như C4.5, SLIQ, v.v. [7, 8, 10, 12]. Các nút phân chia nhị phân được tính theo dựa theo điểm phân chia với các giá trị ngôn ngữ đã có thứ tự và hoàn toàn xác định tương ứng với một giá trị số trong ĐSGT đã xây dựng.

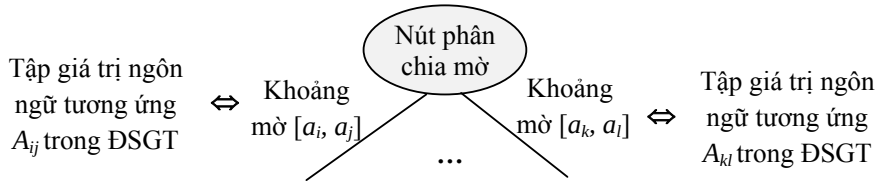
Tuy vậy, cách thuần nhất dựa trên phương pháp định lượng ngữ nghĩa theo điểm này vẫn xuất hiện nhiều sai số vì một đoạn con các giá trị rõ đã có sẽ được quy về một điểm tức là một giá trị ngôn ngữ tương ứng, điều này cũng làm xuất hiện các giá trị gần nhau có thể được phân hoạch ở hai đoạn con khác nhau nên kết quả phân lớp khác nhau. Bên cạnh đó, cây kết quả cũng khó đưa ra dự đoán trong các trường hợp cần dự đoán mà tại đó có sự đan xen ở điểm phân chia mờ, ví dụ ta cần dự đoán cho trường hợp đoạn con $[x_1, x_2]$, với $x_1 < x < x_2$ tại nút phân chia mờ ở Hình 2.

Cách tiếp cận thứ ba là bằng cách thức đối sánh theo khoảng mờ dựa trên ĐSGT, cây kết quả thu được đã giảm thiểu sai số và linh hoạt nhờ chúng ta vẫn giữ nguyên miền giá trị rõ trong khi vẫn đối sánh được với các giá trị mờ trong tập mẫu cho quá trình huấn luyện [20], như Hình 3. Tuy vậy, số nút trên cây kết quả có khả năng tăng cao nên không đơn giản với người dùng và mất thời gian duyệt cây khi dự đoán. Do vậy, hàm mục tiêu đã nêu ở (2) có thể không đạt được.

Trong bài báo này, chúng tôi đề xuất một phương pháp học cây quyết định mờ khi tập mẫu huấn luyện không thuần nhất giá trị dựa trên khái niệm khoảng mờ lớn nhất, trong khi giữ nguyên các ưu điểm của phương pháp đối sánh theo khoảng mờ nhưng tối ưu về số nút trên cây kết quả, nhằm thu được cây quyết định có đơn giản và hiệu quả.



Hình 2. Điểm phân chia nhị phân theo giá trị ngôn ngữ hoặc giá trị số tại thuộc tính mờ, dựa trên phương pháp định lượng ngữ nghĩa theo điểm trong ĐSGT.



Hình 3. Điểm phân chia theo khoảng mờ tại thuộc tính mờ, dựa trên phương pháp định lượng ngữ nghĩa theo khoảng mờ trong ĐSGT.

II. KHOẢNG MỜ VÀ KHÁI NIỆM KHOẢNG MỜ LỚN NHẤT

1. Khoảng mờ và đối sánh dựa trên khoảng mờ

Cho một ĐSGT $\underline{X} = (X, G, H, \leq)$, với $G = \{c^+, c^-\}$, trong đó c^+ và c^- tương ứng là phần tử sinh dương và âm; X là tập nền; $H = H^+ \cup H^-$ với $H^- = \{h_{-p}, h_{-p+1}, \ldots, h_{-1}\}$ và $H^+ = h_1, \ldots, h_q$, $h_{-1} > h_{-2} > \cdots > h_{-p}$ và $h_1 < \cdots < h_q$.

Định nghĩa 1. Khoảng mờ của một phần tử x là một đoạn con của $[0, 1]$, có độ dài đúng bằng độ đo tính mờ $fm(x)$, ký hiệu là $I(x)$, được xác định bằng cách quy nạp theo độ dài của x như sau [17]:

i) Với độ dài x bằng 1 ($l(x) = 1$), tức là $x \in \{c^+, c^-\}$, $I_{fm}(c^-)$ và $I_{fm}(c^+)$ là các khoảng con và tạo thành một phân hoạch của $[0, 1]$, thỏa $I_{fm}(c^-) \leq I_{fm}(c^+)$. Tức là với mọi $u \in I_{fm}(c^-)$ và mọi $v \in I_{fm}(c^+)$: $u \leq v$. Điều này hoàn toàn phù hợp với thứ tự ngữ nghĩa của c^- và c^+.
Ký hiệu độ dài của $I_{fm}(x)$ là $\left|I_{fm}(x)\right|$, ta có $\left|I_{fm}(c^-)\right| = I_{fm}(c^-)$ và $\left|I_{fm}(c^+)\right| = I_{fm}(c^+)$;

ii) Giả sử với mọi $x \in X$ có độ dài bằng k ($l(x) = k$), có khoảng mờ là $I_{fm}(x)$ và $\left|I_{fm}(x)\right| = fm(x)$. Các khoảng mờ của $y = h_i x$, với mọi $i \in [-p, -p + 1, \ldots, -1, 1, 2, \ldots, q]$, lúc này $l(y) = k + 1$, là tập $\{I_{fm}(h_i x)\}$ thoả mãn một phân hoạch của $I_{fm}(x)$, $\left|I_{fm}(h_i x)\right| = I_{fm}(h_i x)$ và có thứ tự tuyến tính tương ứng với thứ tự của tập $\{h_{-p}x, h_{-p+1}x, \ldots, h_q x\}$.
Khi $l(x) = k$, ta ký hiệu $I(x)$ thay cho $I_{fm}(x)$, $X_k = \{\forall x \in X | l(x) = k\}$ là tập các phần tử trong X có độ dài đúng bằng k, $I_k = \{I_k(x) | x \in X_k\}$ là tập tất cả các khoảng mờ mức k.

Định nghĩa 2. Hai khoảng mờ được gọi là bằng nhau, ký hiệu $I(x) = I(y)$ khi chúng được xác định bởi cùng một giá trị ($x = y$), tức là ta có $I_L(x) = I_L(y)$ và $I_R(x) = I_R(y)$. Ngược lại ta gọi chúng là hai khoảng mờ khác nhau và ký hiệu là $I(x) \neq I(y)$. Trong đó, ký hiệu $I_L(x)$ và $I_R(x)$ là điểm mút trái và phải của khoảng mờ $I(x)$.

Định nghĩa 3. Cho 2 khoảng rõ khác nhau $[a_1, b_1]$ và $[a_2, b_2]$ tương ứng với các khoảng mờ $[I_{a_1}, I_{b_1}]$, $[I_{a_2}, I_{b_2}] \subseteq [0, 1]$. Ta nói khoảng $[a_1, b_1]$ đứng trước $[a_2, b_2]$ hay $[a_2, b_2]$ đứng sau $[a_1, b_1]$, viết $[a_1, b_1] < [a_2, b_2]$ hay $[I_{a_1}, I_{b_1}] < [I_{a_2}, I_{b_2}]$ nếu:

i) $b_2 > b_1$ tức $I_{b_2} > I_{b_1}$,
ii) Nếu $I_{b_2} = I_{b_1}$ tức $b_2 = b_1$ thì $I_{a_2} > I_{a_1}$ tức $a_2 > a_1$,

và lúc này ta nói dãy $[a_1, b_1]$, $[a_2, b_2]$ là dãy có 2 khoảng có quan hệ thứ tự trước sau.

Định lý 1. *Cho k khoảng khác nhau từng đôi một* $[a_1, b_1]$, $[a_2, b_2]$,..., $[a_k, b_k]$, *ta luôn sắp xếp để được một dãy có k khoảng với quan hệ thứ tự trước sau.*

Thật vậy, với k khoảng khác nhau từng đôi một $[a_1, b_1]$, $[a_2, b_2]$,..., $[a_k, b_k]$, ta luôn tìm được khoảng trước nhất của dãy là $[a_i, b_i]$, với $a_i = \min(a_1, a_2, \ldots, a_n)$. Nếu có nhiều khoảng $[a_j, b_j]$, $i = 1 \ldots k$ mà $a_j = a_i$ thì chọn ta khoảng $[a_i, b_i]$ là khoảng có b_i bé nhất trong các giá trị b_j. Việc chọn b_i luôn tìm được duy nhất vì các khoảng đã cho khác nhau từng đôi một nên nếu $a_i = a_j$ thì $b_i \neq b_j$ (Định nghĩa 2).

Sau khi tìm được khoảng trước nhất của dãy là $[a_i, b_i]$, ta tiếp tục tìm khoảng thứ 2 và cứ thế tiếp tục. Sau k lần tìm và sắp xếp ta có được dãy gồm k khoảng mà các phần tử của dãy đã được sắp xếp theo quan hệ thứ tự trước sau.

2. Khoảng mờ lớn nhất và phương pháp tính khoảng mờ lớn nhất

Trong ĐSGT, một hạng từ có thể mang ngữ nghĩa của một hạng từ khác tức hạng từ được dùng để sinh ra nó. Chẳng hạn "*very old*" mang ngữ nghĩa của "*old*", "*very little old*" cũng mang ngữ nghĩa của "*old*". Vậy, hai hạng từ "*very old*" và "*very little old*" đều có quan hệ kế thừa ngữ nghĩa (hay quan hệ kế thừa) của "*old*", ta gọi "*very old*" và "*very little old*" có quan hệ kế thừa ngữ nghĩa.

Định nghĩa 4. Một ĐSGT $\underline{X} = (X, G, H, \leq)$, với mọi $x, y \in X$, được gọi là có quan hệ kế thừa ngữ nghĩa với nhau và được ký hiệu $\sim(x, y)$ nếu tồn tại $z \in X$ sao cho $x = h_{i_n} \cdots h_{i_1} z = h_{j_m} \cdots h_{j_1} z$.

Mệnh đề 1. *Với mọi $x, y \in X$ xác định hai khoảng mờ mức k và mức l lần lượt là $I_k(x)$ và $I_l(y)$, chúng hoặc không có quan hệ kế thừa, hoặc có quan hệ kế thừa với nhau nếu tồn tại $z \in X, |z| = v$, $v \leq \min(l, k)$, $I_L(z) \leq I_L(y)$, $I_R(z) \geq I_R(y)$, và $I_L(z) \leq I_L(x)$, $I_R(z) \geq I_R(x)$ hay $I_v(z) \supseteq I_k(x)$ và $I_v(z) \supseteq I_l(y)$, tức là x, y được sinh ra từ z.*

Chứng minh. Mệnh đề này dễ dàng suy ra từ tính phân hoạch các khoảng mờ.

Khi x và y có quan hệ kế thừa ngữ nghĩa, ta nói rằng khoảng tính mờ $I_v(z)$ bao hàm hai khoảng tính mờ $I_k(x)$ và $I_l(y)$, hay z bao hàm ngữ nghĩa của x và y và ta ký hiệu $z = \sim(x, y)$. Theo định nghĩa, rõ ràng hai hạng từ x, y có quan hệ ngữ nghĩa nếu chúng có dạng $x = h_{i_n} \cdots h_{i_1} z$, $y = h_{j_m} \cdots h_{j_1} z$. Nếu $h_{i_1} = h_{j_1} = h'_1$, $h_{i_2} = h_{j_2} = h'_2 \ldots$ thì ta có $z = h'_1 c$ hoặc $z = h'_1 h'_2 c$ đều bao hàm ngữ nghĩa của x và y. Tuy nhiên, thực tế sử dụng z thay thế cho cả x và y có thể làm mất ngữ nghĩa của chúng và rõ ràng, trên thực tế chúng ta cần phải xác định z sao cho ngữ nghĩa càng gần với x, y càng tốt. □

Định nghĩa 5. Cho một ĐSGT $\underline{X} = (X, G, H, \leq)$, với x, y, $z \in X$, $z = \sim(x, y)$. Nếu $\exists z_1 \in X$, $z_1 = \sim(x, y)$ và $\text{len}(z) \geq \text{len}(z_1)$ thì ta nói z có ngữ nghĩa gần với x, y nhất, hay khoảng mờ z có độ dài lớn nhất và được ký hiệu $z = \sim_{\max}(x, y)$.

Định nghĩa 6. Cho một ĐSGT $\underline{X} = (X, G, H, \leq)$, với $\forall x$, $y \in X$ và $\sim(x, y)$. Mức độ gần nhau của x và y theo quan hệ kế thừa ngữ nghĩa, ký hiệu là $\text{sim}(x, y)$, được định nghĩa như sau:

$$\text{sim}(x, y) = \frac{m}{\max(k, l)} (1 - |v(x) - v(y)|),$$

trong đó $k = \text{len}(x)$, $l = \text{len}(y)$ và $m = \text{len}(z)$ với $z = \sim_{\max}(x, y)$.

Mệnh đề 2. *Cho một ĐSGT $\underline{X} = (X, G, H, \leq)$, với mọi $x, y \in X$, ta có các tính chất về mức độ gần nhau của các hạng từ như sau:*

i) *Hàm $sim(x, y)$ có tính chất đối xứng, tức là $sim(x, y) = sim(y, x)$;*
ii) *x, y không có quan hệ kế thừa ngữ nghĩa khi và chỉ khi $sim(x, y) = 0$;*
iii) *$sim(x, y) = 1 \Leftrightarrow x = y$;*
iv) *$\forall x, y, z \in X_k$, $x \leq y \leq z \Rightarrow sim(x, z) \leq sim(x, y)$ và $sim(x, z) \leq sim(y, z)$.*

Chứng minh. i) Hiển nhiên theo định nghĩa.

ii) Theo định nghĩa, ta có $m = 0$ khi và chỉ khi x, y không có quan hệ kế thừa ngữ nghĩa. Vậy nếu x và y không có quan hệ kế thừa ngữ nghĩa thì $m = 0$ suy ra $\text{sim}(x, y) = 0$. Ngược lại, khi $\text{sim}(x, y) = 0$ thì $m = 0$ hoặc $(1 - |v(x) - v(y)|) = 0$. Khi $m = 0$ tức x, y không có quan hệ kế thừa ngữ nghĩa. Trường hợp $(1 - |v(x) - v(y)|) = 0$ thì $|v(x) - v(y)| = 1$ suy ra $x = 0$, $y = 1$ hoặc $x = 1$, $y = 0$ nên x, y không có quan hệ kế thừa ngữ nghĩa.

iii) Do $m \leq \max(k, l)$ và $0 \leq v(x), v(y) \leq 1$ nên: $\text{sim}(x, y) = 1 \Leftrightarrow m = \max(k, l)$ và $|v(x) - v(y)| = 0 \Leftrightarrow x = y$.

iv) Theo giả thiết $x \leq y \leq z \Rightarrow v(x) \leq v(y) \leq v(z) \Rightarrow 1 - |v(x) - v(z)| \leq 1 - |v(x) - v(y)|$ và $1 - |v(x) - v(z)| \leq 1 - |v(y) - v(z)|$. Mặt khác, cũng theo giả thiết $x, y, z \in X_k \Rightarrow \text{len}(x) = \text{len}(y) = \text{len}(z) = k$. Vậy đặt $w_1 = \sim_{\max}(x, y)$, $w_2 = \sim_{\max}(y, z)$, $w_3 = \sim_{\max}(x, z)$, theo qui tắc sinh các hạng từ của ĐSGT với giả thiết $x \leq y \leq z$ nên $\text{len}(w_1) \geq \text{len}(w_3)$ và $\text{len}(w_2) \geq \text{len}(w_3)$. Vậy theo Định nghĩa 6 ta có $\text{sim}(x, y) \leq \text{sim}(x, z)$ và $\text{sim}(y, z) \leq \text{sim}(x, z)$. □

Định nghĩa 7 (Tính kề nhau của các khoảng mờ)**.** Cho một ĐSGT $\underline{X} = (X, G, H, \leq)$, hai khoảng tính mờ $I(x)$ và $I(y)$ được gọi là kề nhau nếu chúng có một điểm mút chung, tức là $I_L(x) = I_R(y)$ hoặc $I_R(x) = I_L(y)$.

Giả sử $I_R(x) = I_L(y)$ ta có khoảng mờ $I(x)$ nằm kề trái của khoảng mờ $I(y)$ và theo Định nghĩa 3, ta có $I(x) < I(y)$.

Như vậy với hai khoảng mờ x và y, ta có thể sử dụng khoảng mờ z đại diện mà không làm mất nhiều ngữ nghĩa của x và y bằng cách dựa vào hàm $\text{sim}(x, y)$ và việc sử dụng z thay thế cho x và y được xem như phép kết nhập khoảng mờ x, y thành khoảng mờ z.

Để tìm khoảng mờ lớn nhất của hai khoảng mờ cho trước, chúng ta sử dụng Thuật toán 1.

Thuật toán 1: Thuật toán tính khoảng mờ lớn nhất của hai khoảng mờ cho trước

Input:
ĐSGT $\underline{X} = (X, G, H, \leq)$ và $x, y \in X$
Output: $z \in X$, $z = \sim_{\max}(x, y)$.
Method:
$k = \text{len}(x)$;
$l = \text{len}(y)$;
$v = \min(k, l)$;
while $v > 0$ **do**
  begin
    if $\exists z \in X$, $|z| = v$ and $I_k(x) \subseteq I_v(z)$
      and $I_l(y) \subseteq I_v(z)$ **then**
      return $I_v(z)$
    else
      $v = v - 1$;
    end if
  end
end while
return NULL;

III. THUẬT TOÁN HAC4.5* SỬ DỤNG KHÁI NIỆM KHOẢNG MỜ LỚN NHẤT TRONG QUÁ TRÌNH HUẤN LUYỆN CÂY QUYẾT ĐỊNH

1. Ý tưởng đề xuất thuật toán

Ở đây, ta cũng dựa vào thuật toán C4.5 [10, 12] và sử dụng cách thức đối sánh khoảng mờ tại các thuộc tính mờ của tập huấn luyện [20]. Tuy nhiên, việc tính lợi ích thông tin cho thuộc tính mờ theo khoảng mờ có thể xảy ra trường hợp đó là các khoảng mờ khác nhau nhưng lại có chung kết quả dự đoán, điều này làm cho mô hình cây thu được có nhiều nút và khi chúng ta sử dụng mức k có giá trị càng lớn, để tăng tính chính xác, thì vấn đề này càng có khả năng xảy ra. Hơn nữa, tại ngưỡng được chọn $\text{Th}_i^{HA} = [I_{a_i}, I_{b_i}]$, việc chia tập huấn luyện D thành các tập: $D_1 = \left\{\forall [I_{a_j}, I_{b_j}] \middle| [I_{a_j}, I_{b_j}] < \text{Th}_i^{HA}\right\}$ và $D_2 = \left\{\forall [I_{a_j}, I_{b_j}] \middle| [I_{a_j}, I_{b_j}] > \text{Th}_i^{HA}\right\}$ không phải lúc nào cũng là lựa chọn tốt nhất vì có thể thể xảy ra trường hợp một đoạn con của D_1 hoặc D_2 mới có lợi ích thông tin lớn nhất.

Do thuộc tính mờ A của tập huấn luyện đã được đã được phân hoạch theo khoảng mờ là một đoạn con của $[0, 1]$ và miền dữ liệu của nó là một tập được sắp xếp thứ tự tuyến tính theo quan hệ trước sau nên các khoảng mờ của chúng có tính kề trái và kề phải. Như vậy với hai khoảng mờ x và y nếu chúng có chung lớp dự đoán, ta có thể sử dụng khoảng mờ $z = \sim_{\max}(x, y)$ thay thế mà không làm thay đổi ngữ nghĩa của x và y trong quá trình học phân lớp. Việc sử dụng phép kết nhập z thay thế cho x và y được thực hiện cho tất cả các khoảng mờ của thuộc tính mờ A.

Như vậy, thuộc tính mờ A của tập huấn luyện sau khi đã phân hoạch theo khoảng mờ theo khoảng mờ lớn nhất có k khoảng khác nhau và đã được sắp xếp theo thứ tự trước sau là: $[I_{a_1}, I_{b_1}] < [I_{a_2}, I_{b_2}] < \cdots < [I_{a_k}, I_{b_k}]$ và ta tiếp tục việc tìm ngưỡng cho phép tách dựa theo tỷ lệ lợi ích thông tin của các ngưỡng tại nút đó.

2. Thuật toán đề xuất

1) Tính lợi ích thông tin cho các khoảng mờ tại thuộc tính mờ:
Do thuộc tính mờ của tập huấn luyện đã được đã được phân hoạch theo khoảng mờ là một đoạn con của $[0, 1]$ và miền dữ liệu của nó là một tập được sắp xếp thứ tự tuyến tính theo quan hệ trước sau. Ta có thể so sánh để phân ngưỡng cho tập giá trị này tại một khoảng bất kỳ $I(x) = [I_a, I_b] \subseteq [0, 1]$ được chọn, tương tự như các giá trị số liên tục trong C4.5.

Việc tìm ngưỡng để cho phép tách cũng dựa theo tỷ lệ lợi ích thông tin của các ngưỡng trong tập D tại nút đó. Tỷ lệ lợi ích thông tin của các ngưỡng đối với thuộc tính A là thuộc tính số trong tập D tại nút đó.

Giả sử thuộc tính A là thuộc tính mờ đã phân hoạch theo khoảng mờ và có k khoảng khác nhau đã được sắp xếp theo thứ tự trước sau là: $[I_{a_1}, I_{b_1}] < [I_{a_2}, I_{b_2}] < \cdots < [I_{a_k}, I_{b_k}]$.

Ta có k ngưỡng được tính: $\text{Th}_i^{HA} = [I_{a_i}, I_{b_i}]$, (với $1 \leq i < k$). Tại mỗi ngưỡng được chọn Th_i^{HA}, tập dữ liệu D còn lại tại nút này được chia làm các tập

$$D_1 = \left\{\forall [I_{a_j}, I_{b_j}] \middle| [I_{a_j}, I_{b_j}] < \text{Th}_i^{HA}\right\},$$
$$D_2 = \left\{\forall [I_{a_j}, I_{b_j}] \middle| [I_{a_j}, I_{b_j}] > \text{Th}_i^{HA}\right\}.$$

Lúc này ta có

$$\text{Gain}^{HA}\left(D, \text{Th}_i^{HA}\right) = \text{Entropy}(D) - \frac{|D_1|}{|D|} \times \text{Entropy}(D_1) - \frac{|D_2|}{|D|} \times \text{Entropy}(D_2),$$

$$\text{SplitInfo}^{HA}\left(D, \text{Th}_i^{HA}\right) = -\frac{|D_1|}{|D|} \times \log_2 \frac{|D_1|}{|D|} - \frac{|D_2|}{|D|} \times \log_2 \frac{|D_2|}{|D|},$$

$$\text{GainRatio}^{HA}\left(D, \text{Th}_i^{HA}\right) = \frac{\text{Gain}^{HA}\left(D, \text{Th}_i^{HA}\right)}{\text{SplitInfo}^{HA}\left(D, \text{Th}_i^{HA}\right)}.$$

Trên cơ sở tính toán tỷ lệ lợi ích thông tin cho các ngưỡng, ngưỡng nào có tỷ lệ lợi ích thông tin lớn nhất thì được chọn để phân tách tập huấn luyện D.

2) Thuật toán đề xuất:
Trong mục này, chúng tôi đề xuất thuật toán, gọi là HAC4.5*, chi tiết trong Thuật toán 2.

Thuật toán 2: Thuật toán HAC4.5*

Input: Tập mẫu huấn luyện D.
Output: Cây quyết định mờ S.
Method:
for each (thuộc tính mờ X của D)
begin
Xây dựng ĐSGT $\underline{X}_k$ tương ứng thuộc tính mờ X;
Chuyển các giá trị số và giá trị ngôn ngữ của X về các giá trị đoạn $\subseteq [0,1]$;
end
Khởi tạo tập các nút lá S; $S = D$;
for each (nút lá L thuộc S)
if (L thuần nhất) or (L.Tập thuộc tính là rỗng) **then**
L.Nhãn = Tên lớp
else
begin
if (L là thuộc tính mờ) **then**
begin
for each (khoảng mờ x của thuộc tính L)
for each (khoảng mờ y của thuộc tính L mà $y \neq x$)
//Gọi thuật toán 1
Tìm và thay thế x bởi $z = \sim_{\max}(x, y)$
end
end if
X = Thuộc tính tương ứng có GainRatio hay GainRatio^{HA} lớn nhất;
L.Nhãn = Tên thuộc tính X;
if (L là thuộc tính mờ) **then**
begin
T = Ngưỡng có GainRatio^{HA} lớn nhất;
Gán nhãn T của thuộc tính X cho cây S;
$S_1 = \{ I_{x_i} \big| I_{x_i} \subseteq L, I_{x_i} < T \}$;
S_1.Nút cha = L;
S_1.Thuộc tính = L.Thuộc tính - X;
$S_2 = \{ I_{x_i} \big| I_{x_i} \subseteq L, I_{x_i} > T \}$;
S_2.Nút cha = L;
S_2.Thuộc tính = L.Thuộc tính - X;
$S = S + S_1 + S_2 - L$; *//Đánh dấu nút L đã xét và bổ sung thêm các nút con của L ứng với các tập S_1 và S_2*
end
end if
else if (L là thuộc tính liên tục) **then**
begin
T = Ngưỡng có GainRatio^{HA} lớn nhất;
$S_1 = \{ x_i | \, x_i \in L, x_i \leq T \}$;
S_1.Nút cha = L;
S_1.Thuộc tính = L.Thuộc tính - X;
$S_2 = \{ x_i | \, x_i \in L, x_i > T \}$;
S_2.Nút cha = L;
S_2.Thuộc tính = L.Thuộc tính - X;
$S = S + S_1 + S_2 - L$; *//Đánh dấu nút L đã xét và bổ sung thêm 2 nút con của L ứng với tập S_1 và S_2*
end
else *//L là thuộc tính rời*
begin
$P = \{ x_i | \, x_i \in K, x_i$ đơn nhất$\}$;
for each ($x_i \in P$)
begin
$S_i = \{ x_j \big| x_j \in L, x_j = x_i \}$
S_i.Nút cha = L;
S_i.Thuộc tính = L.Thuộc tính - X;
$S = S + S_i$; *//Đánh dấu nút L đã xét và bổ sung tất cả các nút con của L ứng với tập S_i*
end
$S = S - L$;
end
end

3. Đánh giá thuật toán

Như vậy, bằng việc tìm các khoảng mờ lớn nhất cho mọi khoảng mờ trong tập huấn luyện và xác nhập chúng, thuật toán HAC4.5* đã làm giảm thiểu số lượng khoảng mờ tham gia trong quá trình học xây dựng cây nên làm giảm số nút trên cây kết quả. Tuy vậy, do sự kết nhập các khoảng mờ dựa trên độ đo về tính tương tự của các khoảng mờ đó đối với thuộc tính huấn luyện nên nó không sai lệch đến kết quả phân lớp cuối cùng.

Trong mỗi bước lặp của thuật toán, chúng ta đều phải tính các khoảng mờ lớn nhất và thực hiện sát nhập các khoảng mờ với chi phí là $O(n^2)$. Để ý rằng trong thuật toán trên, chúng ta không thể chuyển việc tìm và sát nhập các khoảng mờ lớn nhất ra khỏi vòng lặp. Do khi chúng ta tìm được một thuộc tính phù hợp để tạo cây thì tại các điểm phân chia này, tập mẫu huấn luyện tương ứng trên mỗi nhánh của cây thay đổi nên các khoảng mờ của thuộc tính mờ thay đổi.

Với m là số thuộc tính, n là số thể hiện của tập huấn luyện, độ phức tạp của C4.5 là $O(m \times n \times \log n)$; Với HAC4.5*, trước tiên ta mất $O(n^2)$ cho mỗi một thuộc tính mờ để tính các phân hoạch đoạn mờ. Sau đó, độ phức tạp của thuật toán tại mỗi bước lặp theo thuộc tính m_i là $O(n \times \log n)$ nếu m_i không phải là thuộc tính mờ và là $O(n \times n \times \log n)$ nếu là thuộc tính mờ do phải mất thêm $O(n)$ để tìm ngưỡng cho các khoảng mờ đối với thuộc tính này và $O(n^2)$ để tìm khoảng mờ lớn nhất cho mỗi khoảng mờ trong tập huấn luyện. Tuy vậy, việc tìm khoảng mờ lớn nhất và xác định ngưỡng cho các khoảng mờ là tuần tự nên

độ phứt tạp của HAC4.5* là $O(m \times n^3 \times \log n)$.

Tính đúng và tính dừng của thuật toán được rút ra từ tính đúng của C4.5 và cách thức đối sánh giữa các giá trị khoảng mờ.

IV. THỰC NGHIỆM VÀ SO SÁNH

Chương trình thực nghiệm được cài đặt bằng ngôn ngữ Java (Eclipse Mars Release (4.5.0) trên máy tính có cấu hình: Intel® Core™i5-2450, 4 CPU với mỗi CPU có tốc độc xử lý là 2, 5 GHz, RAM có 4 GB, hệ thống 64 bit, cho đồng thời cả ba thuật toán: C4.5, HAC4.5 và HAC4.5* trên đồng thời hai bộ dữ liệu là Adult và Bank.

1. Kết quả trên dữ liệu dự đoán thu nhập Adult

Bộ dữ liệu Adult có hơn 40.000 mẫu tin gồm 14 thuộc tính bao gồm dữ liệu rời rạc, liên tục, logic và mờ, trong đó có 2 thuộc tính Age (Tuổi) và HoursPerWeek (Số giờ làm việc) chứa cả dữ liệu rõ và mờ. Chúng tôi tách riêng biệt 20.000 mẫu tin cho tập huấn luyện và dùng 20.000 mẫu còn lại để chọn ngẫu nhiên 5.000 mẫu dùng cho việc kiểm tra. Kết quả cụ thể như Bảng I và Bảng II.

Bảng I
ĐỐI SÁNH HUẤN LUYỆN TRÊN DỮ LIỆU ADULT

Thuật toán	Thời gian huấn luyện (s)	Tổng số nút trên cây
C4.5	479,8	682
HAC4.5	1863,7	1873
HAC4.5*	2610,8	1624

Bảng II
TỶ LỆ KIỂM TRA TRÊN DỮ LIỆU ADULT

Thuật toán	Số mẫu kiểm tra				
	1000	2000	3000	4000	5000
C4.5	84,5%	85,7%	85,9%	86,2%	85,7%
HAC4.5	92,3%	91,5%	93,0%	95,0%	96,1%
HAC4.5*	92,8%	91,6%	93,2%	95,1%	96,3%

Bảng III
ĐỐI SÁNH HUẤN LUYỆN TRÊN DỮ LIỆU BANK

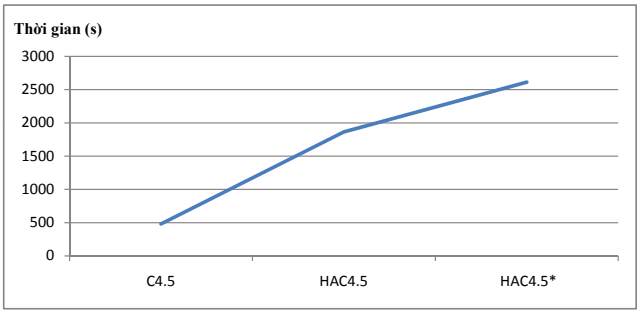
Thuật toán	Thời gian huấn luyện (s)	Tổng số nút trên cây
C4.5	2243	562
HAC4.5	4587,2	1061
HAC4.5*	6610,5	892

2. Kết quả trên dữ liệu dự đoán thu nhập trên dữ liệu Bank

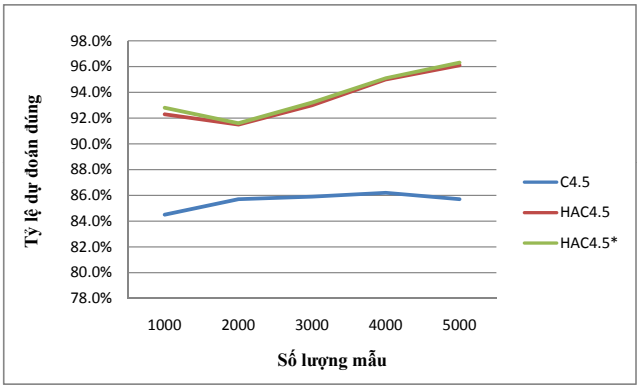
Bộ dữ liệu Bank có hơn 45.000 mẫu tin gồm 17 thuộc tính bao gồm dữ liệu rời rạc, liên tục, logic và mờ, trong đó thuộc tính Age (Tuổi) chứa cả dữ liệu rõ và mờ. Chúng tôi tách riêng biệt 30.000 mẫu tin cho tập huấn luyện và dùng 15.000 mẫu còn lại để chọn ngẫu nhiên 5.000 mẫu dùng cho việc kiểm tra. Kết quả được thể hiện trong Bảng III và Bảng IV.

Bảng IV
TỶ LỆ KIỂM TRA TRÊN DỮ LIỆU BANK

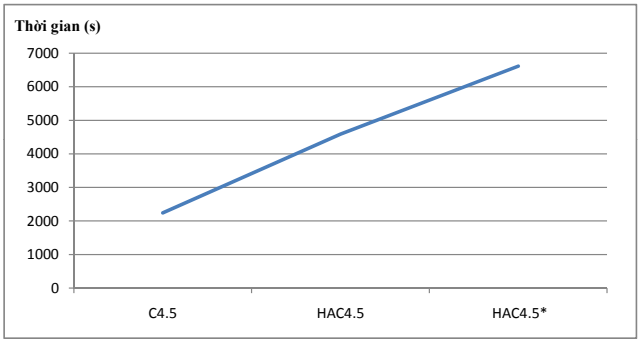
Thuật toán	Số mẫu kiểm tra				
	1000	2000	3000	4000	5000
C4.5	61,3%	58,1%	58,3%	61,2%	64,7%
HAC4.5	76,6%	72,6%	72,9%	76,5%	80,9%
HAC4.5*	85,2%	83,5%	83,2%	82,1%	81,1%



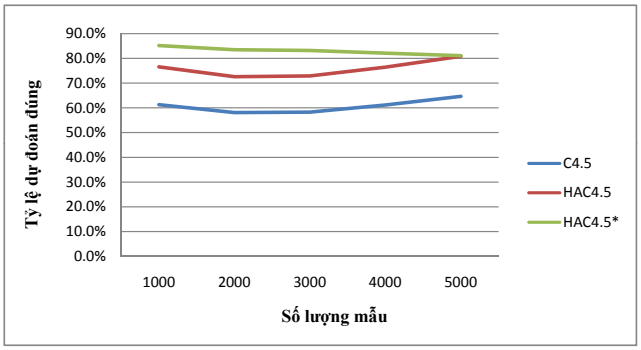
Hình 4. Đối sánh thời gian huấn luyện và số nút của cây kết quả trên dữ liệu Adult.



Hình 5. Đối sánh tỷ lệ kiểm tra từ 1.000 đến 5.000 trên mẫu trên dữ liệu Bank.



Hình 6. Đối sánh thời gian huấn luyện và số nút của cây kết quả trên dữ liệu Bank.



Hình 7. Đối sánh tỷ lệ kiểm tra từ 1.000 đến 5.000 trên mẫu trên dữ liệu Bank.

3. Đánh giá kết quả thực nghiệm

Việc đồng thời cài đặt cả ba thuật toán C4.5, HAC4.5 và HAC4.5*, kết quả đánh giá trên cùng các bộ dữ liệu là Adult và Bank đã cho phép chúng ta có các đánh giá sau. Về chi phí huấn luyện, kết quả ở Hình 4 và Hình 6 cho chúng ta thấy rằng, thuật toán C4.5 luôn cho thời gian nhanh nhất trong tất cả các bộ mẫu kể cả trong quá trình huấn luyện hay kiểm tra, vì nó bỏ qua các giá trị mờ trong tập mẫu nên không mất thời gian xử lý. HAC4.5 phải trải qua quá trình xây dựng các ĐSGT cho các trường mờ và chi phí để chuyển đổi các giá trị về đoạn $[0, 1]$ ban đầu và tại mỗi bước cần thêm thời gian để chọn đoạn phân chia nên tốn nhiều thời gian hơn nhiều so với C4.5. HAC4.5* vì mỗi bước lặp cần thêm thời gian để tìm các khoảng mờ lớn nhất cho miền trị mờ của thuộc tính mờ tương ứng nên HAC4.5* chậm nhất so với các thuật toán khác.

Về dự đoán, kết quả ở Hình 5 và Hình 7 cho chúng ta thấy rằng, trước hết C4.5 bỏ qua các giá trị mờ trong tập mẫu, chỉ quan tâm các giá trị rõ nên cây kết quả thu được khá giản đơn vì ít nút. Tuy nhiên, do việc bỏ qua các giá trị mờ nên làm mất dữ liệu tại các trường mờ, vì thế kết quả dự đoán không cao. Thứ đến, HAC4.5 xây dựng một ĐSGT tại các trường mờ và dùng nó để thuần nhất tập mẫu nên chúng ta đã xử lý được các giá trị mờ mà vẫn giữ nguyên các giá trị rõ nên không làm xuất hiện thêm sai số trong quá trình phân hoạch,vì thế kết quả trong các quá trình dự đoán tốt hơn nhiều so với C4.5. Tuy vậy, so với C4.5 thì cây kết quả thu được không giản đơn vì có nhiều nút. Cuối cùng, HAC4.5* cho kết quả tốt nhất vì trong quá trình huấn luyện cây, chúng ta đã tìm được các điểm phân hoạch tốt nhất tại các thuộc tính mờ nên ít cây kết quả thu được có sai số ít hơn. Hơn nữa, việc tìm các khoảng mờ lớn nhất và kết nhập các giá trị mờ trên thuộc tính mờ đã làm cho lực lượng của các thuộc tính mờ tương ứng giảm, vì thế số nút trên cây thu được cũng giảm nên cây kết quả thu được là tốt nhất.

V. KẾT LUẬN

Bài toán phân lớp cây quyết định mờ đóng vai trò quan trọng trong quá trình khai phá dữ liệu. Tuy vậy, việc phân lớp bằng cây quyết định mờ dựa trên lý thuyết tập mờ luôn gặp phải những hạn chế do xuất phát từ bản thân nội tại của lý thuyết tập mờ. ĐSGT với lợi thế của mình đã trở thành một công cụ thật sự hữu ích để giải quyết bài toán phân lớp cây quyết định.

Bằng việc nhận ra các hạn chế của phương pháp định lượng ngữ nghĩa theo điểm của việc sử dụng ĐSGT trong quá trình huấn luyện, cách thức đối sánh theo khoảng mờ là một giải pháp nhằm giảm thiểu sai số cho việc học cây quyết định mờ. Bằng việc đưa ra khái niệm khoảng mờ lớn nhất, bài báo đã đề xuất một phương pháp học quy nạp cây quyết định mờ HAC4.5*, nhằm thu được cây quyết định mờ đơn giản và hiệu quả cho bài toán phân lớp mờ.

TÀI LIỆU THAM KHẢO

[1] A. Bikas, E. Voumvoulakis, and N. Hatziargyriou, "Neuro-fuzzy decision trees for dynamic security control of power systems," in *15th International Conference on Intelligent System Applications to Power Systems (ISAP'09)*. IEEE, 2009, pp. 1–6.

[2] G. G. Anuradha, "Fuzzy decision tree construction in crisp scenario through fuzzified trapezoidal membership function," *Internetworking Indonesia*, vol. 7, no. 2, pp. 21–28, 2015.

[3] B. Chandra and P. P. Varghese, "Fuzzy SLIQ decision tree algorithm," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 5, pp. 1294–1301, 2008.

[4] H. Ishibuchi and T. Nakashima, "Effect of rule weights in fuzzy rule-based classification systems," *IEEE Transactions on Fuzzy Systems*, vol. 9, no. 4, pp. 506–515, 2001.

[5] K. K. R. C and V. Babu, "A survey on issues of decision tree and non-decision tree algorithms," *International Journal of Artificial Intelligence and Applications for Smart Devices*, vol. 4, no. 1, pp. 9–32, 2016.

[6] S. Moustakidis, G. Mallinis, N. Koutsias, J. B. Theocharis, and V. Petridis, "SVM-based fuzzy decision trees for classification of high spatial resolution remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 1, pp. 149–169, 2012.

[7] M. Mehta, R. Agrawal, and J. Rissanen, "SLIQ: A fast scalable classifier for data mining," *Advances in Database Technology-EDBT'96*, pp. 18–32, 1996.

[8] J. Shafer, R. Agrawal, and M. Mehta, "SPRINT: A Fast Scalable Classifier for Data Mining," *IBM Almaden Reseach Center*, 1996.

[9] P. Fatima, D. Parveen, and M. Sathik, "Fuzzy decision tree based effective imine indexing," *International Journal of Computer Technology and Electronics Engineering*, vol. 1, 2011.

[10] J. R. Quinlan, "Simplifying decision trees," *International Journal of Man-Machine Studies*, vol. 27, no. 3, pp. 221–234, 1987.

[11] R. H. Tajiri, E. Z. Marques, B. B. Zarpelão, and L. de Souza Mendes, "A new approach for fuzzy classification in relational databases," in *International Conference on Database and Expert Systems Applications*. Springer, 2011, pp. 511–518.

[12] S. Ruggieri, "Efficient C4.5," 2000.

[13] Z. Qin and Y. Tang, "Linguistic decision trees for classification," in *Uncertainty Modeling for Data Mining*. Springer, 2014, pp. 77–119.

[14] J. Zhang and V. Honavar, "Learning decision tree classifiers from attribute value taxonomies and partially specified data," in *Proceedings of the International Conference on Machine Learning*, Washington DC, 2003.

[15] M. Zeinalkhani and M. Eftekhari, "Comparing different stopping criteria for fuzzy decision tree induction through IDFID3," *Iranian Journal of Fuzzy Systems*, vol. 11, no. 1, pp. 27–48, 2014.

[16] N. C. Ho and N. Van Long, "Fuzziness measure on complete hedge algebras and quantifying semantics of terms in linear hedge algebras," *Fuzzy Sets and Systems*, vol. 158, no. 4, pp. 452–471, 2007.

[17] N. C. Ho and H. V. Nam, "An algebraic approach to linguistic hedges in Zadeh's fuzzy logic," *Fuzzy Sets and Systems*, vol. 129, no. 2, pp. 229–254, 2002.

[18] N. C. Ho and W. Wechler, "Hedge algebras: an algebraic approach to structure of sets of linguistic truth values," *Fuzzy Sets and Systems*, vol. 35, no. 3, pp. 281–293, 1990.

[19] ——, "Extended hedge algebras and their application to fuzzy logic," *Fuzzy Sets and Systems*, vol. 52, no. 3, pp. 259–281, 1992.

[20] L. V. T. Lân, N. M. Hân, and N. C. Hào, "An algorithm to building a fuzzy decision tree for data classification problem based on the fuzziness intervals matching," *Journal of Computer Science and Cybernetics*, vol. 32, no. 4, pp. 367–380, 2017.

Lê Văn Tường Lân sinh năm 1974 tại thành phố Huế. Ông nhận bằng Thạc sĩ, chuyên ngành Công nghệ Thông tin tại Trường Đại học Bách khoa Hà Nội, năm 2002. Hiện nay, ông đang là giảng viên và là nghiên cứu sinh tại Khoa Công nghệ Thông tin, Trường Đại học Khoa học, Đại học Huế, chuyên ngành Khoa học Máy tính. Lĩnh vực nghiên cứu của ông là khai phá dữ liệu và công nghệ phần mềm.

Nguyễn Mậu Hân sinh năm 1957 tại Thừa Thiên Huế. Ông nhận bằng Tiến sĩ tại Viện Công nghệ Thông tin Hà Nội và được phong hàm Phó Giáo sư năm 2013. Hiện nay, ông là giảng viên tại Khoa Công nghệ Thông tin, Trường Đại học Khoa học, Đại học Huế. Lĩnh vực nghiên cứu của ông là xử lý song song và phân tán, tính toán lưới và điện toán đám mây.

Nguyễn Công Hào sinh năm 1976 tại Thừa Thiên Huế. Ông nhận bằng Tiến sĩ tại Viện Công nghệ Thông tin Hà Nội, năm 2008. Hiện nay, ông là Giám đốc Trung tâm Công nghệ Thông tin, Đại học Huế. Lĩnh vực nghiên cứu của ông là cơ sở dữ liệu mờ, các phương pháp tính toán mềm và các phương pháp lập luận xấp xỉ.