ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

PHAN THÀNH NHÂN - 20C29012

SỬ DỤNG TÍNH TOÁN SONG SONG ĐỂ LỰA CHỌN ĐẶC TRƯNG DỰA TRÊN TIÊU CHÍ TỈ LỆ VẾT

LUẬN VĂN THẠC SĨ

Tp. Hồ Chí Minh - 2022

ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

PHAN THÀNH NHÂN - 20C29012

SỬ DỤNG TÍNH TOÁN SONG SONG ĐỂ LỰA CHỌN ĐẶC TRƯNG DỰA TRÊN TIÊU CHÍ TỈ LỆ VẾT

LUẬN VĂN THẠC SĨ

CHUYÊN NGÀNH CƠ SỞ TOÁN CHO KHOA HỌC DỮ LIỆU

GIẢNG VIÊN HƯỚNG DẪN KHOA HỌC

TS. Nguyễn Thị Thu

Tp. Hồ Chí Minh - 2022

Lời cảm ơn

Tp.HCM, ngày 10 tháng 12 năm 2022

Tác giả

Phan Thành Nhân

Mục lục

Lời nói đầu

Trong kỷ nguyên của dữ liệu lớn, sự phát triển của dữ liệu đặt ra những thách thức đối với việc quản lý và sử dụng dữ liệu hiệu quả. Ví dụ: một tập dữ liệu gen có thể chứa hàng trăm nghìn đặc điểm [13]. Do đó, việc xử lý trực tiếp các tập dữ liệu như vậy có thể gặp sự khó khăn về kích thước. Hơn nữa, các đặc trưng dư thừa có thể làm giảm hiệu suất học tập của các thuật toán phân loại. Để giải quyết vấn đề này, nhiều kỹ thuật giảm chiều dữ liệu đã được phát triển [6, 10, 12, 21, 23] và chúng được phân loại thành các phương pháp trích xuất đặc trưng hoặc lựa chọn đặc trưng [18, 20]. Kỹ thuật trích xuất đặc trưng (ví dụ: Phân tích thành phần chính (Principal Component Analysis - PCA) [16], Phân tích phân biệt tuyến tính (Linear Discriminant Analysis - LDA) [16]) liên quan đến việc chiếu dữ liệu vào một không gian đối tượng mới với số chiều nhỏ hơn thông qua một vài bước biến đổi tuyến tính hoặc phi tuyến từ các đặc trưng gốc. Tuy nhiên, điều này tạo ra một loạt các đặc trưng mới mà không thể diễn giải trực tiếp. Hơn nữa, vì những cách tiếp cận đó sử dụng tất cả các đặc trưng có sẵn trong quá trình trích xuất đặc trưng nên nó không giúp giảm chi phí lưu trữ dữ liệu và chi phí thu thập dữ liệu trong tương lai. Mặt khác, các phương pháp lựa chọn đặc trưng ([15, 28],...) chỉ chọn một tập hợp con các đặc trưng hữu ích để xây dựng mô hình. Do đó, điều này giúp giữ các tính chất của các đặc trưng ban đầu trong khi giảm chi phí lưu trữ và thu thập dữ liệu trong tương lai bằng cách loại bỏ các đặc trưng không liên quan. Tuy nhiên, các dữ liệu từ các lĩnh vực khác nhau như khai thác văn bản, phân tích kinh doanh và sinh học, thường được đo bằng gigabyte hoặc terabyte với hàng triệu đặc trưng [4, 19]. Ví dụ: tập dữ liệu Amazon Review [25] là tập dữ liệu 34 gigabyte. Trong những trường hợp như vậy, hiệu suất của các kỹ thuật lựa chọn đặc trưng mới nhất có thể bị ảnh hưởng [19]. Điều này là do không gian tìm kiếm cho một tập hợp con các đặc trưng hữu ích bị tăng lên đáng kể. Một cách để giải quyết vấn đề này là sử dụng tính toán song song, cho phép sử dụng tốt hơn tài nguyên tính toán của máy tính bằng cách phân vùng dữ liệu và chạy các lựa chọn đặc trưng trên nhiều lõi cùng một lúc.

Trong đề tài luận văn cao học này, chúng tôi phát triển một phương pháp "Sử dụng tính toán song song để lựa chọn đặc trưng dựa trên tiêu chí tỉ lệ vết (Parallel feature selection based on the trace ratio criterion - PFST)"cho bài toán phân loại. Tiêu chí được sử dụng là một thước đo về khả năng tách lớp được sử dụng trong LDA, để đánh giá

tính hữu dụng của đặc trưng. Dựa trên tiêu chí này, PFST sẽ nhanh chóng tìm thấy các đặc trưng quan trọng từ một tập hợp các đặc trưng gốc của tập dữ liệu lớn bằng cách sử dụng sức mạnh của tính toán song song để thực hiện lựa chọn đặc trưng và loại bỏ các đặc trưng dư thừa. Sau khi các đặc trưng quan trọng nhất được đưa vào mô hình, chúng tôi đánh giá phương pháp thông qua các thử nghiệm khác nhau bằng cách sử dụng LDA làm mô hình phân loại. Thực nghiệm cho thấy rằng phương pháp của chúng tôi có thể chọn ra một tập hợp con nhỏ các đặc trưng trong thời gian ngắn, tiết kiệm thời gian hơn so với một số phương pháp khác. Ngoài ra, độ chính xác khi phân loại dựa trên các đặc trưng được lựa chọn bằng PFST cũng đạt độ chính xác cao và tốt hơn các phương pháp khác, và tốt hơn việc phân loại dựa trên tập các đặc trưng gốc ban đầu.

Nội dung khóa luận này bao gồm 3 chương. Trong đó,

Chương 1: Chương này giới thiệu tổng quan về lựa chọn đặc trưng và các nghiên cứu liên quan.

Chương 2: Chương này trình bày về tiêu chí vết - tiếu chí lựa chọn đặc trưng. Thuật toán PSFT - thuật toán lựa chọn đặc trưng song song dựa trên tiêu chí vết.

Chương 3: Chương này trình bày về các tập dữ liệu được sử dụng khi làm thực nghiệm. Thông tin về phần cứng được sử dụng và thảo luận kết quả

Cuối cùng: Phần kết luận và danh mục các tài liệu tham khảo.

Chương 1

Tổng quan

1.1 Lý do chọn đề tài

Trong kỷ nguyên của dữ liệu lớn, sự phát triển của dữ liệu đặt ra những thách thức đối với việc quản lý và sử dụng dữ liệu hiệu quả. Ví dụ: một tập dữ liệu gen có thể chứa hàng trăm nghìn đặc điểm [13]. Do đó, việc xử lý trực tiếp các tập dữ liệu như vậy có thể gặp sự khó khăn về kích thước. Hơn nữa, các đặc trưng dư thừa có thể làm giảm hiệu suất học tập của các thuật toán phân loại. Để giải quyết vấn đề này, nhiều kỹ thuật giảm chiều dữ liệu đã được phát triển [6, 10, 12, 21, 23] và chúng được phân loại thành các phương pháp trích xuất đặc trưng hoặc lựa chọn đặc trưng [18, 20]. Kỹ thuật trích xuất đặc trưng (ví dụ: Phân tích thành phần chính (Principal Component Analysis - PCA) [16], Phân tích phân biệt tuyến tính (Linear Discriminant Analysis - LDA) [16]) liên quan đến việc chiếu dữ liệu vào một không gian đối tượng mới với số chiều nhỏ hơn thông qua một vài bước biến đổi tuyến tính hoặc phi tuyến từ các đặc trưng gốc. Tuy nhiên, điều này tạo ra một loạt các đặc trưng mới mà không thể diễn giải trực tiếp. Hơn nữa, vì những cách tiếp cận đó sử dụng tất cả các đặc trưng có sẵn trong quá trình trích xuất đặc trưng nên nó không giúp giảm chi phí lưu trữ dữ liệu và chi phí thu thập dữ liệu trong tương lai. Mặt khác, các phương pháp lựa chọn đặc trưng ([15, 28],...) chỉ chọn một tập hợp con các đặc trưng hữu ích để xây dựng mô hình. Do đó, điều này giúp giữ các tính chất của các đặc trưng ban đầu trong khi giảm chi phí lưu trữ và thu thập dữ liệu trong tương lai bằng cách loại bỏ các đặc trưng không liên quan. Tuy nhiên, các dữ liệu từ các lĩnh vực khác nhau như khai thác văn bản, phân tích kinh doanh và sinh học, thường được đo bằng gigabyte hoặc terabyte với hàng triệu đặc trưng [4, 19]. Ví dụ: tập dữ liệu Amazon Review [25] là tập dữ liệu 34 gigabyte. Trong những trường hợp như vậy, hiệu suất của các kỹ thuật lựa chọn đặc trưng mới nhất có thể bị ảnh hưởng [19]. Điều này là do không gian tìm kiếm cho một tập hợp con các đặc trưng hữu ích bị tăng lên đáng kể. Một cách để giải quyết vấn đề này là sử dụng tính toán song song, cho phép sử dụng tốt hơn tài nguyên tính toán của

máy tính bằng cách phân vùng dữ liệu và chạy các lựa chọn đặc trưng trên nhiều lõi cùng một lúc.

Từ những lý do trên, trong đề tài luận văn cao học này, chúng tôi nghiên cứu và trình bày một phương pháp "Sử dụng tính toán song song để lựa chọn đặc trưng dựa trên tiêu chí tỉ lệ vết (Parallel feature selection based on the trace ratio criterion - PFST)"cho bài toán phân loại. Tiêu chí đánh giá mức độ hữu dụng của đặc trưng được sử dụng là một thước đo về khả năng tách lớp được sử dụng trong bài toán Linear discriminant analysis (LDA - chúng tôi sẽ trình bày bài toán này bên dưới). Dựa trên tiêu chí này, PFST sẽ nhanh chóng tìm thấy các đặc trưng quan trọng từ một tập hợp các đặc trưng gốc của tập dữ liệu lớn và loại bỏ các đặc trưng dư thừa. Bên cạnh đó, chúng tôi thiết lập và sử dụng tính toán song song để tối ưu nguồn tài nguyên của máy tính nhằm tăng hiệu suất tính toán và thời gian chạy. Cuối cùng, chúng tôi sẽ sử dụng LDA làm mô hình phân loại để đánh giá độ hiệu quả của thuật toán PFST. Bên cạnh đó, chúng tôi cũng so sánh với một số thuật toán khác. Kết quả thu được rất tốt, PFST có thời gian chạy và có độ chính xác cao hơn các phương pháp khác. Bên cạnh đó, tập đặc trưng thu được từ PFST cũng cho ra kết quả tốt hơn nếu sử dụng toàn bộ đặc trưng ban đầu.

1.2 Các công trình nghiên cứu khoa học có liên quan

Đối mặt với các thách thức về kích thước của dữ liệu ngày càng tăng, đã có nhiều nỗ lực trong hướng nghiên cứu về lựa chọn đặc trưng để phát triển các kỹ thuật mới. Bên cạnh các phương pháp lai để kết hợp các chiến lược lựa chọn đặc trưng khác nhau [1, 27], hầu hết các phương pháp lựa chọn đặc trưng có thể được chia thành ba loại.

Đầu tiên, cách tiếp cận "wrapper" dựa trên hiệu suất của một thuật toán học máy cụ thể để đánh giá tầm quan trọng của các đặc trưng được chọn. Một phương pháp "wrapper" điển hình sẽ tìm kiếm một tập con các đặc trưng dựa trên một thuật toán học máy trước, sau đó sẽ đánh giá chúng. Các bước này được lặp lại cho đến khi thỏa mãn một số tiêu chí dừng. Các phương pháp trong loại này thường rất tốn kém về chi phí tính toán vì việc đánh giá tập con các đặc trưng yêu cầu nhiều lần lặp lại. Mặc dùng nhiều các tiếp cận tìm kiếm được đề xuất chẳng hạn như thuật toán tìm kiếm best-first [2] và thuật toán di chuyền (genetic) [11]. Tuy nhiên, việc sử dụng các thuật toán này cho dữ liệu nhiều chiều vẫn không cho thấy sử cải thiện về chi phí tính toán.

Thứ hai, cách tiếp cận "filter" bao gồm các kỹ thuật đánh giá các tập hợp con đặc bằng việc xếp hạng với một số tiêu chí như tiêu chí thông tin [**?**, **?**], khả năng tái tạo [**?**, 22]. Các phương pháp này chọn các đặc trưng độc lập với thuật toán học máy và thường hiệu quả hơn về chi phí tính toán so với các phương pháp "wrapper" [19]. Tuy nhiên, vì không được tối ưu hóa cho bất kỳ thuật toán học máy mục tiêu nào, nên

chúng có thể không tối ưu cho một tuật toán học máy cụ thể.

Thứ ba, các phương pháp "embedded" sử dụng các tiêu chí độc lập để tìm ra tập con tối ưu cho một tập hợp nhất định. Sau đó, một thuật toán học máy được sử dụng để lựa chọn tập con tối ưu cuối cùng trong số các tập con tối ưu trên các tập hợp khác nhau. Vì thế, chúng hiệu quả hơn về chi phí tính toán so với các phương pháp "wrapper" vì chúng không đánh giá đặc trưng dựa trên việc lặp lại các tập con đặc trưng. Ngoài ra, chúng cũng được huấn luyện từ các thuật toán học máy. Vì thế, chúng có thể được coi như sự đánh đổi giữa phương pháp "filter" và phương pháp "wrapper" [19].

Mặc dù, cho đến này, các nhà khoa học đã nỗ lực rất nhiều trong hướng nghiên cứu lựa chọn đặc trưng, nhưng dữ liệu từ các trường, các ngành khác nhau có thể quá phong phú ngay cả đối với các phương pháp "filter"hiệu quả về chi phí tính toán. Điều này đã thúc đẩy nhiều nghiên cứu khác trong việc lựa chọn đặc trưng song song. Một số phương pháp đã được đề xuất trong [**?**, **?**, 6, 10, 23] sử dụng quy trình xử lý song song để đánh giá đồng thời nhiều đặc trưng. Tuy nhiên, các thuật toán này yêu cầu quyên truy cập vào toàn bộ dữ liệu. Mặc khác, trong trong [29], các tác giả đã đề xuất một thuật toán lựa chọn đặc trưng song song cho hồi quy logistic dựa trên framework MapReduce và các đặc trưng được đánh giá thông qua hàm mục tiêu của mô hình hồi quy logistic. Trong khi đó, các tác giả của bài báo [31] đã đề xuất *Song song, Tiến–Lùi với thuật toán Tỉa (Parallel, Forward–Backward with Pruning algorithm)* (PFBP) để lựa chọn đặc trưng bằng cách bỏ sớm một số đặc trưng trong các lần lặp lại tiếp theo và sớm trả ra kết quả đặc trưng tốt nhất trong mỗi lần lặp. Tuy nhiên, các tiếp cận này yêu cầu tính toán bootstrap của p-giá trị, rất tốn kém chi phí tính toán. Trong bài báo [33], Zhao và các cộng sự đã giới thiệu một thuật toán lựa chọn đặc trưng song song để chọn các đặc trưng dựa trên khả năng giải thích phương sai của dữ liệu. Tuy nhiên, theo các tiếp cận của họ, việc xác định số lượng các đặc trưng trong mô hình dựa trên việc chuyển đổi các nhãn phân loại thành các giá trị số và sử dụng tổng bình phương sai số. Việc tính tổng bình phương sai số đòi hỏi phải điều chỉnh mô hình và do đó thuật toán vẫn còn tốn kém rất nhiều chi phí tính toán.

Chương 2

Kiến thức nền tảng

2.1 Quy trình của lớp các bài toán phân loại

Để giải quyết một bài toán phân loại, ta cần phải hiểu dữ liệu, tính chất các đặc trưng và quá trình chọn các đặc trưng phù hợp với mô hình cũng cần phải có một quy trình rõ ràng.

Bước đầu tiên là xác định bài toán, nghĩa là xác định nhãn của bài toán và xây dựng tập dữ liệu (data collection). Phải đảm bảo là tập dữ liệu phải liên quan đến bài toán được mô hình hóa. Bước này rất quan trọng vì sẽ ảnh hưởng rất nhiều đến kết quả phân loại. Ngoài ra, chỉ những đặc trưng có thông tin hữu ích về bài toán là nên được sử dụng. Trong một số trường hợp, khi gặp khó khăn về kiến thức cũng như khả năng thu thập dữ liệu. Chúng ta có thể sử dụng phương pháp brute-force để thay thế. Brute-force là một phương pháp giải quyết vấn đề bằng cách thực hiện tất cả các giải pháp có thể có và chọn ra giải pháp tốt nhất. Nó được sử dụng khi không có một thuật toán cụ thể nào có thể giải quyết vấn đề hoặc khi không có đủ kiến thức về vấn đề để thiết kế một giải pháp tối ưu. Tuy nhiên, phương pháp này có thể rất tốn kém và thời gian, đặc biệt là đối với các vấn đề có kích thước lớn. Do vậy, trong trường hợp này sẽ có một lượng rất lớn các biến được đo, xử lý và thêm vào tập dữ liệu. Tuy nhiên, hi vọng, trong tương lai có thể tác được các đặc trưng tốt nhất và phù hợp nhất với bài toán.

Nếu vấn đề về dữ liệu có thể được giải quyết, chúng ta sẽ nên bước tiếp theo trong quy trình phân loại là tiền xử lý dữ liệu (data pre-processing). Ở bước này, vấn đề chính là thiếu dữ liệu (missing data) và dữ liệu ngoại lai (outlier) cần phải được xử lý. Có một vài phương pháp phân tích thống kê [**?**, **?**] để có thể dử lý các vấn đề này. Hơn nữa, đây là bước mà số lượng các đặc trưng của bài toán có thể giảm đi bằng việc áp dụng thuật toán lựa chọn đặc trưng.

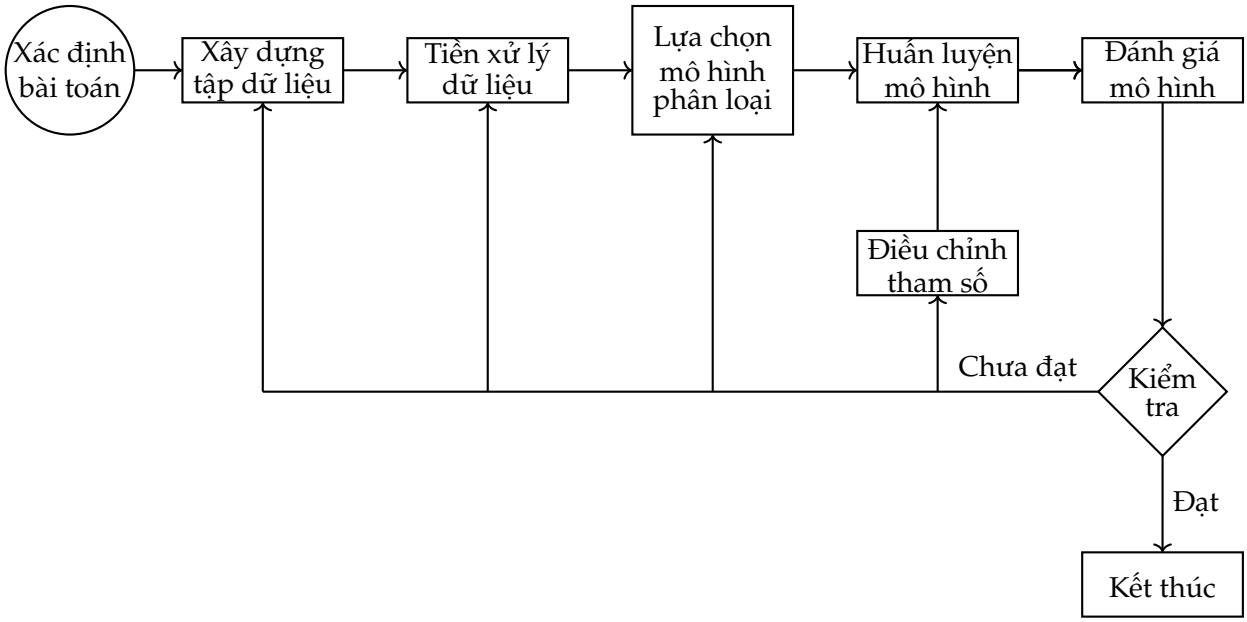
Vấn đề này có thể được giải quyết, đưa chúng ta đến bước tiếp theo trong quy trình

phân loại: tiền xử lý dữ liệu. Ở giai đoạn này, các vấn đề chính như giá trị bị thiếu và phát hiện ngoại lệ nên được xử lý. Có nhiều phương pháp phân tích thống kê để giải quyết các vấn đề này [1, 26]. Ngoài ra, đây là giai đoạn trong vấn đề mà số lượng đặc trưng của vấn đề có thể được giảm bằng thuật toán lựa chọn đặc trưng.

Bước tiếp theo là lựa chọn thuật toán phân loại. Có rất nhiều các thuật tóa phân loại, và mặc dù các thuật toán rất đa dạng và khác nhau về ý tưởng, nhưng không dễ dàng chọn được thuật toán nào là tốt nhất cho một bài toán cụ thể. Do đó, việc thử nghiệm và so sánh một số thuật toán là một cách làm tương đối phổ biến, mục tiêu cuối cùng là lựa chọn được thuật toán cho kết quả tốt nhất [**?**].

Việc đánh giá các thuật toán phân loại thường dựa trên độ chính xác của việc dự đoán. Một kỹ thuật điển hình là chia dữ liệu thành hai phần để huấn luyện mô hình và sử dụng phần còn lại để kiểm tra độ chính xác. Tuy nhiên, quy trình này thường dẫn đến kết quả không tốt khi áp dụng vào tập dữ liệu bên ngoài. Do đó, để giảm thiểu sai số, một số kỹ thuật phức tạp hơn như kiểm tra chéo (cross-validation) [**?**] có thể được sử dụng.

Để có cái nhìn tổng quan, chúng tôi xin trình bày theo sơ đồ khối của toàn bộ quy trình của một bài toán phân loại.



Trong toàn bộ quy trình, nếu có bất kì bước nào không tốt, quy trình phải quy trở lại bước trước đó. Có nhiều nguyên nhân có thể ảnh hưởng đến hiếu suất của một bài toán phân loại [**?**] như:

- Đặc trưng phù hợp không được lựa chọn tốt.
- Tập dữ liệu không đủ, ít mẫu quan sát.

- Số lượng các đặc trưng quá nhiều.
- Kỹ thuật tiền xử lý dữ liệu chưa được tốt.
- Mô hình phân loiaj được chọn không phù hợp cho vấn đề hoặc cần điều chỉnh tham số.

Vì vậy, không thể chỉ ra rõ ràng bước nào trong quy trình cần trở lại. Tuy nhiên, mục tiêu cuối cùng là giải quyết bài toán phân loại đạt kết quả tốt nhất cho dữ liệu chưa được quan sát. Đây là bài toán khó, và mỗi bước thường được thực hiện trong thời gian dài, thông thường chúng ta cần phải liên tục thực hiện nhiều thử nghiệm mới để cải thiện khả năng dự đoán của mô hình.

2.2 Lựa chọn đặc trưng

Ngày nay, các phương pháp học máy xuất hiện ngày càng nhiều và rất mạnh mẽ để giải quyết các bài toán dữ liệu lớn. Các mô hình học máy phổ biến hiện nay như cây quyết định (decision tree), rừng ngẫu nhiên (random forest), SVM, KNN,... đều là những mô hình mạnh mẽ, linh hoạt và có độ chính xác cao cả trong bài toán phân loại hay bài toán hồi quy. Tuy nhiên, bên cạnh việc áp dụng các mô hình học máy, chúng ta cần phải chuẩn hóa dữ liệu tốt, bởi vì dữ liệu là nguyên liệu để mô hình học máy học dựa trên đó. Kết quả của một bài toán sử dụng học máy có thể sẽ được cải thiện rõ rệt nếu có bước chuẩn bị dữ liệu tốt. Và việc lựa chọn đặc trưng là một kĩ thuật quan trọng bênh cạnh việc trích xuất đặc trưng hay biến đổi đặc trưng. Trong phạm vi của khóa luận này, chúng tôi chỉ xin nhắc lại khái quát về lựa chọn đặc trưng và các thuật toán liên quan.

2.2.1 Lựa chọn đặc trưng là gì?

Để xây dựng mô hình, chúng ta sẽ cần đến thông tin. Thông tin đến tự những bộ dữ liệu, nhưng với sự bùng nổ của dữ liệu lớn (bigdata), dữ liệu dường như trở nên quá nhiều, khiến việc xây dựng mô hình gặp nhiều khó khăn như tăng chi phí tính toán, quá nhiều đặc trưng có thể dẫn tới hiện tượng quá khớp (overfitting) - là hiện tượng mô hình hoạt động tốt trên tập huấn luyện (training set), nhưng tệ trên tập thử nghiệm (testing set), một số đặc trưng có thể gây nhiễu và làm giảm chất lượng mô hình,...

Có rất nhiều thuật toán lựa chọn đặc trưng đã được phát triển từ rất lâu đến tận thời điểm hiện tại. Trong đó, lựa chọn tiến (forward feature selection), lựa chọn lùi (backward feature selection) và lựa chọn từng bước (stepwise feature selection) là ba thuật toán rất phổ biến. Trong phần này, chúng tôi sẽ tóm tắt lại ba kĩ thuật này, và trình bày chi tiết cách chúng tôi sử dụng trong các thuật toán 1, 2, và 3.

Từ đây, để thuận tiện trong việc trình bày, chúng tôi sẽ sử dụng "ffs" để chỉ thuật toán lựa chọn tiến, "bfs" để chỉ thuật toán lựa chọn lùi và "sfs" để chỉ thuật toán lựa chọn từng bước.

2.2.2 Thuật toán lựa chọn tiến

Ffs được sử dụng rất rộng rãi vì sự hiệu quả trong việc tính toán của nó, cùng với khả năng xử lý hiệu quả các vấn đề bao gồm việc số lượng đặc trưng vượt quá số lượng quan sát. Tuy nhiên, một số đặc trưng có thể xuất hiện dư thừa sau khi đã lựa chọn các đặc trưng khác. Về các điều kiện đủ để lựa chọn tiến nhằm khôi phục mô hình ban đầu và tính ổn định của nó, chúng tôi tham khảo từ [30] và [7]. Dưới đây là chi tiết thuật toán

Thuật toán 1. Lựa chọn tiến
(Forward Feature Selection)

Input: Một tập dữ liệu gồm p đặc trưng $f_1, f_2, ..., f_p$, tham số tiến là α.
Output: Một tập hợp R gồm các đặc trưng được chọn.

$R \leftarrow \emptyset$
$S \leftarrow \{f_1, f_2, \ldots, f_p\}$
while True **do**
    $f_j \leftarrow$ đặc trưng hữu ích nhất trong S
    if mô hình được cải thiện tốt hơn một lượng là α sau khi thêm vào f_j **then**
        $R \leftarrow R \cup \{f_j\}$
        $S \leftarrow S \setminus \{f_j\}$
    else
        return R
    end if
end while

2.2.3 Thuật toán lựa chọn lùi

Bfs [15] được trình bày chi tiết trong thuật toán 2. Bắt đầu với toàn bộ đặc trưng, sau đó lần lượt loại bỏ các đặc trưng ít hữu ích nhất từ từ, mỗi lần một đặc trưng. Yêu cầu của bfs bảo đảm những đặc trưng dư thừa được loại bỏ khỏi mô hình. Tuy nhiên, bfs thì có tốc độ tính toán khá chậm, và chậm hơn nhiều khi so với fsf.

Thuật toán 2. Lựa chọn lùi

(Backward Feature Selection)

Input: Một tập dữ liệu gồm p đặc trưng $f_1, f_2, ..., f_p$, tham số lùi là β.
Output: Một tập hợp R gồm các đặc trưng được chọn.

$R \leftarrow \{f_1, f_2, \ldots, f_p\}$
while True, **do**
    $f_j \leftarrow$ là đặc trưng ít hữu ích nhất trong R.
    if giá trị mất mát của mô hình sau khi loại f_j là nhỏ hơn β **then**
        $R \leftarrow R \setminus \{f_j\}$
    else
        return R
    end if
end while

2.2.4 Thuật toán lựa chọn từng bước

Ngoài hai thuật toán được nêu ra ở trên, một thuật toán khác, là phiên bản kết hợp của cả thuật toán lựa chọn tiến và thuật toán lựa chọn lùi là thuật toán lựa chọn từng bước, được trình bày chi tiết trong thuật toán 3. Trong cách tiếp cận này, các đặc trưng được thêm vào mô hình một cách có tuần tự như trong lựa chọn tiến. Tuy nhiên, sau khi thêm vào các đặc trưng mới, phương pháp này có thể loại bỏ bất kỳ đặc trưng nào mà có vẻ không còn phù hợp.

Thuật toán 3. Lựa chọn từng bước
(Stepwise Feature Selection)

Input: Một tập hợp gồm p đặc trưng $f_1, f_2, ..., f_p$, tham số tiến là α, tham số lùi là β.
Output: Một tập hợp R gồm các đặc trưng được chọn.

$R \leftarrow \emptyset$
$S \leftarrow \{f_1, f_2, \ldots, f_p\}$
while True, **do**
    $f_j \leftarrow$ là đặc trưng hữu ích nhất trong S
    if mô hình cải thiện hơn một lượng là α sau khi thêm f_j **then**
        $R \leftarrow R \cup \{f_j\}$
        $S \leftarrow S \setminus \{f_j\}$
        while True **do**
            $f_k \leftarrow$ là đặc trưng ít hữu ích nhất trong R
            if hiệu suất của mô hình giảm một lượng nhỏ hơn β sau khi loại bỏ f_j. **then**
                $R \leftarrow R \setminus \{f_k\}$

```
12:             else break
13:             end if
14:         end while
15:     else
16:         return R
17:     end if
18: end while
```

2.3 Các nghiên cứu liên quan

Đối mặt với các thách thức về kích thước của dữ liệu ngày càng tăng, đã có nhiều nỗ lực trong hướng nghiên cứu về lựa chọn đặc trưng để phát triển các kỹ thuật mới. Bên cạnh các phương pháp lai để kết hợp các chiến lược lựa chọn đặc trưng khác nhau [1, 27], hầu hết các phương pháp lựa chọn đặc trưng có thể được chia thành ba loại.

Đầu tiên, cách tiếp cận "wrapper" dựa trên hiệu suất của một thuật toán học máy cụ thể để đánh giá tầm quan trọng của các đặc trưng được chọn. Một phương pháp "wrapper" điển hình sẽ tìm kiếm một tập con các đặc trưng dựa trên một thuật toán học máy trước, sau đó sẽ đánh giá chúng. Các bước này được lặp lại cho đến khi thỏa mãn một số tiêu chí dừng. Các phương pháp trong loại này thường rất tốn kém về chi phí tính toán vì việc đánh giá tập con các đặc trưng yêu cầu nhiều lần lặp lại. Mặc dùng nhiều các tiếp cận tìm kiếm được đề xuất chẳng hạn như thuật toán tìm kiếm best-first [2] và thuật toán di chuyền (genetic) [11]. Tuy nhiên, việc sử dụng các thuật toán này cho dữ liệu nhiều chiều vẫn không cho thấy sử cải thiện về chi phí tính toán.

Thứ hai, cách tiếp cận "filter" bao gồm các kỹ thuật đánh giá các tập hợp con đặc bằng việc xếp hạng với một số tiêu chí như tiêu chí thông tin [**?**, **?**], khả năng tái tạo [**?**, 22]. Các phương pháp này chọn các đặc trưng độc lập với thuật toán học máy và thường hiệu quả hơn về chi phí tính toán so với các phương pháp "wrapper" [19]. Tuy nhiên, vì không được tối ưu hóa cho bất kỳ thuật toán học máy mục tiêu nào, nên chúng có thể không tối ưu cho một tuật toán học máy cụ thể.

Thứ ba, các phương pháp "embedded" sử dụng các tiêu chí độc lập để tìm ra tập con tối ưu cho một tập hợp nhất định. Sau đó, một thuật toán học máy được sử dụng để lựa chọn tập con tối ưu cuối cùng trong số các tập con tối ưu trên các tập hợp khác nhau. Vì thế, chúng hiệu quả hơn về chi phí tính toán so với các phương pháp "wrapper" vì chúng không đánh giá đặc trưng dựa trên việc lặp lại các tập con đặc trưng. Ngoài ra, chúng cũng được huấn luyện từ các thuật toán học máy. Vì thế, chúng có thể được coi như sự đánh đổi giữa phương pháp "filter" và phương pháp "wrapper" [19].

Mặc dù, cho đến này, các nhà khoa học đã nỗ lực rất nhiều trong hướng nghiên cứu lựa chọn đặc trưng, nhưng dữ liệu từ các trường, các ngành khác nhau có thể quá phong phú ngay cả đối với các phương pháp "filter"hiệu quả về chi phí tính toán. Điều này đã thúc đẩy nhiều nghiên cứu khác trong việc lựa chọn đặc trưng song song. Một số phương pháp đã được đề xuất trong [**?**, **?**, 6, 10, 23] sử dụng quy trình xử lý song song để đánh giá đồng thời nhiều đặc trưng. Tuy nhiên, các thuật toán này yêu cầu quyên truy cập vào toàn bộ dữ liệu. Mặc khác, trong trong [29], các tác giả đã đề xuất một thuật toán lựa chọn đặc trưng song song cho hồi quy logistic dựa trên framework MapReduce và các đặc trưng được đánh giá thông qua hàm mục tiêu của mô hình hồi quy logistic. Trong khi đó, các tác giả của bài báo [31] đã đề xuất *Song song, Tiến–Lùi với thuật toán Tỉa (Parallel, Forward–Backward with Pruning algorithm)* (PFBP) để lựa chọn đặc trưng bằng cách bỏ sớm một số đặc trưng trong các lần lặp lại tiếp theo và sớm trả ra kết quả đặc trưng tốt nhất trong mỗi lần lặp. Tuy nhiên, các tiếp cận này yêu cầu tính toán bootstrap của p-giá trị, rất tốn kém chi phí tính toán. Trong bài báo [33], Zhao và các cộng sự đã giới thiệu một thuật toán lựa chọn đặc trưng song song để chọn các đặc trưng dựa trên khả năng giải thích phương sai của dữ liệu. Tuy nhiên, theo các tiếp cận của họ, việc xác định số lượng các đặc trưng trong mô hình dựa trên việc chuyển đổi các nhãn phân loại thành các giá trị số và sử dụng tổng bình phương sai số. Việc tính tổng bình phương sai số đòi hỏi phải điều chỉnh mô hình và do đó thuật toán vẫn còn tốn kém rất nhiều chi phí tính toán.

Chương 3

Tiêu chí vết - Tiêu chí lựa chọn đặc trưng

3.1 Tiêu chí vết

Chương này sẽ trình bày chi tiết động cơ của việc sử dụng tiêu chí vết cho việc lựa chọn đặc trưng.

Tiêu chí vết là một thước đo hữu ích để phân lớp hữu ích để lựa chọn đặc trưng cho lớp bài toán toán phân loại (chi tiết hơn trong [9, 16]). Có nhiều phiên bản định nghĩa tiêu chí vết, chúng là tương đương nhau. Tuy nhiên, để phù hợp với lớp bài toán phân loại, chúng tôi sẽ giới thiệu định nghĩa sau

Định nghĩa 3.1.1. Cho một tập dữ liệu gồm C lớp, và có n_i quan sát cho lớp thứ i, và đặt $\mathbf{x}_{ij}$ là mẫu thứ j của lớp thứ i. Khi đó, ta có định nghĩa tiêu chí vết là

$$trace(S_w^{-1} S_b), \tag{3.1.1}$$

trong đó,

$$S_b = \sum_{i=1}^{C} n_i (\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})', \tag{3.1.2}$$

và

$$S_w = \sum_{i=1}^{C} \sum_{j=1}^{n_i} (\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)(\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)'. \tag{3.1.3}$$

S_b còn được gọi là ma trận phân tán giữa các lớp (between-class scatter matrix) và S_w là ma trận tán xạ trong lớp.

Ở đây A' là ma trận chuyển vị của ma trận A, và $\bar{\boldsymbol{x}}_i$ là giá trị trung bình của lớp thứ

i, và $\bar{\boldsymbol{x}}$ là giá trị trung bình của toàn bộ dữ liệu. Nghĩa là,

$$\bar{\boldsymbol{x}}_i = \frac{1}{n_i}\sum_{j=1}^{n_i}(\bar{\boldsymbol{x}}_{ij} - \bar{\boldsymbol{x}}_i)$$
$$\bar{\boldsymbol{x}} = \frac{1}{\sum_{i=1}^{C} n_i}\sum_{i=1}^{C}\sum_{j=1}^{n_i}(\bar{\boldsymbol{x}}_{ij} - \bar{\boldsymbol{x}}_i).$$

Từ đây, ta có kết quả sau

Định lý 3.1.2.

$$trace(S_w^{-1}S_b) = \sum_{i=1}^{C} n_i(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})' S_w^{-1}(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}}) \tag{3.1.4}$$

Chứng minh.

$$\begin{aligned}
trace(S_w^{-1}S_b) &= trace\left(S_w^{-1}\sum_{i=1}^{C} n_i(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})'\right) \\
&= trace\left(\sum_{i=1}^{C} n_i S_w^{-1}(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})'\right) \\
&= \sum_{i=1}^{C} n_i\, trace\left(S_w^{-1}(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})'\right) \\
&= \sum_{i=1}^{C} n_i\, trace\left((\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})' S_w^{-1}(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})\right) \\
&= \sum_{i=1}^{C} n_i(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})' S_w^{-1}(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})
\end{aligned}$$

□

Từ định lý trên, ta có thể thấy rằng tiêu chí vết 3.1.4 có thể được xem là tổng bình phương khoảng cách Mahalanobis từ trung bình của mỗi class đến trung bình của toàn bộ dữ liệu. Hơn nữa khi số lượng lớp là hai, tiêu chí 3.1.4 có thể được xem là ước lượng thực nghiệm của phân kỳ Kullback–Leibler cho hai phân phối chuẩn có cùng ma trận hiệp phương sai (covariance matrix) [17].

Vì tiêu chí này đo khả năng tách biệt của các lớp, nên chúng tôi sẽ định hướng vào việc tối đa nó. Để đơn giản ký hiệu, chúng tối sẽ viết $\{R, f\}$ thay cho $R \cup \{f\}$, và $t_{R,f}$ thay cho $t_{\{R,f\}}$. Ta có định lý sau

Định lý 3.1.3. *Với R là tập hợp các đặc trưng được chọn, và f là một đặc trưng tùy ý mà không nằm trong R. Đặt S_{Rf} là giá trị của S_w khi $\{R, f\}$ là tập hợp các đặc trưng được chọn, và S_R là giá trị của S_w khi R là tập hợp các đặc trưng được chọn*

$$S_{Rf} = \begin{pmatrix} S_R & v \\ v' & u \end{pmatrix}, \tag{3.1.5}$$

với $u \in \mathbb{R}^+, v \in \mathbb{R}^{|R|\times 1}$, $|R|$ *số lượng phần tử trong R.*
Nếu $u - v'S_R^{-1}v > 0$ *thì*

$$t_{R,f} \geq t_R. \tag{3.1.6}$$

Ngược lại, nếu $u - v'S_R^{-1}v < 0$ *thì*

$$t_{R,f} < t_R. \tag{3.1.7}$$

Chứng minh. Đặt R là tập hợp các đặc trưng đưa vào mô hình. Đặt $\bar{\boldsymbol{x}}_{iR}, \bar{\boldsymbol{x}}_R$ lần lượt là trung bình của lớp thứ i, và trung bình toàn bộ dữ liệu của tập hợp R. Tương tự, đặt $\bar{\boldsymbol{x}}_{iRf}, \bar{\boldsymbol{x}}_{Rf}$ lần lượt là trung bình của lớp thứ i và trung bình của toàn bộ dữ liệu toàn bộ dữ liệu của tập hợp $\{R, f\}$. Khi đó, ta có

$$\bar{\boldsymbol{x}}_{iRf} = \begin{pmatrix} \bar{\boldsymbol{x}}_{iR} \\ \bar{\boldsymbol{x}}_{if} \end{pmatrix}, \quad \bar{\boldsymbol{x}}_{Rf} = \begin{pmatrix} \bar{\boldsymbol{x}}_{R} \\ \bar{\boldsymbol{x}}_{f} \end{pmatrix}, \tag{3.1.8}$$

với $\bar{\boldsymbol{x}}_{if}, \bar{\boldsymbol{x}}_f$ lần lượt là trung bình lớp thứ i và trung bình của toàn bộ dữ liệu của đặc trưng f.

Nếu $u \neq v'S_R^{-1}v$ thì $M = (u - v'S_R^{-1}v)^{-1}$ tồn tại. Do đó,

$$S_{Rf}^{-1} = \begin{pmatrix} S_R^{-1} + S_R^{-1}vMv'S_R^{-1} & -S_R^{-1}vM \\ -Mv'S_R^{-1} & M \end{pmatrix} \tag{3.1.9}$$

Tiếp theo, đặt

$$\begin{aligned} \eta_{iRf} &= (\bar{\boldsymbol{x}}_{iRf} - \bar{\boldsymbol{x}}_{Rf})'S_{Rf}^{-1}(\bar{\boldsymbol{x}}_{iRf} - \bar{\boldsymbol{x}}_{Rf}) \\ &= \eta_{iR} + (\bar{\boldsymbol{x}}_{iR} - \bar{\boldsymbol{x}}_R)'S_R^{-1}vMv'S_R^{-1}(\bar{\boldsymbol{x}}_{iR} - \bar{\boldsymbol{x}}_R) \\ &- 2(\bar{\boldsymbol{x}}_{iR} - \bar{\boldsymbol{x}}_R)'S_R^{-1}vM(\bar{\boldsymbol{x}}_{if} - \bar{\boldsymbol{x}}_f) \\ &+ (\bar{\boldsymbol{x}}_{if} - \bar{\boldsymbol{x}}_f)'M(\bar{\boldsymbol{x}}_{if} - \bar{\boldsymbol{x}}_f), \end{aligned}$$

với $\eta_{iR} = (\bar{\boldsymbol{x}}_{iR} - \bar{\boldsymbol{x}}_R)'S_R^{-1}(\bar{\boldsymbol{x}}_{iR} - \bar{\boldsymbol{x}}_R)$. Để ý rằng $M \in \mathbb{R}, M \neq 0$. Do đó,

$$\begin{aligned} \frac{\eta_{iRf} - \eta_{iR}}{M} &= (\bar{\boldsymbol{x}}_{iR} - \bar{\boldsymbol{x}}_R)'S_R^{-1}vv'S_R^{-1}(\bar{\boldsymbol{x}}_{iR} - \bar{\boldsymbol{x}}_R) \\ &- 2(\bar{\boldsymbol{x}}_{iR} - \bar{\boldsymbol{x}}_R)'S_R^{-1}v(\bar{\boldsymbol{x}}_{if} - \bar{\boldsymbol{x}}_f) \\ &+ (\bar{\boldsymbol{x}}_{if} - \bar{\boldsymbol{x}}_f)'(\bar{\boldsymbol{x}}_{if} - \bar{\boldsymbol{x}}_f). \end{aligned}$$

Áp dụng bất đẳng thức Cauchy-Schwarz, $||a||^2 + ||b||^2 \geq 2\langle a, b\rangle$, với $a = (\bar{\boldsymbol{x}}_{iR} - \bar{\boldsymbol{x}}_R)S_R^{-1}v$, và $b = (\bar{\boldsymbol{x}}_{if} - \bar{\boldsymbol{x}}_f)$, ta thấy rằng

$$\frac{\eta_{iRf} - \eta_{iR}}{M} \geq 0.$$

Vì $t_{R,f} - t_R = \sum_{i=1}^{C}(\eta_{iRf} - \eta_{iR})$, nên nếu $u - v'S_R^{-1}v > 0$ thì $t_{R,f} \geq t_R$,, ngược lại nếu $u - v'S_R^{-1}v < 0$ thì $t_{R,f} < t_R$. □

Ý nghĩa của định lý 3.1.3 là việc cải thiến hay suy thoái của một mô hình sau khi thêm một đặc trưng không chỉ phụ thuộc vào bản thân các đặc trưng mà còn phụ thuộc vào sự tương quan giữa đặc trưng đó với các đặc trưng khác đã có trong mô hình. Hơn nữa, từ phương trình 3.1.7, ta có thể thấy rằng thêm một đặc trưng có thể làm giảm đi khả năng phân lớp của một mô hình. Vì thế, lựa chọn đặc trưng là cần phải loại bỏ các đặc trưng mà đóng góp của chúng cho hiệu suất của mô hình là không đáng kể. Thuộc tính này là mong muốn và không phải tất cả các tiêu chí lựa chọn đặc trưng đều có thể đáp ứng được. Ví dụ, với trung bình bình phương sai số (mean square error - MSE), việc thêm vào các đặc trưng, dù có dư thừa hay không cũng không làm tăng MSE. Định lý dưới đây sẽ làm rõ điều này.

Định lý 3.1.4. *Đặt MSE_X là giá trị trung bình bình phương sai số khi ta hồi quy Y dựa trên X. Giả sử T chứa các đặc trưng tùy ý được thêm vào mô hình và đặt*

$$U = (X \quad T). \tag{3.1.10}$$

Đặt MSE_U là giá trị trung bình bình phương sai số khi ta hồi quy Y dựa trên U. Khi đó, nếu $M = (T'T - T'X(X'X)^{-1}X'T)^{-1}$ tồn tại thì $MSE_U \leq MSE_X$.

Chứng minh. Đặt $H = X(X'X)^{-1}X'$, khi đó ta có $H^2 = H$, nghĩa là H là ma trận lũy đẳng, do đó $I - H$ cũng là ma trận lũy đẳng, thật vậy

$$(I - H)^2 = I - 2H + H^2 = I - 2H + H = I - H.$$

Do đó, tổng bình phương sai số khi hôi quy Y trên X là

$$SSE_X = (Y - \hat{Y}_X)'(Y - \hat{Y}_X) = Y'(I - H)Y$$

với $\hat{Y}_X$ là vector giá trị dự đoán của mô hình khi hồi quy Y trên X.

Chú ý rằng M là ma trận đối xứng, khi đó đặt $\hat{Y}_U$ là vector giá trị dự đoán của mô hình khi hồi quy Y trên U. Do đó,

$$\begin{aligned}
&(U'U)^{-1}U'Y \\
&= HY + HTMT'HY - HTMT'Y - TMT'HY + TMT'Y \\
&= HY - (I - H)TMT' + (I - H)TMT'Y.
\end{aligned}$$

Suy ra

$$\begin{aligned}
&Y - \hat{Y}_U \\
&= Y - \hat{Y}_X + (I - H)TMT'HY - (I - H)TMT'Y \\
&= Y - \hat{Y}_X - (I - H)TMT'(I - H)Y
\end{aligned}$$

Nên tổng bình phương sai số khi hồi quy Y trên U là

$$SSE_U = (Y - \hat{Y}_U)'(Y - \hat{Y}_U)$$

$$
\begin{aligned}
&= SSE_X - 2Y'(I-H)TMT'(I-H)(Y-\hat{Y}_X) \\
&\quad + Y'(I-H)TMT'(I-H)TMT'(I-H)Y
\end{aligned}
$$

Hơn nữa, ta cũng có $(I-H)TMT'$ là ma trận lũy đẳng, và $Y'(I-H)TMT'(I-H)Y \geq 0$. Vì thế,

$$
\begin{aligned}
&Y'(I-H)TMT'(I-H)TMT'(I-H)Y \\
&= Y'(I-H)TMT'(I-H)Y \\
&\leq 2Y'(I-H)TMT'(I-H)Y.
\end{aligned}
$$

Suy ra, $SSE_U \leq SSE_X$. Lại có $U = (XT)$ có nghĩa là số lượng mẫu trên X bằng với số lượng mẫu trên T, nên $MSE_U \leq MSE_X$. □

3.2 Thuật toán lựa chọn đặc trưng song song dựa trên tiêu chí vết

Trong mục này, chúng tôi sẽ trình bày chi tiết về thuật toán lựa chọn đặc trưng song song dựa trên tiêu chí vết(PFST). Hơn nữa, để có được các thuộc tính mong muốn của tiêu chí vết như đã đề cập ở trên, phương pháp này sẽ dựa trên các nhận xét sau.

Đầu tiên, như đã trình bày trong chương 1, việc lựa chọn tiến nhanh hơn lựa chọn lùi bởi vì nó liên tục thêm các đặc trưng phù hợp nhất vào mô hình. Tuy nhiên, ta vẫn phải chịu một chi phí tính toán tương đối lớn. Một cách tiềm năng để tăng tốc độ xử lý của quá trình này là áp dụng thuật toán Early Dropping (Early Dropping heuristic) [5]. Thuật toán này sẽ loại bỏ khỏi trong các lần lặp lại tiếp theo các đặc trưng được cho là không có khả năng làm tăng hiệu suất của mô hình. Trong bài báo [5], nhóm tác giả đã đánh giá khả năng đó bằng trị số p ($p-$value). Tuy nhiên, việc tính toán trị số p đòi hỏi sử dụng bootstrap rất tốn kém chi phí tính toán. Chúng tôi sẽ tiếp cận theo một hướng khác, chúng tôi sử dụng thuật toán Early Dropping và dùng tiêu chí vết để làm công cụ đánh giá. Điều này không yêu cầu phải lặp lại nhiều lần trong cả dữ liệu cho mỗi bước tiến như việc sử dụng trị số p.

Thứ hai, lựa chọn đặc trưng từng bước dường như là một biện pháp khắc phục sai số trong lựa chọn tiến. Lựa chọn đặc trưng từng bước thêm các đặc trưng vào mô hình một cách tuần tự như trong lựa chọn tiến. Hơn nữa, sau khi thêm vào một đặc trưng mới, cách tiếp cận này sẽ loại bỏ khỏi mô hình các đặc trưng mà không còn quan trọng dựa theo tiêu chí lựa chọn đặc trưng đang được sử dụng. Tuy nhiên, điều này sẽ làm tăng chi phí tính toán khi kết hợp các bước lùi để loại bỏ các đặc trưng không cần thiết. Trong bài báo [32], nhóm tác giả có đề xuất một thuật toán là chỉ lùi một bước khi bình phương sai số (squared error) giảm không quá một nửa mức giảm bình phương sai số trong các bước tiến trước đó. Thực tế, bài báo [24] đã chỉ ra một ví dụ mô phỏng để cho

thấy rằng việc lựa chọn đặc trưng từng bước hiếm khi thực hiện một bước lùi. Vì vậy, trong cách tiếp cận của chúng tôi, chúng tôi sẽ thêm các đặc trưng đến khi đạt được mô hình thỏa mãn, và sau đó sử dụng lựa chọn lùi để loại bỏ các đặc trưng dư thừa.

Thứ ba, với tập dữ liệu chỉ có một đặc trưng, thì $trace(S_w^{-1}S_b)$ có thể được rút gọn thành

$$t = \frac{\sum_{i=1}^{C} n_i(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})^2}{\sum_{i=1}^{C}\sum_{j=1}^{n_i}(\boldsymbol{x}_{ij} - \bar{\boldsymbol{x}}_i)^2}, \tag{3.2.1}$$

còn với tập dữ liệu có những đặc trưng đã được chuẩn hóa thì S_b có thể được rút gọn thành $S_b = \sum_{i=1}^{C} n_i \bar{x}_i \bar{x}_i'$. Đặt

$$t_R = trace(S_w^{-1}S_b) \tag{3.2.2}$$

với R là tập hợp các đặc trưng được chọn trong mô hình.

Thuật toán chúng tôi đề xuất là thuật toán PFST được trình bày trong thuật toán 4. Chúng tôi trước tiên chia các đặc trưng thành B khối để chạy song song B khối. Bắt đầu với tập hợp các đặc trưng được chọn R là tập rỗng. Do đó, chúng tôi tính $\arg\max_f\{t_f : f \in F_b\}, b = 1, ..., B$ song song trên B khối và sinh ra một tập hợp R đầu tiên.

Sau đó, chúng tôi sử dụng thuật toán 5 để lần lượt thêm vào các đặc trưng tốt với mô hình và loại bỏ các đặc trưng không quan trọng khỏi các lần lặp lại tiếp theo (ta gọi bước này là *forward-dropping*). Cụ thể hơn, thuật toán loại bỏ tất các đặc trưng không thể cái thiện được tiêu chí vết một lượng lớn hơn β trong một khối. Điều này giúp giảm được chi phí tính toán của việc quét lại các đặc trưng mà dường như chúng không thể cải thiện được mô hình khi so sánh với các đặc trưng khác.

Tuy nhiên, một nhược điểm của cách làm này là ta vô tình loại bỏ đi một đặc trưng tưởng chừng là không quan trọng. Nghĩa là, ở bước này nó không quan trọng, nhưng khi kết hợp được với một số đặc trưng khác thì nó sẽ giúp cải thiện tốt hơn cho mô hình. Vì thế, sau khi những bước này kết thúc, chúng tôi sử dụng thuật toán 6 để chạy lại từ đầu với tập hợp các đặc trưng là $P = A \setminus R$ và lại chia chúng vào B khối. Sau đó tiến hành lựa chọn lần nữa và lần này không loại bỏ những đặc trưng sớm (ta gọi bước này là *re-forward*). Vì hầu hết các đặc trưng quan trọng đã có trong mô hình, nên giai đoạn này cũng sẽ không tốn quá nhiều chi phí tính toán như việc chạy song song lựa chọn tiến. Thực nghiệm cũng cho thấy, thuật toán PFST mất ít thời gian hơn so với các kĩ thuật, thuật toán khác.

Cuối cùng, việc đưa vào các đặc trưng mới, có thể làm cho những đặc trưng được đưa vào trước đó trở nên dư thừa, lý do giải thích đơn giản nhất có thể là do tính tương quan cao của các đặc trưng. Do đó, giai đoạn cuối cùng của thuật toán PFST sẽ cố gắng loại bỏ các đặc trưng dư thừa nếu có. Trong giai đoạn này, chúng tôi tiến hành lựa chọn lùi để loại bỏ các đặc trưng dư thừa còn lại trong mô hình (ta gọi bước này là *backward*). Nhắc lại, điều này giúp tránh phát sinh chi phí tính toán cho việc kiểm tra lùi sau mỗi

lần đưa vào một đặc trưng mới. Điều này khác với các bước lùi trong lựa chọn từng bước.

Chú ý rằng một giá trị γ cao thì sẽ phải loại bỏ nhiều đặc trưng hơn và quét lại ít hơn trong quá trình forward-dropping. Tuy nhiên, phụ thuộc vào dữ liệu và tiêu chí chọn, ta có thể ưu tiên sử dụng một giá trị γ thấp hơn cho dữ liệu nhiều chiều. Lý do là một giá trị γ cao có khả năng loại bỏ quá nhiều đặc trưng, dẫn đến ít đặc trưng có cơ hội được thêm vào mô hình trong quá trình forward-dropping. Điều này làm cho quá trình forward-dropping sẽ kết thúc sớm, và khi đó sẽ còn một lượng lớn đặc trưng cần quét lại trong quá trình re-forward.

Một điều quan trọng cần chú ý là sau bước forward-dropping là re-foward. Do đó, sau các bước forward-dropping, thuật toán sẽ xếp hạng các đặc trưng theo mức độ quan trọng. Có thể chỉ định số lượng tối đa các đặc trưng được đưa vào mô hình nếu người dùng muốn bộ bộ đặc trưng có số lượng nhỏ hơn những gì thật sự thu được. Do đó, chúng tôi có thể chỉ định số lượng đặc trưng tối đa được đưa vào mô hình cho gia đoạn re-foward.

Điều cuối cùng nhưng cũng không kém phần quan trọng, nếu số lượng các đặc trưng được chọn sau giai đoạn re-forward là không đáng kể, người dùng nên sử dụng tính năng lùi an toàn thay vì lùi song song. Điều này là do khi số lượng đặc trưng trong mô hình ít, sẽ dẫn đến việc tính toán song song làm thuật toán chạy chậm hơn.

Thuật toán 4. Thuật toán tiến-lùi song song với loại bỏ sớm
(Parallel forward-backward algorithm with early dropping)

Input: Tập hợp A gồm tất cả các đặc trưng. Các feature được chia vào các khối $F_1, ..., F_B$, và nhãn Y tương ứng, tham số tiến là α, tham số lùi β, tham số loại bỏ sớm là γ, số bước tối đa sau khi bắt đầu lại là maxRef.
Output: Tập hợp R các đặc trưng được chọn.

Procedure:

$f_b \leftarrow \arg\max_f \{t_{\{f\}} : f \in F_b\}, b = 1, ..., B$ chạy song song trên B khối.
$R \leftarrow \{f_1, ..., f_B\}$.
 # Forward with forward-dropping stage:
while $\bigcup_{1 \le b \le B} F_b \neq \emptyset$ **do**
    $f_b, F_b \leftarrow OneForwardDropping(F_b, Y, \alpha, \gamma)$ (chạy song song trên B khối).
    $R \leftarrow R \cup \{f_1, ..., f_B\}$
end while
 # Re-forward stage:
Bắt đầu lại với tập hợp $A \setminus R$ và lại chia chúng vào B khối $F_1, ..., F_B$.
runs $\leftarrow 0$

while runs $<$ maxRef & $\bigcup_{1\leq b\leq B} F_b \neq \emptyset$ **do**
    $f_b, F_b \leftarrow$ OneReforward(F_b, Y, α) (chạy song song trên B khối)
    $R \leftarrow R \cup \{f_1, ..., f_B\}$
    runs $\leftarrow$ runs $+1$
end while
 # Backward stage:
Chia R thành B khối.
$f_b \leftarrow \arg\max_{f\in F_b} t_{R\setminus\{f\}}, b = 1, ...B$ chạy song song trên B khối
$f_r \leftarrow \arg\max_{f\in\{f_1,...,f_B\}} t_{R\setminus\{f\}}$
if $t_R - t_{R\setminus\{f_r\}} < \beta$ **then**
    $R \leftarrow R \setminus \{f_r\}$
else
    break
end if
return R

Thuật toán 5. OneForwardDropping

Input: Một khối các đặc trưng F_b và nhãn Y tương ứng, tham số loại bỏ sớm là γ.
Output: Một đặc trưng tốt f_b được chọn, một khối các đặc trưng còn lại F_b.

$f_b \leftarrow \arg\max_{f\in F_b} t_{R,f}$
if $t_{R,f_b} - t_R < \alpha$ **then**
    # if there is no more relevant features, drop block by setting it to $\emptyset$
    $F_b \leftarrow \emptyset$
else
    # Early dropping:
    $D \leftarrow \{f : t_{\{R,f\}} - t_R < \gamma\}$
    $F_b \leftarrow F_b \setminus \{D, f_b\}$
end if
return f_b, F_b

Thuật toán 6. OneReforward

Input: Một khối các đặc trưng F_b và nhãn Y tương ứng, tham số tiến là α. **Output:** Một đặc trưng tốt f_b được chọn, một khối các đặc trưng còn lại F_b.

$f_b \leftarrow \arg\max_{f\in F_b} t_{R,f}$
if $t_{R,f_b} - t_R < \alpha$ **then**
    # if there is no more relevant features, drop block by setting it to $\emptyset$:
    $F_b \leftarrow \emptyset$
else
    $F_b \leftarrow F_b \setminus \{f_b\}$

7: **end if**
8: **return** f_b, F_b

Chương 4

Thực nghiệm

Bảng 4.1: Các tập dữ liệu được sử dụng trong thực nghiệm

Tập dữ liệu	# Số lớp	# Số đặc trưng	# Kích thước mẫu
Breast cancer	2	30	569
Parkinson	2	754	756
Mutants	2	5408	31419
Gene	5	20531	801
Micromass	10	1087	360

Để mô tả hiệu suất của PFST, chúng tôi so sánh PFST với một số kĩ thuật gồm

- Parallel Sequential Forward Selection (PSFS) [26],
- Parallel Sequential Backward Selection (PSBS) [26],
- Parallel Support Vector Machine Feature Selection based on Recursive Feature Elimination with Cross-Validation (PSVMR) [14],
- Parallel Mutual Information-based Feature Selection (PMI) [3].

4.1 Thông tin về các tập dữ liệu và các thiết lập

Các thực nghiệm đã được hoàn thành trên các tập dữ liệu lấy từ thư viện Scikit-learn [26] và kho dữ liệu máy học UCI [8]. Thông tin số liệu của các tập dữ liệu này được trình bày trong bảng 4.1.

Chúng tôi đã cộng một lượng nhỏ nhiễu vào tập dữ liệu Gene và tập dữ liệu Mutants để tránh lỗi không tìm được nghịch đảo của các ma trận đơn khi chạy thuật toán PSFT. Ngoài ra, đối với tập dữ liệu Mutants, chúng tôi đã loại bỏ một dòng mà tất cả các giá trị trong dòng đều là rỗng.

Với thuật toán PSFT, các tham số được sử dụng là $\alpha = \gamma = 0.05$, $\beta = 0.01$. Với thuật toán PSBS và PSFS, chúng tôi đã sử dụng thuật toán KNN với $K = 3$ làm hàm ước lượng. Với thuật toán PSVMR, chúng tôi sử dụng kernel tuyến tính và sử dụng kernel "JMI" cho thuật toán PMI. Để cho việc so sánh được công bằng, chúng tôi đã set số lượng đặc trưng cần được chọn từ các kĩ thuật khác bằng với số lượng đặc trưng mà PSFT chọn được.

Về cấu hình, chúng tôi chạy thực nghiệm trên một CPU là AMD Ryzen 7 3700X với 8 nhân và 16 luồng, 3.6GHz và 16GB ram. Sau khi chọn được các đặc trưng, chúng tôi thực hiện bài toán phân loại bằng việc sử dụng mô hình phân tích biệt thức tuyến tính (linear discriminant analysis - LDA) và trình bày kết quả của 5-fold misclassification rate trong bảng 4.2. Ngoài ra, chúng tôi cũng trình bày thời gian chạy và số lượng đặc trưng được chọn từ tất cả đặc trưng ban đầu trong bảng 4.3

Chúng tôi sẽ bỏ những trường hợp nếu không nhận được kết quả sau 5 giờ hoặc khi có vấn đề về việc tràn ram, và ký hiueej NA trong bảng 4.2 và 4.3) để chỉ những trường hợp như vậy.

4.2 Kết quả và thảo luận

Bảng 4.2: 5-fold misclassification rate

Datasets	# Selected Features	PFST (our)	PSFS	PSBS	PSVMR	PMI	Full Features
Breast cancer	3	**0.042**	0.111	0.074	0.051	0.076	0.042
Parkinson	11	**0.112**	0.234	NA	NA	0.181	0.362
Mutants	6	**0.008**	NA	NA	NA	NA	0.010
Gene	12	**0.006**	0.009	NA	NA	0.007	0.042
Micromass	19	0.115	0.310	0.218	**0.096**	0.228	0.129

Từ bảng 4.2 và 4.3, ta có thể thấy rằng phương pháp lựa chọn đặc trưng PFST không chỉ có hiệu suất tốt về mặt tốc độ mà con đạt độ chính xác cao cho bài toán phân loại, Ví dụ, với tập dữ liệu "Breast cancer", PFST đã chọn ra tập hợp gồm 3 đặc trưng từ 30 đặc trưng ban đầu với chỉ 0.095 giây, trong khi PSBS cần 8.268 giây và PSVMR cần 12.470 giây. Không chỉ có thời gian chạy tốt nhất, phương pháp PFST còn đạt được kết quả phân loại tốt nhất với tỉ lệ phân loại sai chỉ 0.042, đây cũng là tỉ lệ phân loại sai khi sử dụng tất cả đặc trưng. Rõ ràng, kết quả này đã cho thấy, PFST lựa chọn được các đặc trưng tốt, ảnh hưởng thật sự đến kết quả phân loại. Ngoài ra, ta cũng thấy rằng hiệu

Bảng 4.3: Running time and number of selected features

Datasets	# selected features	# Features	Running Time (s)				
			PFST (our)	PSFS	PSBS	PSVMR	PMI
Breast cancer	3	30	**0.095**	1.403	8.268	12.470	0.646
Parkinson	11	754	**3.219**	163.32	NA	NA	77.269
Mutants	6	5408	**674.702**	NA	NA	NA	NA
Gene	12	20531	**172.386**	5350.14	NA	NA	2706.45
Micromass	19	1087	**16.1**	252.9	10561.3	46.909	144.684

suất của PFST tốt hơn PSBS - một phiên bản lựa chọn đặc trưng lùi.

Với tập dữ liệu Parkinson, PFST đã lựa chọn ra 11 trong tổng số 754 đặc trưng. Kết quả thu được là tỉ lệ phân loại sai vẫn là thấp nhất với chỉ 0.112, khoảng 33% tỉ lệ phân loại sai khi sử dụng toàn bộ đặc trưng (0.362), và 66.87% tỉ lệ phân loại sai của phương pháp tốt thứ hai là PMI (0.181). Điều này cho thấy PFST đã loại bỏ rất hiệu quả các đặc trưng dư thừa và đẩy được hiệu suất lên đáng kể. Về mặt thời gian chạy, PFST chỉ mất 3.219 giây để chọn 11 đặc trưng từ 754 đặc trưng, trong khi PMI cần đến 77.269 giây, PSFS cần 163.32 giây. PSBS và PSVMR không thể thu được kết quả khi thời gian chạy vượt quá 5 giờ.

Trong tập dữ liệu nhiều đặc trưng nhất, tập dữ liệu Gene, PFST đã rút gọn 20531 đặc trưng xuống còn 12 đặc trưng trong chưa đầy 3 phút và đạt được tỉ lệ phân loại sai tốt nhất với chỉ 0.006. Mặc dùng PSFS và PMI cũng đạt được kết quả phân loại tốt với tỉ lệ phân loại sai lần lượt là 0.009 và 0.007, nhưng hai phương pháp này chạy lâu hơn PFST, còn đối với PSBS và PSVMR đã không thể thu được kết quả trong vòng 5 giờ chạy.

Mặc dù PFST là thuật toán nhanh nhất trong số các thuật toán được sử dụng trong phần thực nghiệm này, nhưng phương pháp này cũng làm tốt hơn các phương pháp khác về tỉ lệ phân loại sai với bốn trong năm tập dữ liệu. Với tập dữ liệu Micromass, hiệu suất tốt nhất về kết quả phân loại thuộc về phương pháp PSVMR, và tốt thứ hai là PFST. Tuy nhiên, PSVMR chỉ tốt hơn PFST rất ít, không đáng kể (chỉ thấp hơn 1.9% tỉ lệ phân loại sai), nhưng thời gian chạy lại gần gấp 3 lần PSFT (46.909 giây so với 16.1 giây).

Kết luận

Qua khóa luận này, chúng tôi đã trình bày phương pháp PFST, một cách tiếp cận mới về việc lựa chọn song song các đặc trưng từ các tập dữ liệu lớn trong bài toán phân loại. Phương pháp của chúng tôi đã sử dụng tiêu chí vết, là tiêu chí có các thuộc tính để đánh giá các đặc trưng tốt. Chúng tôi ước lượng các tiếp cận này thông qua nhiều thực nghiệm và các tập dữ liệu khác nhau. Chúng tôi sử dụng LDA làm mô hình phân loại trên các đặc trưng được chọn. Thực nghiệm cho thấy phương pháp của chúng tôi có thể chọn ra các đặc trưng tốt với thời gian ngắn hơn khi so với các phương pháp tiếp cận khác. Hơn nữa, phân loại dựa trên các đặc trưng chọn được bằng PFST cũng thu được độ chính xác tốt hơn các phương pháp khác (tốt hơn bốn trong năm tập dữ liệu) và tốt hơn khi chạy mô hình phân loại với toàn bộ đặc trưng. Tuy nhiên, một trong những nhược điểm của PFST là tiêu chí vết chỉ có thể sử dụng cho các đặc trưng liên tục. Do đó, trong tương lại, chúng tôi vẫn sẽ nỗ lực làm việc, và hi vọng sẽ khám phá ra cách mở rộng cho các trường hợp còn lại trong bài toán phân loại.

Ngoài ra, chúng tôi cũng đã xây dựng thuật toán như một thư viện trong python. Người dùng có thể tìm thấy và cài đặt trong github sau: https://github.com/pthnhan/PFST

Tài liệu tham khảo

[1] Jun Chin Ang, Andri Mirzal, Habibollah Haron, and Haza Nuzly Abdull Hamed. Supervised, unsupervised, and semi-supervised feature selection: a review on gene selection. *IEEE/ACM transactions on computational biology and bioinformatics*, 13(5):971–989, 2015.

[2] Hiromasa Arai, Crystal Maung, Ke Xu, and Haim Schweitzer. Unsupervised feature selection by heuristic search with provable bounds on suboptimality. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[3] Mohamed Bennasar, Yulia Hicks, and Rossitza Setchi. Feature selection using joint mutual information maximisation. *Expert Systems with Applications*, 42(22):8520–8532, 2015.

[4] Verónica Bolón-Canedo, Noelia Sánchez-Maroño, and Amparo Alonso-Betanzos. *Feature selection for high-dimensional data*. Springer, 2015.

[5] Giorgos Borboudakis and Ioannis Tsamardinos. Forward-backward selection with early dropping. *The Journal of Machine Learning Research*, 20(1):276–314, 2019.

[6] Jerffeson Teixeira de Souza, Stan Matwin, and Nathalie Japkowicz. Parallelizing feature selection. *Algorithmica*, 45(3):433–456, 2006.

[7] David L Donoho, Michael Elad, and Vladimir N Temlyakov. Stable recovery of sparse overcomplete representations in the presence of noise. *IEEE Transactions on information theory*, 52(1):6–18, 2005.

[8] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[9] Keinosuke Fukunaga. *Introduction to statistical pattern recognition*. Elsevier, 2013.

[10] Daniel J Garcia, Lawrence O Hall, Dmitry B Goldgof, and Kurt Kramer. A parallel feature selection algorithm from random subsets. In *Proceedings of the international workshop on parallel data mining*, volume 18, pages 64–75. Citeseer, 2006.

[11] David E Goldberg and John Henry Holland. Genetic algorithms and machine learning. 1988.

[12] Alberto Guillén, Antti Sorjamaa, Yoan Miche, Amaury Lendasse, and Ignacio Rojas. Efficient parallel feature selection for steganography problems. In *International Work-Conference on Artificial Neural Networks*, pages 1224–1231. Springer, 2009.

[13] Isabelle Guyon, Jiwen Li, Theodor Mader, Patrick A Pletscher, Georg Schneider, and Markus Uhr. Competitive baseline methods set new standards for the nips 2003 feature selection benchmark. *Pattern recognition letters*, 28(12):1438–1444, 2007.

[14] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1):389–422, 2002.

[15] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[16] Richard Arnold Johnson, Dean W Wichern, et al. *Applied multivariate statistical analysis*, volume 5. Prentice hall Upper Saddle River, NJ, 2002.

[17] Deguang Kong and Chris Ding. Pairwise-covariance linear discriminant analysis. In *Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

[18] Vipin Kumar and Sonajharia Minz. Feature selection: a literature review. *SmartCR*, 4(3):211–229, 2014.

[19] Jundong Li, Kewei Cheng, Suhang Wang, Fred Morstatter, Robert P Trevino, Jiliang Tang, and Huan Liu. Feature selection: A data perspective. *ACM Computing Surveys (CSUR)*, 50(6):1–45, 2017.

[20] Huan Liu and Hiroshi Motoda. *Feature selection for knowledge discovery and data mining*, volume 454. Springer Science & Business Media, 2012.

[21] Félix Garcıa López, Miguel Garcıa Torres, Belén Melián Batista, José A Moreno Pérez, and J Marcos Moreno-Vega. Solving feature subset selection problem by a parallel scatter search. *European Journal of Operational Research*, 169(2):477–489, 2006.

[22] Mahdokht Masaeli, Yan Yan, Ying Cui, Glenn Fung, and Jennifer G Dy. Convex principal feature selection. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 619–628. SIAM, 2010.

[23] Nouredine Melab, Sébastien Cahon, and El-Ghazali Talbi. Grid computing for parallel bioinspired algorithms. *Journal of parallel and Distributed Computing*, 66(8):1052–1061, 2006.

[24] Thu Nguyen. Faster feature selection with a dropping forward-backward algorithm. *arXiv preprint arXiv:1910.08007*, 2019.

[25] Jianmo Ni, Jiacheng Li, and Julian McAuley. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197, 2019.

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[27] Yvan Saeys, Iñaki Inza, and Pedro Larrañaga. A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19):2507–2517, 2007.

[28] Kristina P Sinaga, Ishtiaq Hussain, and Miin-Shen Yang. Entropy k-means clustering with feature reduction under unknown number of clusters. *IEEE Access*, 9:67736–67751, 2021.

[29] Sameer Singh, Jeremy Kubica, Scott Larsen, and Daria Sorokina. Parallel large scale feature selection for logistic regression. In *Proceedings of the 2009 SIAM international conference on data mining*, pages 1172–1183. SIAM, 2009.

[30] Joel A Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Transactions on Information theory*, 50(10):2231–2242, 2004.

[31] Ioannis Tsamardinos, Giorgos Borboudakis, Pavlos Katsogridakis, Polyvios Pratikakis, and Vassilis Christophides. A greedy feature selection algorithm for big data of high dimensionality. *Machine learning*, 108(2):149–202, 2019.

[32] Tong Zhang. Adaptive forward-backward greedy algorithm for learning sparse representations. *IEEE transactions on information theory*, 57(7):4689–4708, 2011.

[33] Zheng Zhao, Ruiwen Zhang, James Cox, David Duling, and Warren Sarle. Massively parallel feature selection: an approach based on variance preservation. *Machine learning*, 92(1):195–220, 2013.