TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

ĐỒ ÁN

TỐT NGHIỆP ĐẠI HỌC

NGÀNH CÔNG NGHỆ THÔNG TIN

TÊN ĐỀ TÀI

Tích hợp dữ liệu việc làm

Sinh viên thực hiện: Đoàn Thị Thúy Nga

Lớp CNTT 2.03 K59

Giáo viên hướng dẫn: TS. Đỗ Bá Lâm

HÀ NỘI 6-2019

PHIẾU GIAO NHIỆM VỤ ĐỒ ÁN TỐT NGHIỆP

1. Thông tin sinh viên:

Họ tên sinh viên: Đoàn Thị Thúy Nga

Điện thoại liên lạc: 0868109863

Email: [doannga1234@gmail.com](mailto:doannga1234@gmail.com)

Lớp: CNTT 2.03 K59

Hệ đào tạo: Kỹ sư hệ thống thông tin

Đồ án tốt nghiệp được hiện tại: Trường Đại học Bách Khoa Hà Nội

Thời gian làm đồ án: Từ tháng 1/2019 đến tháng 6/2019

1. Mục đích nội dung của ĐATN:

Xây dựng một hệ thống tích hợp dữ liệu việc làm

1. Các nhiệm vụ của ĐATN:

* Tìm hiểu về bài toán tích hợp dữ liệu
* Xây dựng công cụ thu thập tự động dữ liệu, sử dụng các thuật toán học máy để giải quyết bài toán
* Xử lý bài toán liên quan đến dữ liệu
* Đánh giá kết quả hệ thống
* Xây dựng hệ thống thông tin cơ bản để trực quan hóa dữ liệu thu thập được

1. Lời cam đoan của sinh viên:

Tôi – Đoàn Thị Thúy Nga – cam kết ĐATN là công trình nghiên cứu của bản thân tôi dưới sự hướng dẫn của TS. Đỗ Bá Lâm.

Các kết quả nêu trong ĐATN là trung thực, không phải là sao chép toàn văn của bất kỳ công trình nào khác.

Hà Nội, ngày tháng năm 2019

Tác giả ĐATN

Đoàn Thị Thúy Nga

1. Xác nhận của giáo viên hướng dẫn về mức độ hoàn thành của ĐATN và cho phép bảo vệ:

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

Hà Nội, ngày tháng năm 2019

Giáo viên hướng dẫn

TS. Đỗ Bá Lâm

LỜI CẢM ƠN

Năm năm trôi qua như cơn gió thoảng qua, cảm ơn Bách Khoa cho em những giây phút đáng nhớ của tuổi trẻ, cho em những cơ hội để trưởng thành, chính chắn hơn. Là nơi lưu giữ thanh xuân tươi đẹp của tuổi trẻ.

Em xin chân thành cảm ơn thầy cô, bạn bè, cô bác bảo vệ ở Bách Khoa, đặc biệt là những người chèo lái Viện Công nghệ thông tin và truyền thông, cảm ơn vì chèo lái chúng em đến bến bờ tương lai, cho chúng em những cơ hội và thách thức tuổi trẻ.

Em xin gửi lời cảm ơn chân thành và sâu sắc tới TS. Đỗ Bá Lâm – người thầy đã hướng dẫn, hỗ trợ em rất nhiều trong quá trình thực hiện đồ án tốt nghiệp.

Xin cảm ơn!

TÓM TẮT NỘI DUNG ĐỒ ÁN TỐT NGHIỆP

Danh mục hình vẽ

Danh mục thuật toán

Danh mục công thức

Danh mục hình vẽ và thuật ngữ

Danh mục bảng

Chương 1. Giới thiệu đề tài

* 1. Đặt vấn đề
  2. Mục tiêu và phạm vi đề tài
  3. Định hướng giải pháp
  4. Bố cục đồ án

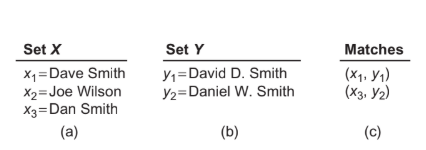
Chương 2. Tình hình nghiên cứu hiện tại

Chương 3. Cơ sở lý thuyết

3.1 Đối sánh lược đồ, đối sánh chuỗi

Đối sánh chuỗi là bài toán tìm các chuỗi trỏ tới cùng một đối tượng trong thế giới thực. Ví dụ, chuỗi David Smith trong một cơ sở dữ liệu có thể chỉ tới cùng một người David R.Smith trong cơ sở dữ liệu khác. Hay chuỗi 1210 W. Dayton St, Madison WI và 1210 West Dayton, Madison WI 53706 cùng chỉ tới một địa chỉ vật lý. Như vậy, đối sánh chuỗi đóng vai trò quan trọng trong bài toán tích hợp dữ liệu bao gồm đối sánh lược đồ, đối sánh dữ liệu và trích rút thông tin.

Mô tả bài toán: Cho hai tập chuỗi X và Y; tìm tất cả các cặp chuỗi (x, y), x∈X và y∈Y, sao cho x và y trỏ tới cùng thực thể thế giới thực. Gọi các cặp như vậy là các sánh đôi (match)



Hình x. Ví dụ về đối sánh tên người. Cột c là kết quả đối sánh giữa cơ sở dữ liệu a và b

Hai thách thức lớn nhất của bài toán được đặt ra là độ chính xác và khả năng mở rộng. Việc so sánh các chuỗi với độ chính xác cao thường rất khó vì các chuỗi liên quan đến cùng một thực thể trong thế giới thực thường rất khác nhau. Nguyên nhân có thể do lỗi đánh máy, sự khác nhau về quy ước định dạng, sử dụng tên viết tắt, hay tên gọi, biệt danh khác, ngoài ra, nguồn dữ liệu có thể không đủ thông tin để xác định xem các chuỗi có cùng trỏ đến một thực thế hay không?

Để giải quyết thách thức về độ chính xác, người ta sử dụng độ đo tương đồng s ánh xạ cặp (x,y) ∈ [0, 1], s càng lớn thì x, y càng tương đồng.

Các phương pháp tính độ tương đồng

3.2 Đối sánh lược đồ

3.3 Trích rút thuộc tính

Xây dựng Wrapper thủ công, với đầu vào là một tập các website, lập trình viên có thể tạo thủ công lược đồ đích , chương trình trích xuất . Có rất nhiều cách mô hình hóa trang web, nếu coi trang web là một cây DOM, ta có thể sử dụng đường dẫn Xpath để lấy các thuộc tính trong tập . Nếu coi trang web như là một xâu, thì wrapper có thể viết thông qua ngôn ngữ Perl. Nếu coi như một website trực quan, bao gồm khối header, khối nội dung, khối footer, ta có thể xây dựng wrapper thông qua ngôn ngữ bậc cao, chẳng hạn như HRLT.

Xây dựng một Wrapper có thể học tự động từ dữ liệu huấn luyện, dữ liệu huấn luyện có thể được gán/ tạo bởi người sử dụng

Học Wrapper không cần Schema, còn gọi là cách tiếp cận tự động. Đầu vào là tập các webpages từ nguồn Đánh giá độ tương tự và khác nhau giữa các trang, sau đó, suy diễn tự động lược đồ và chương trình trích xuất . Phương pháp tiếp cận điển hình là RoadRunner

Xây dựng Wrapper tương tác có sử dụng rất ít hoặc không cần sử dụng đầu vào, tìm kiếm cho tới khi phát hiện các vấn đề chưa chắc chắn. Hỏi người dùng và yêu cầu phản hồi, sau đó tiếp tục tìm kiếm. Lặp lại quá trình đó cho đến khi tạo ra một Wrapper mà người dung mong muốn. Phản hồi người dùng có thể tồn tại ở nhiều dạng: Gán nhãn các trang, xác định kết quả trích xuất là chính xác, tạo các luật trích xuất, trả lời câu hỏi của hệ thống, xác định các mẫu,… Thách thức liên quan đến yêu cầu và phản hồi của người dung: Khi nào cần hỏi? Hỏi cái gì? Có 3 hệ thống điển hình giải quyết bài toán này:

+ Gán nhãn trang với Stalker

+ Xác định kết quả trích xuất với Poly

+ Tạo luật trích xuất với Lixto

## 3.4 Phương pháp học máy cho bài toán phân loại

### 3.3.1 Phương pháp k-NN

K-nearest neighbor là một trong những thuật toán supervised-learning đơn giản nhất, hiệu quả với một số trường hợp trong Học máy. Khi training, thuật toán này không học một điều gì từ dữ liệu training, đây là lý do thuật toán này được xếp vào loại lazy learning, mọi tính toán được thực hiện chỉ khi nó cần dự đoán kết quả của dữ liệu mới. K-NN có thể áp dụng được vào cả hai loại của bài toán Supervised-learning là Classification và Regression. K-NN còn được gọi là một thuật toán Instance-based hay Memory-based learning.

Với k-NN, trong bài toán Classification, nhãn lớp của một điểm dữ liệu mới được suy ra trực tiếp từ k điểm dữ liệu gần nhất trong tập training. Nhãn lớp của một dữ liệu test có thể được quyết định bằng việc bầu chọn theo số phiếu giữa các điểm gần nhất, hoặc nó có thể được suy ra bằng cách đánh trọng số khác nhau cho mỗi điểm trong các điểm gần nhất đó rồi suy ra nhãn lớp.

Tóm lại, k-NN là thuật toán đi tìm đầu ra của một điểm dữ liệu mới chỉ dựa trên thông tin của k điểm dữ liệu trong tập training gần nó nhất (k-lân cận), không quan tâm đến việc có một vài điểm dữ liệu trong những điểm gần nhất ấy là nhiễu. Dưới đây là một ví dụ k-NN trong Classification với k=1.

Giải thuật k-NN cho phân lớp: Mỗi ví dụ học x được biểu diễn bởi 2 thành phần: Mô tả: ,…, ) với và nhãn lớp: c ∈ C, với C là tập các nhãn lớp được xác định trước. Giai đoạn học sẽ lưu lại các ví dụ học trong tập học D. Giai đoạn phân lớp để phân lớp cho một ví dụ mới z. Với mỗi ví dụ học x ∈ D, tính khoảng cách giữa x và z, ta xác định tập NB(z) – các láng giềng gần nhất của z, gồm k ví dụ học trong D gần nhất với z tính theo một hàm khoảng cách d, sau đó phân z vào lớp chiếm số đông (the majority class) trong số các lớp của các ví dụ trong NB(z)

Ưu điểm của phương pháp k-NN: Chi phí thấp cho quá trình huấn luyện. Dự đoán kết quả của dữ liệu mới rất đơn giản. Hoạt động tốt với các bài toán phân loại gồm nhiều lớp, không cần phải học c bộ phân loại cho c lớp. Về mặt lý thuyết, có thể đạt khả năng phán đoán tối ưu khi gặp một số điều kiện. Rất linh động trong việc chọn hàm khoảng cách.

Nhược điểm của phương pháp k-NN: Phải lựa chọn hàm tính khoảng cách thích hợp với bài toán. Nhạy cảm với nhiễu khi k nhỏ. Chi phí tính toán cao tại thời điểm phân loại, k càng lớn thì độ phức tạp càng tăng.

### 3.3.2 Phương pháp SVM

Máy vectơ hỗ trợ (Support vector machine – SVM) được đề cử bởi V.Vapnik và các đồng nghiệp của ông vào những năm 1970 ở Nga và sau đó trở nên nổi tiếng và phổ biến vào những năm 1990.

SVM là một phương pháp phân lớp tuyến tính (linear classifier), với mục đích xác định một siêu phẳng (hyperplane) để phân tách hai lớp của dữ liệu. Ví dụ: một lớp có nhãn dương, một lớp có nhãn âm. Một cách trực giác, để phân loại tốt nhất thì các siêu phẳng nằm ở càng xa các điểm dữ liệu của tất cả các lớp (gọi là hàm lề càng tốt) vì nói chung, lề càng lớn thì sai số tổng quát hóa của thuật toán phân loại càng bé.

Trong nhiều trường hợp, không thể phân chia các lớp dữ liệu một cách tuyến tính trong không gian ban đầu được dùng để mô tả vấn đề. Vì vậy, nhiều khi cần phải ánh xạ các điểm dữ liệu trong không gian ban đầu vào một không gian mới nhiều chiều hơn, giúp việc phân tách chúng trở nên dễ dàng hơn trong không gian mới. Để việc tính toán được hiệu quả, ánh xạ sử dụng trong thuật toán SVM chỉ đòi hỏi tích vô hướng của các vectơ dữ liệu trong không gian mới có thể được tính dễ dàng từ các tọa độ trong không gian cũ. Tích vô hướng này được xác định bằng một hàm hạt nhân K(x,y) phù hợp. Một siêu phẳng trong không gian mới được định nghĩa là tập các điểm có tích vô hướng với một vectơ cố định trong không gian đó là một hằng số. Vectơ xác định một siêu phẳng sử dụng trong SVM là một tổ hợp tuyến tính của các vectơ dữ liệu luyện tập trong không gian mới với các hệ số αi. Với siêu phẳng lựa chọn như trên, các điểm x trong không gian đặc trưng được ánh xạ vào một siêu phẳng là các điểm thỏa mãn:

= hằng số

Ghi chú rằng nếu K(x,y) nhận giá trị ngày càng nhỏ khi y xa dần khỏi x thì mỗi số hạng của tổng trên được dùng để đo độ tương tự giữa x với điểm xi tương ứng trong dữ liệu luyện tập. Như vậy, tác dụng của tổng trên chính là so sánh khoảng cách giữa điểm cần dự đoán với các điểm dữ liệu đã biết. Lưu ý là tập hợp các điểm x được ánh xạ vào một siêu phẳng có thể có độ phức tạp tùy ý trong không gian ban đầu, nên có thể phân tách các tập hợp thậm chí không lồi trong không gian ban đầu. SVM có một nền tảng lý thuyết chặt chẽ. Đây là một phương pháp phù hợp với những bài toán phân lớp không gian rất nhiều chiều (các đối tượng cần phân lớp được biểu diễn bởi một tập rất lớn các thuộc tính). SVM được biết đến là một trong số các phương pháp phân lớp tốt nhất đối với các bài toán phân lớp văn bản.

Biểu diễn tập r các quan sát:

SVM xác định một hàm phân tách tuyến tính:

Trong đó, w là vectơ trọng số các thuộc tính; b là một giá trị số thực

Sao cho với mỗi :

Ưu điểm của phương pháp SVM: Xử lý trên không gian số chiều cao: SVM là một công cụ tính toán hiệu quả trong không gian chiều cao, trong đó đặc biệt áp dụng cho các bài toán phân loại văn bản và phân tích quan điểm khi chiều có thể cực kỳ lớn. Tiết kiệm bộ nhớ: Do chỉ có một tập hợp con của các điểm được sử dụng trong quá trình huấn luyện và ra quyết định thực tế cho các điểm dữ liệu mới nên chỉ có những điểm cần thiết mới được lưu trữ trong bộ nhớ khi ra quyết định. Tính linh hoạt. Phân lớp thường là phi tuyến tính. Khả năng áp dụng Kernel mới cho phép linh động giữa các phương pháp tuyến tính và phi tuyến tính, từ đó khiến cho hiệu suất phân loại lớn hơn.

Nhược điểm của phương pháp SVM: Chỉ làm việc với không gian đầu vào là các số thực, chỉ phân 2 lớp. Siêu phẳng phân tách xác định bởi SVM thường khó hiểu đối với người dùng. Bài toán số chiều cao: Trong trường hợp số lượng thuộc tính (p) của tập dữ liệu lớn hơn rất nhiều so với số lượng dữ liệu (n) thì SVM cho kết quả khá tồi. Chưa thể hiện rõ tính xác suất: Việc phân lớp của SVM chỉ là việc cố gắng tách các đối tượng vào hai lớp được phân tách bởi siêu phẳng SVM. Điều này chưa giải thích được xác suất xuất hiện của một thành viên trong một nhóm là như thế nào. Tuy nhiên hiệu quả của việc phân lớp có thể được xác định dựa vào khái niệm margin từ điểm dữ liệu mới đến siêu phẳng phân lớp mà chúng ta đã bàn luận ở trên.

### 3.3.3 Thuật toán Random Forest

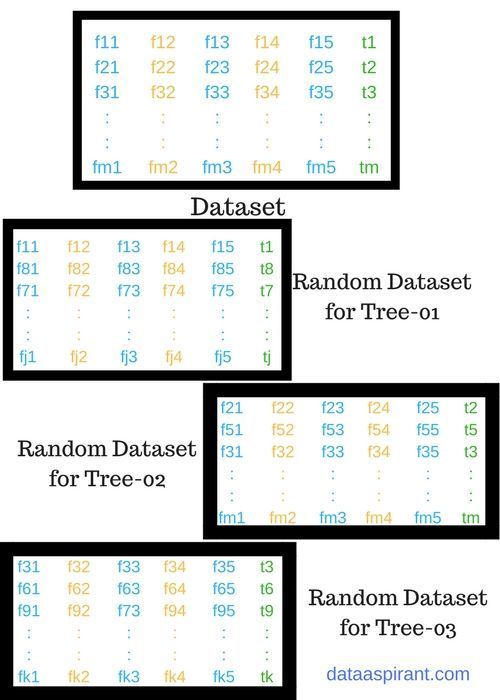
Thuật toán Random Forest là một thuật toán phân loại có giám sát, ta tạo ra một rừng bằng một cách nào đó và sử dụng nó một cách ngẫu nhiên. Có một mối quan hệ trực tiếp giữa số lượng cây trong rừng và kết quả đạt được: số lượng cây càng lớn thì kết quả càng chính xác hơn. Nhưng lưu ý là việc tạo ra rừng không giống như việc xây dựng quyết định với việc đạt được thông tin hoặc tiếp cận chỉ số. Cây quyết định là một công cụ hỗ trợ quyết định. Nó sử dụng một biểu đồ giống cây để hiển thị các hậu quả có thể xảy ra. Nếu bạn nhập một tập dữ liệu huấn luyện với các mục tiêu và các tính năng vào cây quyết định, nó sẽ xây dựng một số quy tắc. Các quy tắc này có thể được sử dụng để thực hiện các dự đoán. Ta sửdụng một ví dụ để minh họa điểm này: giả sử bạn muốn dự đoán liệu con gái bạn có thích phim hoạt hình hay không, bạn nên thu thập các bộ phim hoạt hình mà con mình thích và lấy một số tính năng làm đầu vào. Sau đó, thông qua thuật toán cây quyết định, bạn có thể tạo ra các quy tắc. Sau đó, bạn có thể nhập các tính năng của phim này và xem liệu phim đó có được con gái bạn thích hay không. Sự khác biệt giữa thuật toán Random Forest và thuật toán cây quyết định là trong Random Forest, quá trình es tìm nút gốc và tách các nút tính năng sẽ chạy ngẫu nhiên.

Chúng ta có một ví dụ thực tế để làm cho thuật toán Random Forest dễ hiểu. Giả sử Mady muốn đi đến những nơi khác nhau mà anh ấy có thể thích cho kỳ nghỉ hai tuần của mình, và anh ấy hỏi bạn mình để được tư vấn. Bạn anh ta sẽ hỏi anh ta đã ở đâu, và liệu anh ấy có thích những nơi anh ấy đến thăm không. Dựa trên câu trả lời của Mady, bạn của anh ấy bắt đầu đưa ra lời khuyên. Ở đây, bạn của anh ta tạo thành cây quyết định. Mady muốn hỏi thêm bạn bè để được tư vấn bởi vì anh ta nghĩ chỉ có một người bạn không thể giúp anh ta đưa ra quyết định chính xác. Vì vậy, những người bạn khác của anh ấy cũng hỏi anh ta những câu hỏi ngẫu nhiên, và cuối cùng, đưa ra câu trả lời. Anh xem xét nơi có nhiều phiếu nhất là quyết định kỳ nghỉ của mình. Một người bạn của anh ta hỏi anh ta một số câu hỏi và đưa ra đề nghị về nơi tốt nhất dựa trên câu trả lời. Đây là cách tiếp cận thuật toán cây quyết định điển hình. Người bạn đã tạo các quy tắc dựa trên các câu trả lời và sử dụng các quy tắc để tìm câu trả lời phù hợp với các quy tắc. Bạn bè của Mady cũng hỏi anh ta một cách ngẫu nhiên các câu hỏi khác nhau và đưa ra câu trả lời, mà cho Mady là phiếu bầu cho địa điểm này. Cuối cùng, nơi có số phiếu bầu cao nhất là nơi mà Mady sẽ chọn để đi. Đây là cách tiếp cận thuật toán Random Forest điển hình.

Có hai giai đoạn trong thuật toán Random Forest, một là tạo rừng ngẫu nhiên, cái còn lại là tạo dự đoán từ trình phân loại rừng ngẫu nhiên được tạo ra trong giai đoạn đầu tiên. Toàn bộ quá trình được hiển thị dưới đây, và nó dễ hiểu bằng cách sử dụng hình.

Ở đây, ta có mã giả tạo Random Forest: Chọn ngẫu nhiên các tính năng “K” từ tổng số tính năng “m” trong đó k << m. Trong số các tính năng “K”, tính toán nút “d” bằng cách sử dụng điểm chia tách tốt nhất. Chia nút thành các nút con gái bằng cách chia tách tốt nhất. Lặp lại các bước từ 1 đến c cho đến khi đạt được số lượng nút “l”. Xây dựng rừng bằng cách lặp lại các bước a đến d cho số lần “n” để tạo số cây “n”.

Hình 2. cho thấy quá trình lựa chọn các tính năng ngẫu nhiên:



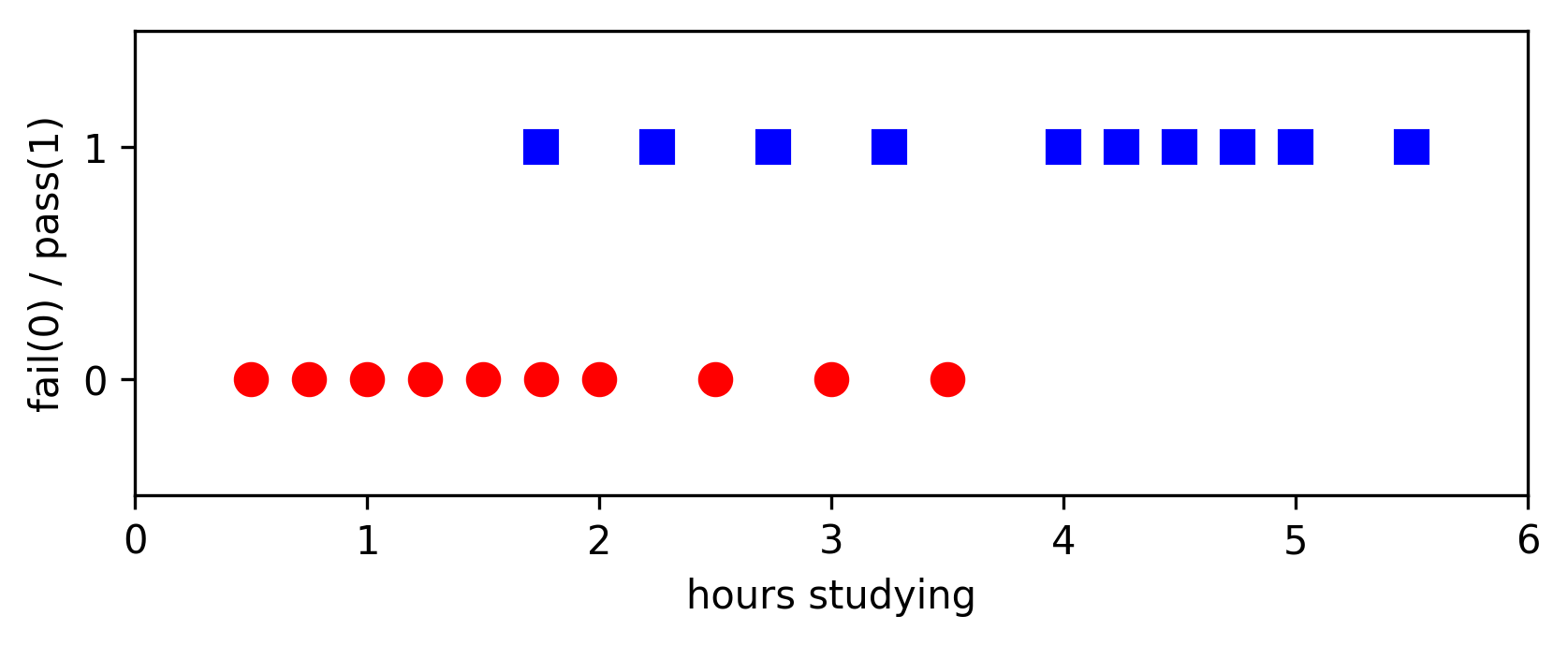
Trong giai đoạn tiếp theo, với trình phân loại rừng ngẫu nhiên được tạo ra, chúng ta sẽ đưa ra dự đoán. Mã giả đoán rừng ngẫu nhiên được hiển thị bên dưới: Đưa các tính năng thử nghiệm và sử dụng các quy tắc của từng cây quyết định được tạo ngẫu nhiên để dự đoán kết quả và lưu trữ kết quả dự đoán (mục tiêu). Tính số phiếu bầu cho từng mục tiêu được dự đoán. - Mục tiêu dự đoán được bình chọn cao là dự đoán cuối cùng từ thuật toán rừng ngẫu nhiên.

Ưu điểm của thuật toán Random Forest: Đối với các ứng dụng trong các vấn đề phân loại, thuật toán Random Forest sẽ tránh được vấn đề overfitting. Đối với cả nhiệm vụ phân loại và hồi quy, có thể sử dụng cùng một thuật toán rừng ngẫu nhiên. Thuật toán Random Forest có thể được sử dụng để xác định các tính năng quan trọng nhất từ tập dữ liệu huấn luyện, nói cách khác là kỹ thuật tính năng.

### 3.3.4 Thuật toán Logistic Regression( Logic hồi quy)

Hồi quy Logistic là một thuật toán hồi quy đặc biệt thường được sử dụng để phân lớp. Hồi quy Logistic là một phương pháp phân tích quan hệ giữa biến phụ thuộc Y với một hay nhiều biến độc lập X. Mô hình hóa sử dụng hàm Logistic. Các tham số của mô hình (hay hàm số) được ước lượng từ dữ liệu.

Ví dụ, để xét xem thời gian ôn thi của một nhóm sinh viên ảnh hưởng đến xác suất sinh viên vượt qua kỳ thi như thế nào. Đầu vào là thời gian ôn thi còn đầu ra là giá trị vượt qua kỳ thi hay không. Với các sinh viên vượt qua kỳ thi sẽ có giá trị pass là 1, sinh viên trượt sẽ có giá trị pass là 0. Dưới đây là một phân bố của kết quả thi.



Hình 3. Phân bố của kết quả thi

Dễ thấy, sinh viên học càng nhiều càng có khả năng đỗ cao, và ngược lại. Tuy nhiên không thể chắc chắn được sinh viên học nhiều sẽ đỗ và sinh viên học ít sẽ trượt. Hàm phân bố của kết quả thi trong trường hợp này không phải là hàm tuyến tính mà là một hàm Logistic.

Hàm Logistic là một hàm sigmoid, nhận vào các giá trị thực và xuất ra một giá trị từ 0 đến 1. Phương trình của nó là:

Hàm sigmoid có đặc điểm: Nếu coi điểm có tung độ là 1/2 làm điểm phân chia thì các điểm càng xa điểm này về phía bên trái có giá trị càng gần 0. Ngược lại, các điểm càng xa điểm này về phía phải có giá trị càng gần 1. Điều này khớp với nhận xét rằng học càng nhiều thì xác suất đỗ càng cao và ngược lại. Mượt (smooth) nên có đạo hàm mọi nơi, có thể được lợi trong việc tối ưu. Đạo hàm dễ tính, dễ tối ưu. Quá trình học hồi quy Logistic là đi tìm hàm Logistic phù hợp, hay hệ số t trong phương trình (1). Vận dụng thuyết phân phối chuẩn, có thể thấy rằng:

Hàm logistic có thể viết lại:

Việc học hồi quy Logistic là đi tìm 2 bộ tham số β0, β1 tốt nhất. Để tìm 2 bộ tham số này, ta sẽ tối ưu hàm mất mát sử dụng phương pháp đạo hàm Newton-Raphson, thu được :

Hồi quy Logistic thường được sử dụng trong các bài toán phân loại. Sau khi tìm được mô hình, việc xác định một điểm dữ liệu x có thuộc class y hay không được xác định bằng việc so sánh hai biểu thức xác suất:

với

Nếu biểu thức thứ nhất lớn hơn thì ta kết luận điểm dữ liệu thuộc class y, ngược lại thì nó không thuộc class này. Vì tổng hai biểu thức này luôn bằng 1 nên một cách gọn hơn, ta chỉ cần xác định xem lớn hơn 0.5 hay không. Nếu có, nó thuộc class y. Nếu không, nó không thuộc.

Mở rộng với bài toán nhiều nhãn lớp, có thể quy về bài toán một nhãn lớp rồi tính xác suất của nhãn lớp lớn nhất làm lớp đích. Hoặc có thể sử dụng mô hình xác suất nhiều nhóm softmax.

Ưu điểm của hồi quy Logistic: Có thể giải khá tốt các bài toán mà dữ liệu của nó là gần tuyến tính.

Nhược điểm của hồi quy Logistic: Dễ bị Overfitting

### 3.3.5 Thuật toán Navie Bayet

### 3.3.6 Thuật toán Cây quyết định

Chương 4. Phương pháp giải quyết bài toán

4.1 Thiết kế kiến trúc

4.2 Chi tiết phương pháp

Chương 5. Thử nghiệm, đánh giá

Chương 6. Phân tích và thiết kế hệ thống

6.1 Biều đồ usecase tổng quát

6.2 Đặc tả usecase

6.3 Biểu đồ hoạt động

6.4 Biểu đồ trình tự

6.5 Biểu đồ lớp cho ca sử dụng

6.6 Thiết kế cơ sở dữ liệu

6.7 Thiết kế giao diện

Chương 7. Cài đặt chương trình

Chương 8. Kết luận và hướng phát triển

8.1 Kết luận

8.2 Hướng phát triển

TÀI LIỆU THAM KHẢO

[1] Principles of Data Integration - AnHai Doan, Alon Halevy, Zachary Ives

[2] Machine learning

<https://codetudau.com/machine-learning-nlp-scikit-learn/index.html>

[3] Classification

<https://scikit-learn.org/stable/supervised_learning.html#supervised-learning>

[4] Schema

[https://schema.org](https://schema.org/)