PHÂN LOẠI DỮ LIỆU GIEN VỚI GIẢI THUẬT MÁY HỌC ARCX4-RODT

Đặng Quốc Bảo[1], Trần Huỳnh Lê[2], Đỗ Thanh Nghị[3]

ABSTRACT

In this paper, we propose a new algorithm, called ArcX4-rODT (ArcX4 of random oblique decision trees) to classify gene data which have very small amount of samples in very high dimensions and noise. Our ArcX4-rODT algorithm constructs sequentially k random oblique trees so that each tree concentrates mostly on the errors produced by the previous ones. Furthermore, the hyper-plane obtained by Fisher's linear discriminant analysis is also used to perform multivariate splitting data at each internal node of the decision tree. Thus, the ArcX4-rODT can deal with very-high-dimensional data and noise. The experimental results on gene datasets from datam.i2r.a-star.edu.sg/datasets/krbd/ showed that our ArcX4-rODT algorithm outperforms random forest of C4.5 (RF-C4.5) and SVM (LibSVM).

Keywords: ArcX4, Random oblique decision tree, Linear discriminant analysis, gene classification

Title: Classification of Gene Expression using ArcX4-rODT Learning Algorithm

TÓM TẮT

Trong bài viết này, chúng tôi trình bày giải thuật máy học mới ArcX4 của cây quyết định ngẫu nhiên xiên phân (ArcX4-rODT). Giải thuật ArcX4-rODT xây dựng tuần tự tập hợp cây xiên phân ngẫu nhiên, cây xây dựng sau sẽ tập trung lên các mẫu bị phân lớp sai bởi các cây trước, mỗi cây thành viên sử dụng siêu phẳng phân chia dữ liệu hiệu quả tại mỗi nút của cây dựa trên phân tích biệt lập tuyến tính. Việc xây dựng cây xiên phân ngẫu nhiên vì thế tạo cho giải thuật có khả năng làm việc tốt trên dữ liệu có số chiều lớn và nhiễu như dữ liệu gien. Kết quả thử nghiệm trên các tập dữ liệu gien từ site datam.i2r.a-star.edu.sg/datasets/krbd/ cho thấy rằng giải thuật ArcX4-rODT mới do chúng tôi đề xuất phân loại tốt hơn khi so sánh với rừng ngẫu nhiên của cây quyết định C4.5 và máy học véctơ hỗ trợ.

Từ khóa: Giải thuật ArcX4, Cây ngẫu nhiên xiên phân, Phương pháp phân tích biệt lập tuyến tính, Phân loại dữ liệu gien

1 GIỚI THIỆU

Phân lớp dữ liệu có số chiều lớn có nhiễu như dữ liệu gien được biết là một trong 10 vấn đề khó của cộng đồng khai mỏ dữ liệu (Yang & Wu, 2006). Mô hình học phân lớp thường cho kết quả tốt trong khi học nhưng lại cho kết quả rất thấp trong tập thử. Vấn đề khó khăn thường gặp chính là số chiều quá lớn lên đến hàng nghìn chiều thậm chí đến cả triệu và dữ liệu thường tách rời nhau trong không gian có số chiều lớn việc tìm mô hình phân lớp tốt có khả năng làm việc với dữ liệu có số chiều lớn là khó khăn do có quá nhiều khả năng lựa chọn mô hình. Việc tìm một

[1] Khoa CNTT, Trường ĐH Đồng Tháp, Số 783 Phạm Hữu Lầu, P.6, Tp. Cao Lãnh
[2] Phòng Thanh Tra Đào Tạo, Trường ĐH Đồng Tháp
[3] Bộ môn Khoa Học Máy Tính, khoa CNTT&TT, Trường Đại học Cần Thơ

mô hình phân lớp hiệu quả (phân lớp dữ liệu tốt trong tập thử) trong không gian giả thiết lớn là vấn đề khó. Đã có hai lớp giải thuật tiêu biểu là máy học véc tơ hỗ trợ của Vapnik (SVM [Vapnik, 1995]) và rừng ngẫu nhiên của [Breiman, 2001] được biết đến như là những giải thuật phân lớp hiệu quả các tập dữ liệu có số chiều lớn như dữ liệu gien.

Từ những năm 1990, cộng đồng máy học đã nghiên cứu cách để kết hợp nhiều mô hình phân loại thành tập hợp các mô hình phân loại để cho tính chính xác cao hơn so với chỉ một mô hình phân loại. Mục đích của các mô hình tập hợp là làm giảm variance và/hoặc bias của các giải thuật học. Bias là khái niệm về lỗi của mô hình học (không liên quan đến dữ liệu học) và variance là lỗi do tính biến thiên của mô hình so với tính ngẫu nhiên của các mẫu dữ liệu học. (Buntine, 1992) đã giới thiệu các kỹ thuật Bayes để giảm variance của các phương pháp học. Phương pháp xếp chồng (Wolpert, 1992) hướng tới việc cực tiểu hóa bias của các giải thuật học. Trong khi (Freund & Schapire, 1995) đưa ra Boosting, (Breiman, 1998) đề nghị ArcX4 để cùng giảm bias và variance, còn Bagging (Breiman, 1996) thì giảm variance của giải thuật học nhưng không làm tăng bias quá nhiều. Tiếp cận rừng ngẫu nhiên (Breiman, 2001) là một trong những phương pháp tập hợp mô hình thành công nhất. Giải thuật rừng ngẫu nhiên xây dựng cây không cắt nhánh nhằm giữ cho bias thấp và dùng tính ngẫu nhiên để điều khiển tính tương quan thấp giữa các cây trong rừng. Tiếp cận rừng ngẫu nhiên cho độ chính xác cao khi so sánh với các thuật toán học có giám sát hiện nay, bao gồm cả AdaBoost, ArcX4 và SVM. Khi xử lý dữ liệu cho có số chiều lớn và có số phần tử ít như dữ liệu gien thì rừng ngẫu nhiên và SVM là hai giải thuật học nhanh, chịu đựng nhiễu tốt và không bị tình trạng học vẹt, điều này ngược lại với AdaBoost, ArcX4 rất dễ bị học vẹt và ảnh hưởng lớn với nhiễu (Grove & Schuurmans, 1998).

Việc xây dựng cây quyết định thông thường như giải thuật C4.5 (Quinlan, 1993) và CART (Breiman *et al*., 1984) trong rừng ngẫu nhiên và của AdaBoost, ArcX4 chỉ chọn một thuộc tính dùng để phân hoạch tại mỗi nút. Vì thế, cá nhân mỗi cây kém hiệu quả khi làm việc với dữ liệu có sự phụ thuộc nhau giữa các thuộc tính, thường gặp ở những dữ liệu có số chiều rất lớn. Để nâng cao hiệu quả xử lý dữ liệu có số chiều lớn như dữ liệu gien, chúng tôi đề nghị thay thế cây quyết định thông thường trong ArcX4 bằng cây quyết định ngẫu nhiên xiên phân (rODT). Cây ngẫu nhiên xiên phân sử dụng siêu phẳng phân chia dữ liệu hiệu quả tại mỗi nút của cây dựa trên phương pháp phân tích biệt lập tuyến tính LDA (Fisher, 1936) (khác với chiến lược heuristics của OC1 (Murthy *et al*., 1993)). Việc xây dựng cây xiên phân ngẫu nhiên vì thế tạo cho giải thuật ArcX4 cây quyết định ngẫu nhiên xiên phân (ArcX4-rODT) có khả năng làm việc tốt trên dữ liệu có số chiều lớn và nhiễu như dữ liệu gien. Các kết quả kiểm thử trên 10 tập dữ liệu gien có số chiều lớn (Jinyan & Huiqing, 2002) đã cho thấy ArcX4 cây quyết định ngẫu nhiên xiên phân mà chúng tôi đề xuất cho độ chính xác cao hơn rừng ngẫu nhiên thông thường của C4.5 (RF-C4.5) và LibSVM (Chang & Lin, 2001) dựa trên các tiêu chí về precision, recall, F1-measure và độ chính xác accuracy (van Rijsbergen, 1979).

Phần tiếp theo của bài viết này được trình bày như sau: phần 2 trình bày ngắn gọn về giải thuật ArcX4-rODT của chúng tôi đề xuất. Phần 3 trình bày các kết quả thực nghiệm tiếp theo sau đó là kết luận và hướng phát triển.

2 GIẢI THUẬT ARCX4-RODT

Hiệu quả của một giải thuật học như đã nghiên cứu của (Breiman, 1996, 1998, 2001) dựa trên cơ sở của 2 thành phần lỗi là bias và variance mà ở đó, thành phần lỗi bias là lỗi của mô hình học và variance là lỗi do tính biến thiên của mô hình so với tính ngẫu nhiên của các mẫu dữ liệu học. Trong nghiên cứu kết hợp nhiều mô hình phân loại thành tập hợp các mô hình phân loại để cho tính chính xác cao hơn so với chỉ một mô hình đơn.

Đầu vào:

- m phần tử dữ liệu : $\{(x_i, y_i)\}i=1,m$ với $x_i \epsilon R^n$ và $y_i \epsilon \{1, -1\}$
- số bước lặp T

Huấn luyện:

► khởi động phân phối của m phần tử dữ liệu $Dist_1(j)$

cho j = 1 tới m thực hiện

$Dist_1(j) = 1/m$

► cho i = 1 tới T thực hiện (lặp T bước)

- lấy mẫu S_i phần tử dựa trên phân phối $Dist_i$
- học mô hình cây xiên phân ngẫu nhiên h_i từ tập mẫu S_i

$h_i = rODT(S_i)$

- tính lại lỗi dự đoán của từng phần tử x_j khi sử dụng các bộ phân lớp được xây dựng trước đó

$$\varepsilon_j = \sum_{t=1}^{i} ht(xj) \neq yj$$

- cập nhật lại phân phối của m phần tử dữ liệu

cho j = 1 tới m thực hiện

$Dist_{i+1}(j) = (1 + \varepsilon_j^4)/fac_i$ với $fac_i = \sum_{j=1}^{m}(1+\varepsilon_j^4)$

► trả về tập T mô hình cây xiên phân $\{h_i\}i=1,T$

Phân lớp:

► phân lớp phần tử x: bình chọn số đông của $\{h_i(x)\}i=1,T$

Giải thuật 1: ArcX4 cây quyết định ngẫu nhiên xiên phân

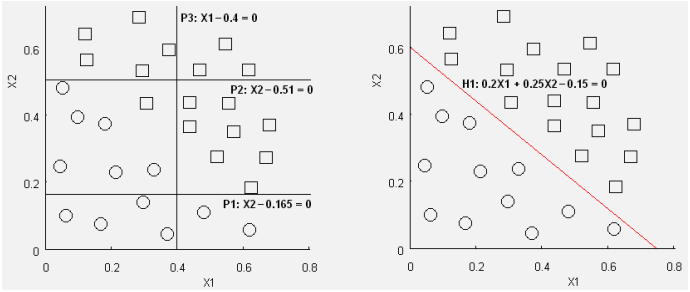
Boosting được Freund và các đồng nghiệp của ông phát triển trong thập niên 1990. Đây là một phương pháp áp dụng một tập các bộ phân lớp yếu (weak learner) để nâng cao hiệu quả của các bộ phân lớp này bằng cách giảm bias và variance. Trong cùng thời điểm Breiman cũng đề xuất lớp các giải thuật Arcing [Breiman, 1998] nhằm giảm cả bias và variance. Theo Breiman, Boosting là một dạng trong lớp giải thuật Arcing. Trong đó có giải thuật ArcX4 cho kết quả tương tự như AdaBoost (Freund & Schapire, 1995).

Ý tưởng chính của giải thuật ArcX4 (như mô tả trong giải thuật 1) lặp lại quá trình học của một bộ phân lớp yếu nhiều lần. Sau mỗi bước lặp, bộ phân lớp yếu (ví dụ như: Naïve Bayes, cây quyết định, …) sẽ tập trung học trên các phần tử bị phân

lớp sai trong các lần trước. Để làm được điều này, ta gán cho mỗi phần tử một trọng số. Khởi tạo, trọng số của các phần tử bằng nhau trong lần lặp đầu tiên. Sau mỗi bước học, các trọng số này sẽ được cập nhật lại (tăng trọng số cho các phần tử bị phân lớp sai). Ở bước thứ i, ta lấy tập mẫu S_i trên tập dữ liệu và xây dựng mô hình h_i từ tập mẫu S_i. Lặp lại quá trình này sau T bước, ta sẽ được T mô hình cơ sở, kết hợp các mô hình cơ sở này lại ta sẽ có được một bộ phân lớp mạnh. ArcX4 rất dễ cài đặt và cho kết quả tốt trong thực tế.

ArcX4 thường dùng giải thuật cơ sở yếu là cây quyết định CART (Breiman, 1984]) hay C4.5 (Quinlan, 1993). Như đã nghiên cứu của (Grove & Schuurmans, 1998), Boosting và Arcing mặc dù cho kết quả tốt trong thực tế nhưng thường bị học vẹt khi tăng số bước lặp vượt qua một ngưỡng nào đó. Để khắc phục nhược điểm này, Friedman và các cộng sự (Friedman *et al*., 2008) đề xuất sử dụng mô hình cơ sở cây quyết định phải đơn giản (cây có kích thước không quá 8 nút), khi đó số bước lặp tăng cao vẫn đảm bảo rằng Boosting và Arcing không bị tình trạng học vẹt.

Chúng tôi đề xuất xây dựng mô hình cơ sở dùng trong ArcX4 là cây ngẫu nhiên xiên phân (gọi là rODT) thay vì sử dụng cây quyết định thông thường như C4.5 hay CART. Ngoài việc giới hạn kích thước, tại mỗi nút trong của cây, xây dựng phân hoạch xiên phân (siêu phẳng phân hoạch hiệu quả thu được từ phân tích biệt lập tuyến tính FDA) dựa trên tập ngẫu nhiên các thuộc tính. Việc xây dựng phân hoạch xiên phân giúp cải thiện tính mạnh mẽ của cây khi làm việc với các tập dữ liệu có số chiều lớn và phụ thuộc lẫn nhau.

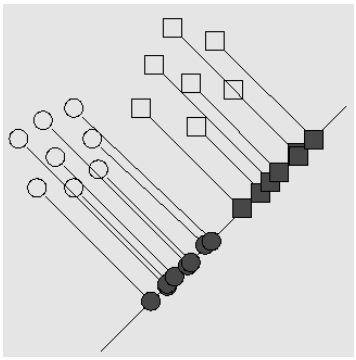


Hình 1: Phân hoạch đơn thuộc tính (trái), phân hoạch đa thuộc tính (phải)

Ví dụ như trong hình 1, bất kỳ việc phân hoạch đơn thuộc tính nào (song song với trục tọa độ như giải thuật C4.5 hay CART) đều không thể tách dữ liệu một lần duy nhất thành hai lớp một cách hoàn toàn mà phải thực hiện nhiều lần phân hoạch, nhưng việc phân hoạch đa chiều (xiên phân, kết hợp 2 thuộc tính) có thể thực hiện một cách hoàn hảo với duy nhất một lần. Vì thế, việc phân hoạch đơn thuộc tính được dùng để xây dựng cây thông thường thì không hiệu quả trong trường hợp này. Để khắc phục nhược điểm trên, nhiều giải thuật xây dựng cây quyết định sử dụng phân hoạch đa thuộc tính (xiên phân) tại các nút được đề nghị. Nghiên cứu tiên phong của (Murthy *et al*., 1993) đã đưa ra giải thuật OC1, một hệ thống dùng để xây dựng các cây quyết định xiên trong đó dùng leo đồi để tìm một phân hoạch xiên tốt dưới dạng một siêu phẳng.

ArcX4 cây ngẫu nhiên xiên phân (ArcX4-rODT) của chúng tôi xây dựng các cây xiên phân ngẫu nhiên dựa trên siêu phẳng hiệu quả (phân hoạch hiệu quả cao, khả

năng chịu đựng nhiễu tốt) thu được từ huấn luyện LDA (Fisher, 1936). Ý tưởng chính của LDA là tìm véctơ sao cho khi chiếu dữ liệu lên đó thì độ biệt lập giữa trung bình dữ liệu của 2 lớp là lớn nhất và độ chồng lấp giữa 2 lớp là nhỏ nhất.



Hình 2: Minh hoạ véctơ (w) dùng để chiếu dữ liệu 2 thuộc tính (chiều)

Một cách ngắn gọn, xét một ví dụ phân lớp nhị phân tuyến tính (hình tròn, vuông) như trong hình 2, với m điểm dữ liệu x_i ($i=1,m$) trong không gian n chiều (thuộc tính). Tập dữ liệu phân làm 2 lớp R_1 (có N_1 phần tử), và R_2 (có N_2 phần tử). Để tìm véctơ chiếu tối ưu (w) ta cần tính như sau.

Trung bình (trọng tâm) mỗi lớp:

$$m_1 = \frac{1}{N_1}\sum_{x_i \in R_1} x_i, \quad m_2 = \frac{1}{N_2}\sum_{x_i \in R_2} x_i \tag{1}$$

Chiếu m_1, m_2 lên véctơ w:

$$\tilde{m}_1 = \frac{1}{N_1}\sum_{x_i \in R_1} w^T x_i = w^T m_1 \qquad \tilde{m}_2 = \frac{1}{N_2}\sum_{x_i \in R_2} w^T x_i = w^T m_2 \tag{2}$$

Khoảng cách giữa m_1 và m_2 sau khi chiếu lên w (độ biệt lập tuyến tính) :

$$|\tilde{m}_2 - \tilde{m}_1| = |w^T(m_2 - m_1)| \tag{3}$$

Mật độ phân bố (scatter) của dữ liệu 2 lớp sau khi chiếu :

$$\tilde{s}_1^2 = \sum_{y_i : x_i \in R_1} (y_i - \tilde{m}_1)^2 = \sum_{x_i \in R_1} (w^T x_i - w^T m_1)^2 = w^T S_1 w \tag{4}$$

$$s_2^2 = \sum_{y_i : x_i \in R_2} (y_i - \tilde{m}_2)^2 = \sum_{x_i \in R_2} (w^T x_i - w^T m_2)^2 = w^T S_2 w$$

Với S_1, S_2 là:

$$S_1 = \sum_{x_i \in R_1} (x_i - m_1)(x_i - m_1)^T \qquad S_2 = \sum_{x_i \in R_2} (x_i - m_2)(x_i - m_2)^T \tag{5}$$

Tỉ số giữa độ biệt lập tuyến tính và tổng mật độ phân bố :

$$f(w) = \frac{(\tilde{m}_2 - \tilde{m}_1)^2}{\tilde{s}_1^2 + \tilde{s}_2^2} = \frac{w^T S_B w}{w^T S_w w} \tag{6}$$

Trong đó S_W là ma trận tán xạ bên trong mỗi lớp và S_B là ma trận tán xạ giữa 2 lớp, được tính như sau :

$$S_w = S_1 + S_2 \quad (7)$$

$$S_B = (m_2 - m_1)(m_2 - m_1)^T \quad (8)$$

Mục tiêu là cực đại hoá *f(w)*, đưa đến việc giải (9) :

$$w = S_w^{-1}(m_2 - m_1) \quad (9)$$

Ngoài ra, w tối ưu còn được tìm bằng phương pháp lấy đạo hàm f(w) dẫn đến bài toán tìm giá trị riêng suy rộng (the generalized eigenvalue problem) trong (10):

$$S_w^{-1} S_B w = \lambda w \quad (10)$$

Mục đích của bài toán phân lớp là cần xác định siêu phẳng phân chia dữ liệu. Nên, *w* là véctơ pháp tuyến của siêu phẳng. Độ lệch (*b*) của siêu phẳng *(w, b)* được tính dựa trên (11):

$$b = -\frac{1}{2} w^T (m_1 + m_2) \quad (11)$$

Việc tìm *w* tối ưu theo LDA chỉ cần lời giải của các đẳng thức tuyến tính trên. Siêu phẳng do LDA tìm được sẽ không tốt khi mà độ biệt lập tuyến tính của dữ liệu không dựa vào hai trọng tâm m_1 và m_2 (trường hợp dữ liệu phi tuyến). Vấn đề trên sẽ không ảnh hưởng lớn đến kết quả vì mô hình cây quyết định thực hiện nhiều phân hoạch xiên phân LDA cho đến nút lá chứ không phải chỉ thực hiện duy nhất một lần phân hoạch.

Việc sử dụng mô hình cơ sở là các cây quyết định ngẫu nhiên xiên phân thay vì là phân hoạch 1 chiều như C4.5 hay CART giúp cho ArcX4-rODT trở nên hiệu quả, có khả năng chịu đựng nhiễu cao, tránh học vẹt khi xử lý dữ liệu có số phần tử nhỏ nhưng số chiều rất lớn và lại có nhiễu như dữ liệu gien.

3 KẾT QUẢ THỰC NGHIỆM

Để có thể đánh giá hiệu quả của giải thuật, chúng tôi cài đặt giải thuật ArcX4 cây quyết định ngẫu nhiên xiên phân (ArcX4-rODT) bằng ngôn ngữ lập trình C/C++. Dữ liệu gien chúng tôi chạy thử nghiệm, có số chiều rất lớn, được lấy tại (Jinyan & Huiqing, 2002). Bên cạnh đó, chúng tôi quan sát kết quả của ArcX4 cây quyết định ngẫu nhiên xiên phân trong thực nghiệm bằng cách so sánh với rừng ngẫu nhiên của cây quyết định C4.5 và SVM. Chúng tôi cũng sử dụng mã nguồn của C4.5 được cung cấp bởi (Quinlan, 1993) để tạo ra giải thuật rừng ngẫu nhiên cây quyết định C4.5 (RF-C4.5 (Do *et al*., 2009)). Sau cùng chúng tôi cũng sử dụng giải thuật SVM chuẩn LibSVM (Chang & Lin, 2001). Tất cả các kết quả đều được thực hiện trên một máy tính cá nhân chạy hệ điều hành Linux.

Chúng tôi tiến hành thực nghiệm trên 10 tập dữ liệu gien có số chiều rất lớn từ kho dữ liệu sinh-y học. Mô tả các tập dữ liệu được tìm thấy trong bảng 1. Chúng tôi chú ý đến các phương pháp kiểm tra được liệt kê trong cột cuối của bảng 1. Với

những tập dữ liệu có sẵn tập học và tập thử, chúng tôi dùng tập học để thử điều chỉnh các tham số ở đầu vào của các giải thuật nhằm thu được độ chính xác tốt khi học. Sau đó, dùng mô hình thu được để phân lớp tập thử. Nếu tập học và tập thử không có sẵn, các giao thức kiểm tra chéo (cross-validation protocol) để đánh giá. Do các tập dữ liệu có ít hơn 300 phần tử, chúng tôi dùng giao thức kiểm tra chéo leave-one-out (loo). Tức là dùng một phần tử trong tập dữ liệu để thử, các phần tử khác dùng để học. Lặp lại đến khi tất cả các phần tử đều được dùng để thử một lần.

Bảng 1: Mô tả các tập dữ liệu gien

ID	Tập dữ liệu	Số phần tử	Số chiều	Lớp	Nghi thức
1	Colon Tumor	62	2000	Tumor, normal	loo
2	ALL-AML-Leukemia	72	7129	ALL, AML	trn-tst
3	Breast Cancer	97	24481	relapse, non-relapse	trn-tst
4	Prostate Cancer	136	12600	cancer, normal	trn-tst
5	Lung Cancer	181	12533	cancer, normal	trn-tst
6	Central Nervous System	60	7129	positive, negative	loo
7	Diffuse Large B-Cell Lymphoma	47	4026	germinal, activated	loo
8	*Subtypes of Acute Lymphoblastic (Hyperdip)	327	12558	Hyperdip, rest	trn-tst
9	*Subtypes of Acute Lymphoblastic (TEL-ML1)	327	12558	TEL-AML1, rest	trn-tst
10	*Subtypes of Acute Lymphoblastic (Others)	327	12558	others, diagnostic groups	trn-tst

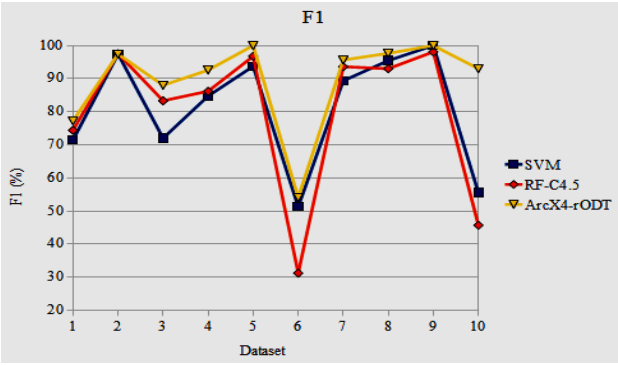
Để thấy rõ hơn tính hiệu quả của ArcX4-rODT so với RF-C4.5 và LibSVM, chúng tôi tiến hành phân tích hiệu quả của các thuật toán phân lớp dựa trên các tiêu chí như precision, recall, F1-measure và accuracy [van Rijsbergen, 1979]. Precision của một lớp là số điểm dữ liệu được phân lớp đúng về lớp này chia cho tổng số điểm dữ liệu được phân về lớp này. Recall của một lớp là số điểm dữ liệu được phân lớp đúng về lớp này chia cho tổng số điểm dữ liệu của lớp. F1-measure là tổng hợp của precision và recall, và được định nghĩa là hàm trung bình điều hòa giữa hai giá trị precision và recall. Độ chính xác accuracy là số điểm dữ liệu được phân lớp đúng của tất cả các lớp chia cho tổng số điểm dữ liệu. Chúng tôi thu được kết quả như trình bày trong bảng 2. Những kết quả tốt nhất sẽ được tô đậm.

Từ bảng kết quả phân lớp thu được của các giải thuật khi xử lý 10 tập dữ liệu gien cho thấy giải thuật của chúng tôi, ArcX4 cây quyết định ngẫu nhiên xiên phân (ArcX4-rODT) cho kết quả tốt hơn so với các giải thuật rừng ngẫu nhiên thông thường RF-C4.5 và SVM chuẩn LibSVM. Dựa trên tiêu chí precision, ArcX4-rODT thắng 8 trong 10 tập dữ liệu. Với tiêu chí recall và cả F1, ArcX4-rODT thắng 9 trong 10 tập dữ liệu.

Bảng 2: Kết quả phân lớp của LibSVM, RF-C4.5 và ArcX4-rODT

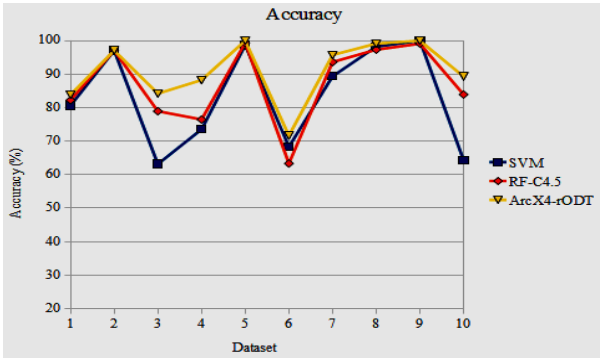
ID	Precision			Recall			F1-measure		
	Lib-SVM	RF-C4.5	ArcX4-rODT	Lib-SVM	RF-C4.5	ArcX4-rODT	Lib-SVM	RF-C4.5	ArcX4-rODT
1	68.18	76.19	**77.27**	75.00	72.73	**77.27**	71.43	74.42	**77.27**
2	**100**	95.24	**100**	95.00	**100**	95.00	97.44	**97.56**	97.44
3	69.23	83.33	**84.62**	75.00	83.33	**91.67**	72.00	83.33	**88.00**
4	73.53	75.76	**86.21**	**100**	**100**	**100**	84.75	86.21	**92.59**
5	88.26	93.75	**100**	**100**	**100**	**100**	93.75	96.77	**100**
6	**47.62**	45.46	**47.62**	55.56	23.81	**62.5**	51.28	31.25	**54.05**
7	91.30	**95.65**	91.67	87.50	91.67	**100**	89.36	93.62	**95.65**
8	95.46	95.24	**100**	**95.46**	90.91	**95.46**	95.46	93.02	**97.67**
9	**100**	**100**	**100**	**100**	96.30	**100**	**100**	98.11	**100**
10	92.59	**100**	91.95	39.68	29.63	**94.12**	55.56	45.71	**93.02**

Quan sát đồ thị 1 trình bày kết quả với tiêu chí F1 của cả 3 giải thuật, giải thuật ArcX4-rODT của chúng tôi đề xuất luôn ở cận trên.



Đồ thị 1: So sánh tiêu chí F1 của 3 giải thuật trên 10 tập dữ liệu

Đồ thị 2 trình bày kết quả với tiêu chí accuracy cho thấy ArcX4-rODT của chúng tôi luôn tốt hơn LibSVM và RF-C4.5. Những kết quả đạt được cho phép chúng tôi tin rằng giải thuật ArcX4 cây quyết định ngẫu nhiên xiên phân của chúng tôi đề nghị phân lớp hiệu quả trên dữ liệu gien có số chiều rất lớn.



Đồ thị 2: So sánh tiêu chí accuracy của 3 giải thuật trên 10 tập dữ liệu

4 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Chúng tôi vừa trình bày giải thuật máy học mới ArcX4 của cây quyết định ngẫu nhiên xiên phân (ArcX4-rODT) cho phép phân lớp hiệu quả dữ liệu gien có số chiều lớn và nhiễu. Giải thuật ArcX4-rODT xây dựng tuần tự tập hợp cây xiên phân ngẫu nhiên, cây xây dựng sau sẽ tập trung lên các mẫu bị phân lớp sai bởi các cây trước. Ý tưởng chính là mỗi cây thành viên sử dụng siêu phẳng phân chia dữ liệu hiệu quả tại mỗi nút của cây dựa trên phân tích biệt lập tuyến tính. Việc xây dựng cây xiên phân ngẫu nhiên vì thế tạo bộ phân lớp mạnh có khả năng làm việc tốt trên dữ liệu có số chiều lớn và nhiễu như dữ liệu gien. Các kết quả thử nghiệm trên các tập dữ liệu gien cho thấy rằng giải thuật ArcX4 cây quyết định ngẫu nhiên xiên phân chính xác hơn dựa trên tiêu chí về precision, recall, F1-measure và độ chính xác accuracy khi so sánh với rừng ngẫu nhiên của cây quyết định C4.5 và cả giải thuật SVM.

TÀI LIỆU THAM KHẢO

L. Breiman, J.H. Friedman, R.A. Olshen and C. Stone. *Classification and Regression Trees*. Wadsworth International, 1984.

L. Breiman. Bagging predictors. *Machine Learning 24(2)*:123–140, 1996.

L. Breiman. Arcing classifiers. *The annals of statistics*, 26(3): 801–849, 1998.

L. Breiman. Random forests. *Machine Learning 45(1)*:5–32, 2001.

W. Buntine. Learning classification trees. *Statistics and Computing 2*, 1992, pp. 63–73.

C.C. Chang and C.J. Lin. Libsvm – a library for support vector machines. 2001. http://www.csie.ntu.edu.tw/cjlin/libsvm.

T.N. Do, S. Lallich, N.K. Pham and P. Lenca. Classifying very-high-dimensional data with random forests of oblique decision trees. in *Advances in Knowledge Discovery and Management Vol. 292*, Springer-Verlag, 2009, pp. 39-55.

R.A. Fisher. The Use of Multiple Measurements in Taxonomic Problems. in *Annals of Eugenics*, No 7, 1936, pp. 179-188.

Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Computational Learning Theory*, 1995, pp. 23–37.

J. Friedman, T. Hastie and R. Tibshirani. Response to Mease and Wyner, Evidence Contrary to the Statistical View of Boosting. *Journal Machine Learning Research Vol. 9*, 2008, pp. 175-180.

A.J. Grove and D. Schuurmans. Boosting in the limit: Maximizing the margin of learned ensembles. In Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98), 1998, pp. 692–699.

L. Jinyan and L. Huiqing. Kent ridge bio-medical dataset repository. 2002, http://datam.i2r.a-star.edu.sg/datasets/krbd/.

S. Murthy, S. Kasif, S. Salzberg and R. Beigel. Oc1: Randomized induction of oblique decision trees. In Proceedings of the Eleventh National Conference on Artificial Intelligence, 1993, pp. 322–327.

J.R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

C.V. van Rijsbergen. *Information Retrieval*. Butterworth, 1979.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.

D. Wolpert. Stacked generalization. *Neural Networks 5*, 1992, pp. 241–259.

Q. Yang and X. Wu. 10 Challenging Problems in Data Mining Research. *Journal of Information Technology & Decision Making 5(4)*:597-604, 2006.