

## Author Picks

FREE



# Exploring Data Science

Chapters selected by  
John Mount and Nina Zumel

manning



Khám phá khoa học dữ liệu

Lựa chọn của John Mount và Nina Zumel

Lựa chọn của tác giả Manning

Copyright 2016 Manning Publications Để đặt  
hàng trước hoặc tìm hiểu thêm về những cuốn sách này, hãy truy cập [www.manning.com](http://www.manning.com)

Để biết thông tin trực tuyến và đặt mua những cuốn sách này và những cuốn sách khác của Manning,  
vui lòng truy cập [www.manning.com](http://www.manning.com). Nhà xuất bản giảm giá cho những cuốn sách này khi đặt hàng với số lượng lớn.

Vui lòng liên lạc để biết thêm thông tin

Bộ phận bán hàng đặc biệt  
Công ty xuất bản Manning  
Đường 20 Baldwin  
Hộp thư bưu điện 761  
Đảo Shelter, NY 11964  
Email: [order@manning.com](mailto:order@manning.com)

©2016 của Manning Publications Co. Bảo lưu mọi quyền.

Không phần nào của ấn phẩm này có thể được sao chép, lưu trữ trong hệ thống truy xuất hoặc truyền đi dưới bất kỳ hình thức nào hoặc bằng phương tiện điện tử, cơ khí, sao chụp hoặc cách khác mà không có sự cho phép trước bằng văn bản của nhà xuất bản.

Nhiều ký hiệu được nhà sản xuất và người bán sử dụng để phân biệt sản phẩm của họ được tuyên bố là nhãn hiệu. Khi những ký hiệu đó xuất hiện trong cuốn sách và Manning Publications đã biết về khiếu nại nhãn hiệu, thì các ký hiệu đó đã được in hoa hoặc viết hoa toàn bộ.

- ⊗ Nhận thức được tầm quan trọng của việc bảo quản những gì đã được viết, chính sách của Manning là in những cuốn sách mà chúng tôi xuất bản trên giấy không có axit và chúng tôi nỗ lực hết mình để đạt được mục tiêu đó. Cũng thừa nhận trách nhiệm của chúng ta trong việc bảo tồn các nguồn tài nguyên trên hành tinh của chúng ta, sách Manning được in trên giấy được tái chế và xử lý ít nhất 15 phần trăm mà không sử dụng clo nguyên tố.

 Công ty xuất bản Manning  
20 Đường Baldwin Hộp thư kỵ  
thuật 761  
Đảo Shelter, NY 11964

Thiết kế trang bìa: Leslie Haimes

ISBN 9781617294181  
In tại Hoa Kỳ  
1 2 3 4 5 6 7 8 9 10 - EBM - 21 20 19 18 17 16

# nội dung

---

Giới thiệu iv

## KHÁM PHÁ DỮ LIỆU 1

khám phá dữ liệu

Chương 3 từ Khoa học dữ liệu thực tế với R 2

## THỜI GIAN SERIES 32

Chuỗi thời gian

Chương 15 từ R in Action, Phiên bản thứ hai 33

## HỌC SÂU VÀ MẠNG NƠ ÁN 63

Học sâu và mạng lưới thần kinh

Chương 6 từ Algorithms of the Intelligent Web, Second Edition 64

## KHAI THÁC VĂN BẢN VÀ PHÂN TÍCH VĂN BẢN 95

Khai thác văn bản và phân tích văn bản

Chương 8 từ Giới thiệu Khoa học Dữ liệu 96

## MÔ HÌNH PHỤ THUỘC VỚI BAYESIAN

VÀ MẠNG MARKOV 132

Lập mô hình phụ thuộc với mạng Bayesian và Markov

Chương 5 từ Lập trình xác suất thực tế 133

chỉ số 177

# Giới thiệu

---

Khoa học dữ liệu là một lĩnh vực rộng lớn liên quan đến các khía cạnh của thống kê, học máy và kỹ thuật dữ liệu. Các công cụ, phương pháp và công việc trông như thế nào phụ thuộc rất nhiều vào lĩnh vực vấn đề và quan điểm của bạn. Cuốn sách của chúng tôi, Khoa học dữ liệu thực tế với R, giới thiệu người đọc về mô hình dự đoán cơ bản bằng ngôn ngữ R. Nhưng chúng tôi chưa bao giờ có ý định ngụ ý rằng các nhà khoa học dữ liệu có thể giới hạn bản thân trong một miền vấn đề hoặc một ngôn ngữ triển khai.

Đây là thời điểm tuyệt vời để tham gia vào khoa học dữ liệu. Số lượng công cụ và tài liệu miễn phí đã bùng nổ. Lưu trữ và quản lý các tập dữ liệu lớn giờ đây dễ dàng hơn rõ rệt. Tuy nhiên, sự đa dạng này có vẻ quá sức và gây chia rẽ. Một nhà thống kê truyền thống có thể không coi phân tích văn bản là khoa học dữ liệu và tương tự, ai đó sử dụng mạng nơ-ron để phân tích hình ảnh có thể không đánh giá cao suy luận thống kê cổ điển.

Chúng tôi tin rằng vấn đề của bạn sẽ giúp chọn kỹ thuật của bạn. Để minh họa khái niệm này, chúng tôi đã tập hợp các chương lấy mẫu này từ cuốn sách của chúng tôi và các tựa sách khác của Manning. Chúng bao gồm các chủ đề đa dạng liên quan đến khoa học dữ liệu, làm nổi bật nhiều lĩnh vực và ngôn ngữ lập trình. Chúng tôi hy vọng những lựa chọn này cung cấp cho bạn hình ảnh tốt hơn về nhiều công cụ có sẵn để giải quyết các vấn đề khoa học dữ liệu cụ thể của bạn. Trên thực tế, chúng tôi hy vọng rằng sau khi bạn đánh giá cao những minh họa rõ ràng này về các lĩnh vực và phương pháp khác nhau, đối với miền của riêng bạn, bạn sẽ có một "Aha!" thời điểm mà bạn thấy một phương pháp cụ thể áp dụng như thế nào cho một vấn đề mà bạn quan tâm.

## khám phá dữ liệu

Dữ liệu      khám phá và làm sạch dữ liệu là những chủ đề thường bị bỏ qua, nhưng chúng cũng là phần quan trọng nhất—và tồn thời gian nhất—của quy trình khoa học dữ liệu. Nếu không có dữ liệu tốt, bạn không thể phát triển các mô hình hiệu quả hoặc thu thập thông tin quan trọng những hiểu biết sâu sắc. Môi trường tương tác và khả năng hiển thị của R khiến nó đặc biệt phù hợp với nhiệm vụ tìm hiểu dữ liệu của bạn. chương sau bao gồm một số vấn đề phổ biến về dữ liệu và cách phát hiện chúng thông qua trực quan hóa và các kỹ thuật thăm dò khác trong R. Chương này cũng giới thiệu về Gói trực quan hóa ggplot2 của R, một ngữ pháp linh hoạt để tạo các biểu đồ giàu thông tin và phong phú.

Chương 3 từ Khoa học dữ liệu thực tế với  
R của Nina Zumel và John Mount

# khám phá dữ liệu

## Chương này bao gồm

Sử dụng số liệu thống kê tóm tắt để khám phá dữ  
liệu      Khám phá dữ liệu bằng trực quan  
hóa      Phát hiện các vấn đề và sự cố trong quá trình  
                khám phá dữ liệu

Trong hai chương trước, bạn đã học cách đặt phạm vi và mục tiêu của dự án khoa học dữ liệu cũng như cách tải dữ liệu của bạn vào R. Trong chương này, chúng ta sẽ bắt đầu tìm hiểu dữ liệu.

Giả sử mục tiêu của bạn là xây dựng một mô hình để dự đoán khách hàng nào của bạn không có bảo hiểm y tế; có lẽ bạn muốn tiếp thị các gói bảo hiểm y tế rẻ tiền cho họ. Bạn đã thu thập một tập dữ liệu về những khách hàng có tình trạng bảo hiểm y tế mà bạn biết. Bạn cũng đã xác định được một số thuộc tính của khách hàng mà bạn tin rằng sẽ giúp dự đoán khả năng được bảo hiểm: tuổi, tình trạng việc làm, thu nhập, thông tin về nơi cư trú và xe cộ, v.v. Bạn đã đặt tất cả dữ liệu của mình vào một khung dữ liệu duy nhất có tên là custdata mà bạn đã nhập vào R.<sup>1</sup> Bây giờ, bạn đã sẵn sàng bắt đầu xây dựng mô hình để xác định những khách hàng mà bạn quan tâm.

<sup>1</sup> Chúng tôi có sẵn một bản sao của tập dữ liệu tổng hợp này để tải xuống từ <https://github.com/WinVector/zmPDSwR/tree/master/Custdata>, và sau khi đã lưu, bạn có thể tải nó vào R bằng lệnh `custdata <- read.table('custdata.tsv', header=T, sep='\t')`.

Thật hấp dẫn khi đi sâu vào bước lập mô hình mà không cần tìm hiểu kỹ về tập dữ liệu trước, đặc biệt là khi bạn có nhiều dữ liệu. Chóng lại sự cảm dỗ. Không có bộ dữ liệu nào là hoàn hảo: bạn sẽ thiếu thông tin về một số khách hàng của mình và bạn sẽ có dữ liệu không chính xác về những người khác. Một số trường dữ liệu sẽ bị bẩn và không nhất quán. Nếu bạn không dành thời gian để kiểm tra dữ liệu trước khi bắt đầu lập mô hình, bạn có thể thấy mình phải làm lại công việc của mình nhiều lần khi bạn phát hiện ra các trường dữ liệu hoặc biến không hợp lệ cần được chuyển đổi trước khi lập mô hình. Trong trường hợp xấu nhất, bạn sẽ xây dựng một mô hình trả về các dự đoán không chính xác—và bạn sẽ không biết chắc tại sao. Bằng cách giải quyết sớm các vấn đề về dữ liệu, bạn có thể tiết kiệm cho mình một số công việc không cần thiết và rất nhiều vấn đề đau đầu!

Bạn cũng muốn biết khách hàng của mình là ai: Họ là thanh niên, trung niên hay cao tuổi?

Họ giàu có như thế nào? Họ sống ở đâu? Biết câu trả lời cho những câu hỏi này có thể giúp bạn xây dựng một mô hình tốt hơn, bởi vì bạn sẽ có ý tưởng cụ thể hơn về thông tin nào dự đoán xác suất được bảo hiểm chính xác hơn.

Trong chương này, chúng tôi sẽ trình bày một số cách để tìm hiểu dữ liệu của bạn và thảo luận về một số vấn đề tiềm ẩn mà bạn đang tìm kiếm khi khám phá. Khám phá dữ liệu sử dụng kết hợp các số liệu thống kê tóm tắt—giá trị trung bình và trung bình, phương sai và số lượng—và trực quan hóa hoặc biểu đồ của dữ liệu. Bạn có thể phát hiện ra một số vấn đề chỉ bằng cách sử dụng thống kê tổng hợp; các vấn đề khác dễ dàng tìm thấy trực quan hơn.

#### Tổ chức dữ liệu để phân tích

phần lớn cuốn sách này, chúng tôi sẽ giả định rằng dữ liệu bạn đang phân tích nằm trong một khung dữ liệu duy nhất. Đây không phải là cách dữ liệu đó thường được lưu trữ. Ví dụ, trong cơ sở dữ liệu, dữ liệu thường được lưu trữ ở dạng chuẩn hóa để giảm thiểu: thông tin về một khách hàng được trải rộng trên nhiều bảng nhỏ. Trong dữ liệu nhật ký, dữ liệu về một khách hàng có thể trải rộng trên nhiều mục nhật ký hoặc phiên. Các định dạng này giúp dễ dàng thêm (hoặc trong trường hợp là cơ sở dữ liệu, sửa đổi) dữ liệu, nhưng không phải là tối ưu cho phân tích. Bạn thường có thể nối tất cả dữ liệu bạn cần vào một bảng duy nhất trong cơ sở dữ liệu bằng SQL, nhưng trong phần A, chúng ta sẽ thảo luận về các lệnh như nối mà bạn có thể sử dụng trong R để hợp nhất dữ liệu hơn nữa.

### 3.1 Sử dụng số liệu thống kê tóm tắt để phát hiện vấn đề

Trong R, thông thường bạn sẽ sử dụng lệnh tóm tắt để có cái nhìn đầu tiên về dữ liệu.

#### Lịch kê 3.1 Lệnh summary()

```
> tóm tắt (dữ liệu lưu ký)
lưu ký          tinh dục
tối thiểu : 2068 F:440
Qu.1: 345667 M:560
Trung vị : 693403
Có nghĩa là: 698500
Qu.3:1044606 Tối đa.
:1414286
```

is.employed	Ché	thu nhập	Biến is.employed bị thiếu khoảng một phần ba dữ liệu. Thu nhập biến có giá trị âm, có khả năng không hợp lệ.
độ: logic Min. : -8700			
SAI:73	Qu. 1: 14600		
ĐÚNG :599	Trung bình: 35000		
NA :328	Có nghĩa là: 53505		
	Quang 3: 67000		
	tối đa. :615000		
hôn nhân.stat			
Ly hôn/ly thân:155			
Đã cưới	:516		
Không bao giờ kết hôn	:233		
Góa chồng	: 96		
sức khỏe.ins			Khoảng 84% khách hàng có bảo hiểm y tế.
Ché độ: hợp lý			
SAI:159			
ĐÚNG :841			
Bắc Mỹ :0			
nhà Ø.type			Các biến housing.type, recent.move và num.vehicles đều thiếu 56 giá trị.
Chính chủ tự do, rõ ràng	:157		
Chủ nhà có thể chấp/vay vốn:412			
Không có tiền thuê : 11			
Đã thuê	:364		
NA	: 56		
gần đây.move		num.phương tiện	
Ché độ: logic Min. :0.000			
FALSE:820 Qu.1.000			
ĐÚNG :124 Trung bình :2.000			
Bắc Mỹ :56	Ý nghĩa :1.916		
	Qu.3:2.000		
	tối đa. :6.000		
	Bắc Mỹ :56		
tuổi		trạng thái.of.res	
tối thiểu : 0.0California :100			
Qu. 1: 38,0 New York: 71			
Trung bình : 50,0 Pennsylvania: 70			
Trung bình : 51,7 Texas : 56			
Qu. 3: 64,0 Michigan: 52			
tối đa. :146,7 Ohio	: 51		
(Khác)	:600		

Lệnh tóm tắt trên khung dữ liệu báo cáo nhiều thống kê tóm tắt về cột số của khung dữ liệu và đếm số liệu thống kê trên bất kỳ cột phân loại nào (nếu các cột phân loại đã được đọc dưới dạng factor2). Bạn cũng có thể yêu cầu thống kê tóm tắt trên các cột số cụ thể bằng cách sử dụng các lệnh mean, phương sai, trung vị, tối thiểu, tối đa và lượng tử (sẽ trả về các phần tử của dữ liệu theo mặc định).

<sup>2</sup> Các biến phân loại có yếu tố lớp trong R. Chúng có thể được biểu diễn dưới dạng chuỗi (ký tự lớp) và một số hàm phân tích sẽ tự động chuyển biến chuỗi thành biến nhân tố. Để có được một bắn tóm tắt của một biến, nó cần phải là một yếu tố.

Như bạn thấy trong danh sách 3.1, bản tóm tắt dữ liệu giúp bạn nhanh chóng phát hiện ra các vấn đề tiềm ẩn, chẳng hạn như dữ liệu bị thiếu hoặc các giá trị không chắc chắn. Bạn cũng có được một ý tưởng sơ bộ về cách dữ liệu phân loại được phân phối. Hãy đi vào chi tiết hơn về các vấn đề điển hình mà bạn có thể phát hiện bằng cách sử dụng bản tóm tắt.

### 3.1.1 Các vấn đề điển hình được tiết lộ bởi tóm tắt dữ liệu

Ở giai đoạn này, bạn đang tìm kiếm một số vấn đề phổ biến: thiếu giá trị, giá trị không hợp lệ và giá trị ngoại lai cũng như phạm vi dữ liệu quá rộng hoặc quá hẹp. Hãy giải quyết từng những vấn đề này một cách chi tiết.

#### GIÁ TRỊ BỊ MẤT

Một vài giá trị bị thiếu có thể không thực sự là vấn đề, nhưng nếu một trường dữ liệu cụ thể phần lớn không có dân cư, nó không nên được sử dụng làm đầu vào mà không sửa chữa (như chúng ta sẽ thảo luận trong chương 4, phần 4.1.1). Ví dụ, trong R, nhiều thuật toán mô hình hóa sẽ, bằng cách mặc định, lặng lẽ loại bỏ các hàng có giá trị bị thiếu. Như bạn thấy trong danh sách 3.2, tất cả những gì còn thiếu các giá trị trong biến `is.employed` có thể khiến R lặng lẽ bỏ qua gần một phần ba dữ liệu.

#### Lỗi kê 3.2 Biến `is.employed` có hữu ích cho việc lập mô hình không?

`is.employed`

Chế độ: hợp lý

SAI:73

ĐÚNG :599

NA :328



Biến `is.employed` bị thiếu khoảng một phần ba dữ liệu. Tại sao? Tình trạng việc làm không rõ? Có phải công ty mới bắt đầu thu thập dữ liệu việc làm gần đây? NA có nghĩa là "không thuộc lực lượng lao động đang hoạt động" (ví dụ: sinh viên hoặc phụ huynh nội trợ)?

`nhà ở.type`

:157



Các biến `housing.type`, `recent.move` và `num.vehicles` chỉ thiếu một vài giá trị. Có thể an toàn nếu chỉ loại bỏ các hàng thiếu giá trị đặc biệt nếu các giá trị bị thiếu đều có 56 hàng giống nhau.

Chính chủ tự do, rõ ràng

Chủ nhà có thể chấp/vay vốn:412

Không có tiền thuê : 11

Đã thuê

:364

NA

: 56

gần đây.move

num.phương tiện

Chế độ: logic Min. :0.000

SAI:820

Qu.1:1.000

ĐÚNG :124

Trung bình :2.000

Bắc Mỹ :56

Ý nghĩa :1.916

Qu.3:2.000

tối đa. :6.000

Bắc Mỹ :56

Nếu một trường dữ liệu cụ thể phần lớn không được phô biến, bạn nên cố gắng xác định lý do tại sao;

đôi khi thực tế là một giá trị bị thiếu là thông tin trong chính nó. Ví dụ,

tại sao biến `is.employed` lại thiếu nhiều giá trị như vậy? Có rất nhiều lý do có thể xảy ra, như chúng tôi đã lưu ý trong danh sách 3.2.

Dù lý do thiếu dữ liệu là gì, bạn phải quyết định cách thích hợp nhất

hoạt động. Bạn có bao gồm một biến có giá trị bị thiếu trong mô hình của mình hay không? Nếu bạn

quyết định bao gồm nó, bạn có bỏ tất cả các hàng mà trường này bị thiếu hay bạn chuyển đổi các giá trị bị thiếu thành 0 hoặc thành một danh mục bổ sung? Chúng ta sẽ thảo luận về cách điều trị thiếu dữ liệu trong chương 4. Trong ví dụ này, bạn có thể quyết định loại bỏ các hàng dữ liệu nơi bạn đang thiếu dữ liệu về nhà ở hoặc xe cộ, vì không có nhiều dữ liệu trong số đó. Bạn có thể không muốn vứt bỏ dữ liệu mà bạn đang thiếu việc làm thông tin, mà thay vào đó coi các NA là một loại việc làm thứ ba. bạn sẽ có khả năng gặp phải các giá trị bị thiếu khi chấm điểm mô hình, vì vậy bạn nên xử lý chúng trong quá trình đào tạo người mẫu.

#### GIÁ TRỊ VÀ NGOẠI LỆ KHÔNG HỢP LỆ

Ngay cả khi một cột hoặc biến không thiếu bất kỳ giá trị nào, bạn vẫn muốn kiểm tra xem các giá trị mà bạn có có ý nghĩa. Bạn có bất kỳ giá trị không hợp lệ hoặc ngoại lệ nào không? Ví dụ về các giá trị không hợp lệ bao gồm các giá trị âm trong giá trị phải là giá trị không âm trường dữ liệu số (chẳng hạn như tuổi hoặc thu nhập) hoặc văn bản mà bạn muốn có số. ngoại lệ là các điểm dữ liệu nằm ngoài phạm vi mà bạn mong đợi dữ liệu. Có thể bạn phát hiện ra các giá trị ngoại lệ và không hợp lệ trong danh sách 3.3?

#### Lỗi kê 3.3 Ví dụ về giá trị không hợp lệ và ngoại lệ

```
> tóm tắt(custdata$thu nhập)
tối thiểu 1 Qu. Trung bình có nghĩa là 3 Qu.
-8700 14600 35000 53500 67000
tối đa.
615000
```

Giá trị âm cho thu nhập có thể chỉ ra dữ liệu xấu. Chúng cũng có thể có một ý nghĩa đặc biệt, chẳng hạn như "số tiền nợ".

Để bằng cách nào, bạn nên kiểm tra mức độ phổ biến của vấn đề và quyết định phải làm gì: Bạn có bỏ dữ liệu với thu nhập âm không? Bạn có chuyển đổi các giá trị âm thành 0 không?

```
> tóm tắt(custdata$age)
tối thiểu 1 Qu. Trung bình có nghĩa là 3 Qu.
0,0      38,0      50,0      51,7      64,0
tối đa.
146,7
```

Khách hàng ở độ tuổi 0 hoặc khách hàng ở độ tuổi lớn hơn khoảng 110 là ngoại lệ. Chúng nằm ngoài phạm vi giá trị mà khách hàng mong đợi.

Ngoại lệ có thể là lỗi nhập dữ liệu. Chúng có thể là các giá trị trọng điểm đặc biệt: số 0 có thể có nghĩa là "không biết tuổi" hoặc "từ chối trạng thái". Và một số khách hàng của bạn có thể đặc biệt tồn tại lâu dài.

Thông thường, các giá trị không hợp lệ chỉ đơn giản là dữ liệu đầu vào không hợp lệ. Số âm trong một trường như tuổi, tuy nhiên, có thể là một giá trị trọng điểm để chỉ định "không xác định". Ngoại lệ cũng có thể là dữ liệu lỗi hoặc giá trị trọng điểm. Hoặc chúng có thể là những điểm dữ liệu hợp lệ nhưng không bình thường—mọi người sinh thường sống qua 100.

Đối với các giá trị bị thiếu, bạn phải quyết định hành động phù hợp nhất: loại bỏ dữ liệu trường, loại bỏ các điểm dữ liệu mà trường này không hợp lệ hoặc chuyển đổi dữ liệu không hợp lệ thành hữn ích giá trị. Ngay cả khi bạn cảm thấy một số ngoại lệ nhất định là dữ liệu hợp lệ, bạn vẫn có thể muốn bỏ qua chúng từ việc xây dựng mô hình (và cả phạm vi dự đoán cho phép cổ áo), vì thông thường mục tiêu có thể đạt được của mô hình hóa là dự đoán chính xác trường hợp điển hình.

#### PHẠM VI DỮ LIỆU

Bạn cũng muốn chú ý đến mức độ thay đổi của các giá trị trong dữ liệu. Nếu bạn tin độ tuổi hoặc thu nhập đó giúp dự đoán khả năng được bảo hiểm y tế, sau đó

bạn nên đảm bảo có đủ sự khác biệt về độ tuổi và thu nhập của khách hàng để bạn thấy được các mối quan hệ. Hãy xem lại thu nhập, trong danh sách 3.4. Phạm vi dữ liệu có rỗng không? Nó có hợp không?

#### Liệt kê 3.4 Nhìn vào phạm vi dữ liệu của một biến

```
> tóm tắt(custdata$thu nhập)
tối thiểu 1 Qu. Trung bình có nghĩa là 3 Qu. -8700 14600
35000 53500 67000
tối đa.
615000
```

Thu nhập dao động từ 0 đến  
hơn nửa triệu đô la; một  
phạm vi rất rộng.

Ngay cả khi bỏ qua thu nhập âm, biến thu nhập trong danh sách 3.4 dao động từ 0 đến hơn nửa triệu đô la. Đó là khá rộng (mặc dù điển hình cho thu nhập). Dữ liệu nằm trong một số bậc độ lớn như thế này có thể là một vấn đề đối với một số phương pháp lập mô hình. Chúng ta sẽ nói về việc giảm thiểu các vấn đề về phạm vi dữ liệu khi chúng ta nói về các phép biến đổi logarit trong chương 4.

Dữ liệu cũng có thể quá hẹp. Giả sử tất cả khách hàng của bạn ở độ tuổi từ 50 đến 55. Bạn nên đặt cược rằng độ tuổi sẽ không phải là yếu tố dự đoán tốt về xác suất bảo hiểm y tế cho nhóm dân số đó, vì nó không thay đổi nhiều.

#### Phạm vi dữ liệu "quá hẹp" hẹp đến mức nào?

Tất nhiên, thuật ngữ hẹp là tương đối. Nếu chúng ta dự đoán khả năng đọc của trẻ em trong độ tuổi từ 5 đến 10, thì tuổi có lẽ là một biến số hữu ích. Đối với dữ liệu bao gồm độ tuổi trưởng thành, bạn có thể muốn chuyển đổi hoặc phân loại độ tuổi theo một cách nào đó, vì bạn không mong đợi có sự thay đổi đáng kể về khả năng đọc giữa độ tuổi 40 và 50. Bạn nên dựa vào thông tin về miền vẫn để đánh giá xem liệu dữ liệu có phạm vi là hàng hẹp, nhưng một nguyên tắc chung là tỷ lệ của độ lệch chuẩn so với giá trị trung bình. Nếu tỷ lệ đó rất nhỏ, thì dữ liệu không thay đổi nhiều.

Chúng ta sẽ xem lại phạm vi dữ liệu trong phần 3.2, khi chúng ta nói về việc kiểm tra dữ liệu bằng đồ thị.

Một yếu tố xác định phạm vi dữ liệu rõ ràng là đơn vị đo lường. Lấy một ví dụ phi kỹ thuật, chúng tôi đo độ tuổi của trẻ sơ sinh và trẻ mới biết đi theo tuần hoặc theo tháng, bởi vì những thay đổi phát triển xảy ra ở quy mô thời gian đó đối với trẻ nhỏ. Giả sử chúng ta tính tuổi của các em bé theo năm. Về mặt số học, có vẻ như không có nhiều khác biệt giữa trẻ một tuổi và trẻ hai tuổi. Trên thực tế, có một sự khác biệt đáng kể mà bất kỳ phụ huynh nào cũng có thể nói với bạn! Các đơn vị cũng có thể trình bày các vấn đề tiềm ẩn trong tập dữ liệu vì một lý do khác.

#### CÁC ĐƠN VỊ

Dữ liệu thu nhập trong danh sách 3.5 có biểu thị tiền lương theo giờ hoặc tiền lương hàng năm theo đơn vị \$1000 không? Trên thực tế, nó là cái sau, nhưng nếu bạn nghĩ nó là cái trước thì sao? Bạn có thể không nhận thấy lỗi trong giai đoạn lập mô hình, nhưng cuối cùng, ai đó sẽ bắt đầu nhập dữ liệu tiền lương theo giờ vào mô hình và đổi lại nhận được các dự đoán sai.

**Liet kê 3.5 Các đơn vị kiểm tra có thể ngăn các kết quả không chính xác sau này**

> tóm tắt(Thu nhập)  
 tối thiểu 1 Qu. Trung bình có nghĩa là 3 Qu. 53,5  
 -8,7      14,6      35,0      67,0      615,0

tối đa.

Biến Thu nhập được định nghĩa là  
 $\text{Thu nhập} = \text{custdata\$ income}/1000$ . Nhưng giả sử bạn không  
 biết điều đó. Chỉ nhìn vào bản  
 tóm tắt, các giá trị có thể  
 được hiểu một cách hợp lý là "tiền  
 lương theo giờ" hoặc "thu nhập  
 hàng năm tính theo đơn vị 1000 đô la".

Các khoảng thời gian được đo bằng ngày, giờ, phút hay mili giây? Là tốc độ trong  
 km trên giây, dặm trên giờ, hay hải lý? Là số tiền tính bằng đô la,  
 hàng nghìn đô la, hay 1/100 của một xu (một thông lệ trong lĩnh vực tài chính, trong đó các phép tính toán  
 thường được thực hiện trong số học điểm cố định)? Đây thực sự là một cái gì đó mà  
 bạn sẽ nắm bắt bằng cách kiểm tra định nghĩa dữ liệu trong từ điển dữ liệu hoặc tài liệu,  
 chứ không phải trong số liệu thống kê tóm tắt; sự khác biệt giữa dữ liệu lương theo giờ và  
 tiền lương hàng năm tính theo đơn vị 1000 đô la có thể không rõ ràng như vậy nếu nhìn thoáng qua. Nhưng nó là  
 vẫn còn điều gì đó cần lưu ý khi xem xét phạm vi giá trị của các biến của bạn,  
 bởi vì bạn thường có thể phát hiện ra khi các phép đo ở các đơn vị không mong muốn. ô tô  
 tốc độ tính theo hải lý trông rất khác so với tốc độ tính bằng dặm một giờ.

### 3.2 Phát hiện các vấn đề bằng đồ họa và trực quan

Như bạn đã thấy, bạn có thể phát hiện ra nhiều vấn đề chỉ bằng cách xem qua các bản tóm tắt dữ liệu. Đối với  
 các thuộc tính khác của dữ liệu, hình ảnh tốt hơn vẫn bẩn.

Chúng ta không thể mong đợi một số lượng nhỏ các giá trị số [thống kê tóm tắt]  
 truyền đạt một cách nhất quán sự giàu có của thông tin tồn tại trong dữ liệu. Giảm số  
 các phương pháp không giữ lại thông tin trong dữ liệu.

-William Cleveland

Các yếu tố của dữ liệu vẽ đồ thị

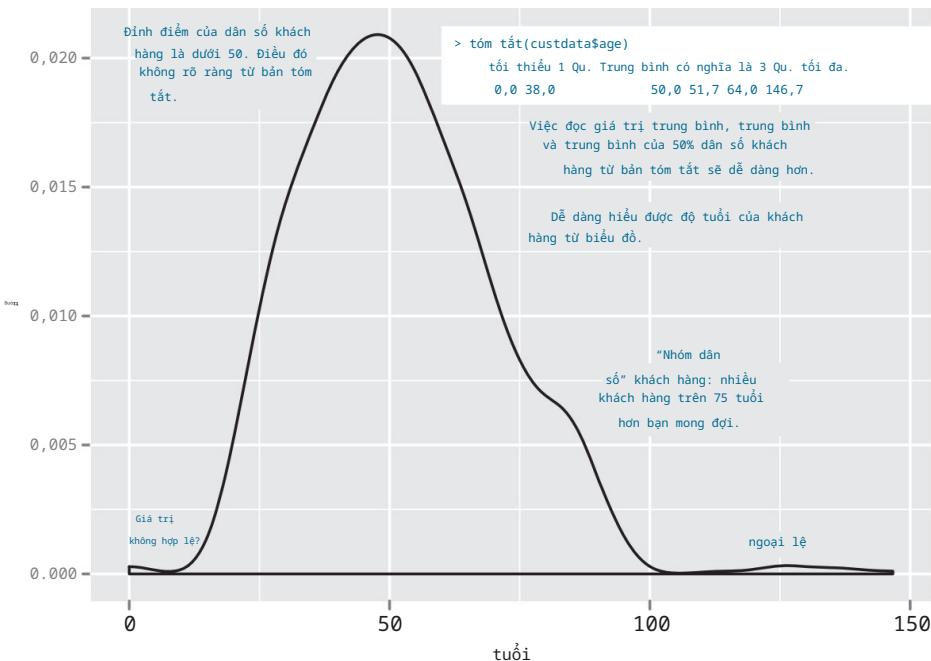
Hình 3.1 cho thấy một biểu đồ về cách phân bố độ tuổi của khách hàng. Chúng ta sẽ nói về những gì  
 trực y của đồ thị có nghĩa là sau này; ngay bây giờ, chỉ cần biết rằng chiều cao của đồ thị  
 tương ứng với bao nhiêu khách hàng trong dân số ở độ tuổi đó. Bạn có thể thấy,  
 thông tin như tuổi cao nhất của phân phối, sự tồn tại của quần thể con, và  
 sự hiện diện của các ngoại lệ dễ tiếp thu trực quan hơn là xác định bằng văn bản.

Việc sử dụng đồ họa để kiểm tra dữ liệu được gọi là trực quan hóa. Chúng tôi có gắng làm theo William  
 Các nguyên tắc của Cleveland để hình dung khoa học. Thông tin chi tiết về các lô đất cụ thể sang một bên, chính  
 những điểm trong triết lý của Cleveland là:

Một đồ họa nên hiển thị càng nhiều thông tin càng tốt, với mức thấp nhất có thể  
 cảng thẳng nhận thức cho người xem.

Phần đầu cho rõ ràng. Làm nổi bật dữ liệu. Mẹo cụ thể để tăng độ rõ ràng  
 bao gồm

-Tránh quá nhiều yếu tố chồng lên nhau, chẳng hạn như quá nhiều đường cong trong cùng một  
 không gian vẽ đồ thị.



Hình 3.1 Một số thông tin dễ đọc hơn từ biểu đồ và một số thông tin từ bản tóm tắt.

-Tim tỷ lệ khung hình phù hợp và chia tỷ lệ để hiển thị chính xác các chi tiết của dữ liệu.

-Tránh để tất cả dữ liệu bị lệch sang một bên hoặc bên kia biểu đồ của bạn.

Trực quan hóa là một quá trình lặp đi lặp lại. Mục đích của nó là để trả lời các câu hỏi về dữ liệu.

Trong giai đoạn trực quan hóa, bạn vẽ biểu đồ dữ liệu, tìm hiểu những gì bạn có thể và sau đó vẽ lại dữ liệu để trả lời các câu hỏi phát sinh từ biểu đồ trước đó của bạn. Đồ họa khác nhau phù hợp nhất để trả lời các câu hỏi khác nhau. Chúng ta sẽ xem xét một số trong số họ trong phần này.

Trong cuốn sách này, chúng tôi sử dụng ggplot2 để minh họa các hình ảnh và đồ họa; của tất nhiên, các gói trực quan hóa R khác có thể tạo ra đồ họa tương tự.

### Một lưu ý về ggplot2

Chủ đề của phần này là cách sử dụng trực quan hóa để khám phá dữ liệu của bạn, chứ không phải cách sử dụng ggplot2. Chúng tôi đã chọn ggplot2 vì nó vượt trội trong việc kết hợp nhiều yếu tố đồ họa với nhau, nhưng cú pháp của nó có thể mất một số thời gian để làm quen. Những điểm chính cần hiểu khi xem các đoạn mã của chúng tôi là:

Đồ thị trong ggplot2 chỉ có thể được xác định trên khung dữ liệu. Các biến trong biểu đồ-biến x, biến y, biến xác định màu hoặc

kích thước của các điểm—được gọi là *thảm mĩ* và được khai báo bằng cách sử dụng hàm aes .

Hàm ggplot() khai báo đối tượng đồ thị. Các đối số cho ggplot() có thể bao gồm khung dữ liệu quan tâm và tính thảm mĩ. Bản thân hàm ggplot () không tạo ra hình ảnh trực quan; hình ảnh được tạo ra bởi các lớp.

Các lớp tạo ra các biểu đồ và phép biến đổi biểu đồ và được thêm vào một đối tượng biểu đồ đã cho bằng cách sử dụng toán tử + . Mỗi lớp cũng có thể lấy một khung dữ liệu và tính thảm mĩ làm đối số, bên cạnh các tham số dành riêng cho cốt truyện. Ví dụ về các lớp là geom\_point (đối với biểu đồ phân tán) hoặc geom\_line (đối với biểu đồ đường).

Cú pháp này sẽ trở nên rõ ràng hơn trong các ví dụ sau. Để biết thêm thông tin, chúng tôi đề xuất trang web tham khảo của Hadley Wickham <http://ggplot2.org>, trong đó có gợi ý đến tài liệu trực tuyến, cũng như ggplot2 của Tiến sĩ Wickham: Đồ họa thanh lịch để phân tích dữ liệu (Sử dụng R!) (Springer, 2009).

Trong hai phần tiếp theo, chúng tôi sẽ trình bày cách sử dụng hình ảnh và biểu đồ để xác định các đặc điểm và vấn đề của dữ liệu. Trong phần 3.2.2, chúng ta sẽ xem xét trực quan hóa cho hai biến.

Nhưng hãy bắt đầu bằng cách xem hình ảnh trực quan cho các biến đơn lẻ.

### 3.2.1 Kiểm tra trực quan các bản phân phối cho một biến

Hình ảnh trực quan trong phần này giúp bạn trả lời các câu hỏi như sau:

Giá trị định của phân bố là bao nhiêu? Có bao nhiêu định trong phân phối (đơn thức và lưỡng thức)? Dữ liệu bình thường (hoặc logic) như thế nào? Chúng ta sẽ thảo luận về normal và lognormal phân phối trọng phu lực B.

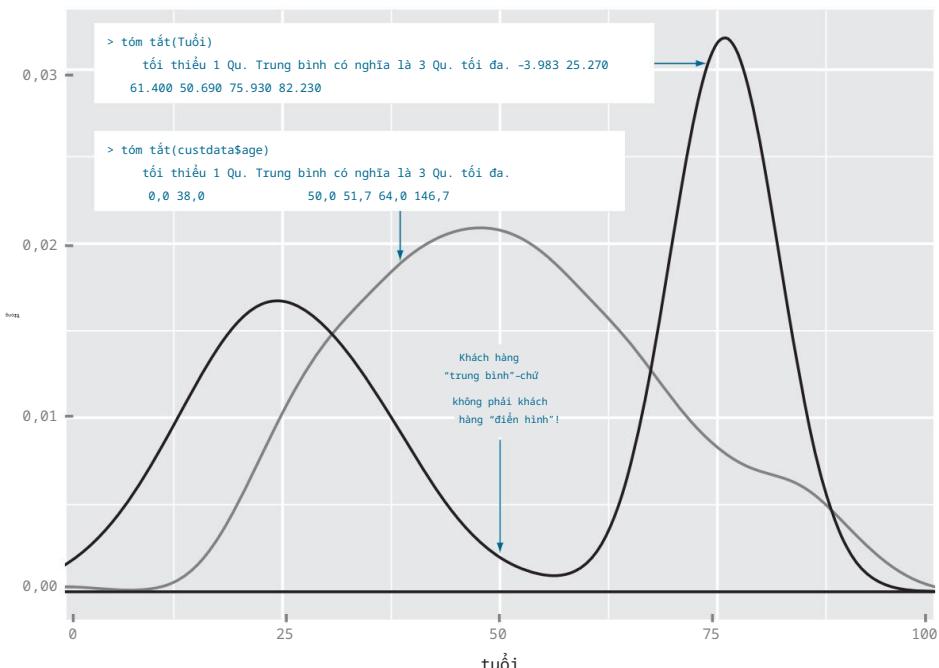
Dữ liệu thay đổi bao nhiêu? Nó tập trung trong một khoảng thời gian nhất định hay trong một khoảng thời gian nhất định thuộc loại nào?

Một trong những điều dễ nắm bắt trực quan hơn là hình dạng của phân phối dữ liệu.

Ngoại trừ điểm sáng bên phải, đồ thị trong hình 3.1 (mà chúng tôi đã sao chép dưới dạng đường cong màu xám trong hình 3.2) gần như có hình dạng giống như phân phối chuẩn (xem phụ lục B). Như phụ lục đó giải thích, nhiều thống kê tóm tắt giả định rằng dữ liệu có phân phối xấp xỉ chuẩn (ít nhất là đối với các biến liên tục), vì vậy bạn muốn xác minh xem đây có phải là trường hợp không.

Bạn cũng có thể thấy rằng đường cong màu xám trong hình 3.2 chỉ có một đỉnh hoặc nó là một đỉnh. phương thức. Đây là một thuộc tính khác mà bạn muốn kiểm tra trong dữ liệu của mình.

Tại sao? Bởi vì (nói một cách đại khái), phân phối không theo phương thức tương ứng với một quần thể đối tượng. Đối với đường cong màu xám trong hình 3.2, độ tuổi trung bình của khách hàng là khoảng 52 và 50% khách hàng trong độ tuổi từ 38 đến 64 (phần tư thứ nhất và thứ ba). Vì vậy, bạn có thể nói rằng một khách hàng “diễn hình” là trung niên và có thể sở hữu nhiều phẩm chất nhân khẩu học của một người trung niên—mặc dù tất nhiên bạn phải xác minh điều đó với thông tin khách hàng thực tế của mình.



Hình 3.2 Một phân phối không theo phương thức (màu xám) thường có thể được lập mô hình như đến từ một nhóm người dùng duy nhất. Với phân phối hai chiều (màu đen), dữ liệu của bạn thường đến từ hai nhóm người dùng.

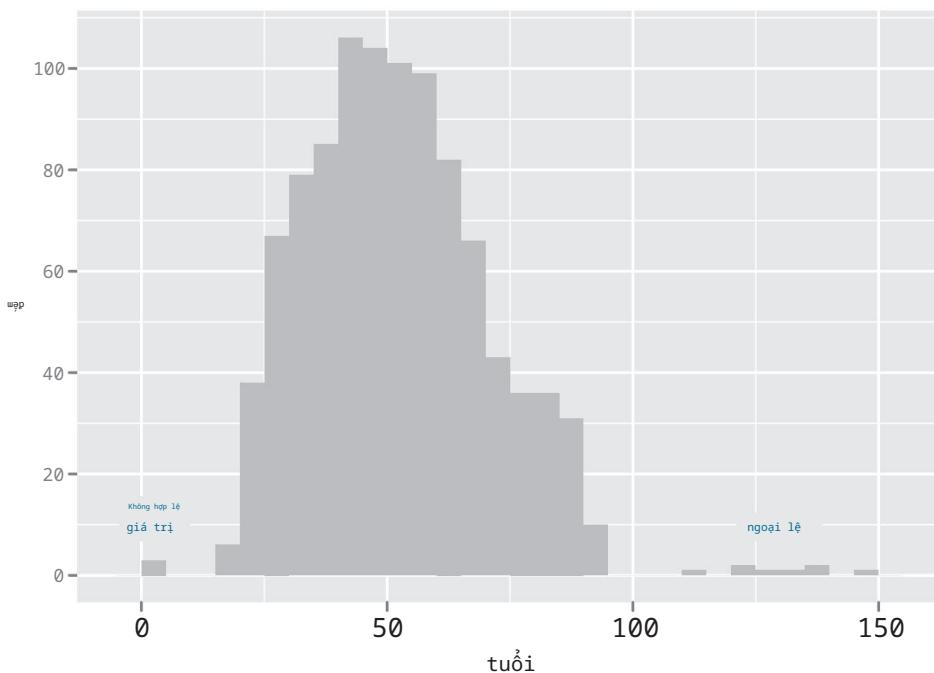
Đường cong màu đen trong hình 3.2 cho thấy điều gì có thể xảy ra khi bạn có hai cực đại hoặc phân bố lưỡng cực. (Phân phối có nhiều hơn hai đỉnh là đa phương thức.) Nhóm khách hàng này có cùng độ tuổi trung bình với những khách hàng được biểu thị bằng đường màu xám—nhưng một người 50 tuổi khó có thể là khách hàng “diễn hình”! Ví dụ (phải thừa nhận là phóng đại) này tương ứng với hai nhóm khách hàng: nhóm dân số khá trẻ chủ yếu ở độ tuổi 20 và 30 và nhóm dân số lớn hơn chủ yếu ở độ tuổi 70. Hai quần thể này có thể có các mẫu hành vi rất khác nhau và nếu bạn muốn lập mô hình liệu một khách hàng có thể có bão hiểm y tế hay không, thì không nên lập mô hình hai quần thể riêng biệt –đặc biệt nếu bạn đang sử dụng tuyến tính hoặc hậu cần hồi quy.

Biểu đồ tần suất và biểu đồ mật độ là hai cách trực quan hóa giúp bạn nhanh chóng kiểm tra sự phân bố của một biến số. Hình 3.1 và 3.2 là các ô mật độ.

Cho dù bạn sử dụng biểu đồ hay biểu đồ mật độ phần lớn là vấn đề sở thích. Chúng tôi có xu hướng thích biểu đồ mật độ hơn, nhưng biểu đồ dễ giải thích hơn đối với những khán giả ít quan tâm đến định lượng hơn.

### BIỂU ĐỒ

Một biểu đồ cơ bản sắp xếp một biến thành các nhóm có chiều rộng cố định và trả về số điểm dữ liệu rơi vào mỗi nhóm. Ví dụ: bạn có thể nhóm khách hàng của mình theo độ tuổi, trong khoảng thời gian 5 năm: 20-25, 25-30, 30-35, v.v. Khách hàng tại một



Hình 3.3 Một biểu đồ cho bạn biết dữ liệu của bạn được tập trung ở đâu. Nó cũng trực quan làm nổi bật các ngoại lệ và dị thường.

tuổi ranh giới sẽ đi vào nhóm cao hơn: những người 25 tuổi đi vào nhóm 25-30. Đối với mỗi nhóm, sau đó bạn có thể xem có bao nhiêu khách hàng trong nhóm đó. Biểu đồ kết quả được hiển thị trong hình 3.3.

Bạn tạo biểu đồ trong hình 3.3 trong ggplot2 bằng lớp geom\_histogram .

#### Lệnh kẽ 3.6 Vẽ một biểu đồ

```
thư viện (ggplot2)
ggplot(đữ liệu lưu ký) +
  geom_histogram(aes(x=age),
  binwidth=5, fill="xám")
```

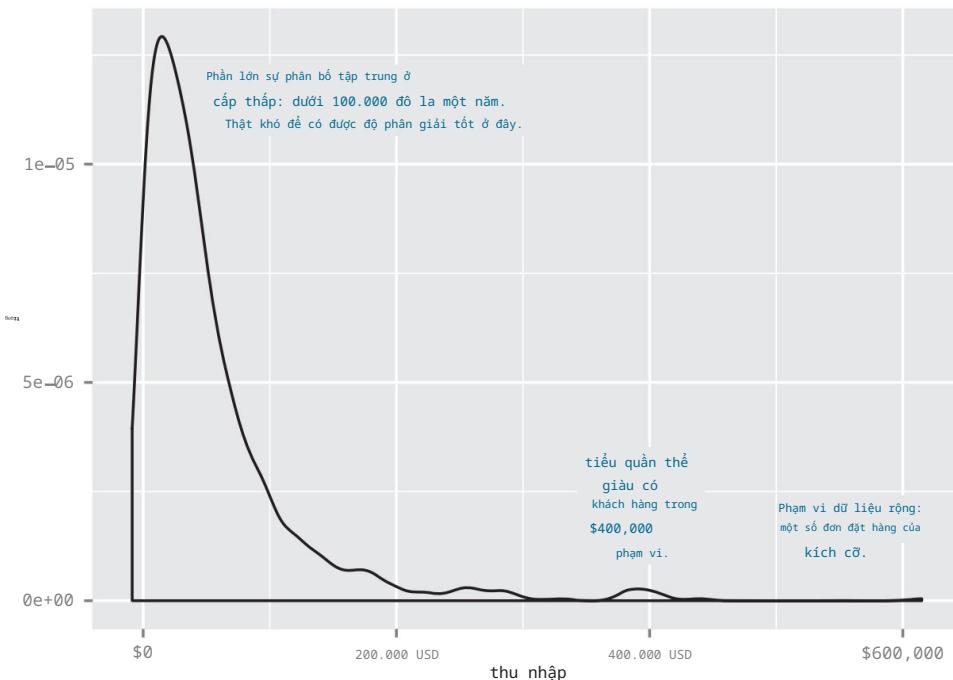
Tải thư viện ggplot2, nếu bạn chưa làm như vậy.

Tham số bằng thông báo cho lệnh gọi geom\_histogram cách tạo các thùng có khoảng thời gian 5 năm (mặc định là datarange/30). Tham số điều chỉ định màu của các thanh biểu đồ (mặc định: màu đen).

Nhược điểm chính của biểu đồ là bạn phải quyết định trước cách rộng các thùng được. Nếu các thùng quá rộng, bạn có thể mất thông tin về hình dạng phân phối. Nếu các thùng quá hẹp, biểu đồ có thể trông quá ôn ào để đọc dễ dàng. Một cách trực quan khác là biểu đồ mật độ.

#### KHU VỰC MẬT ĐỘ

Bạn có thể coi biểu đồ mật độ là một “biểu đồ liên tục” của một biến, ngoại trừ diện tích dưới biểu đồ mật độ bằng 1. Một điểm trên biểu đồ mật độ tương ứng với



Hình 3.4 Biểu đồ mật độ hiển thị nơi dữ liệu được tập trung. Cốt truyện này cũng làm nổi bật một nhóm khách hàng có thu nhập cao hơn.

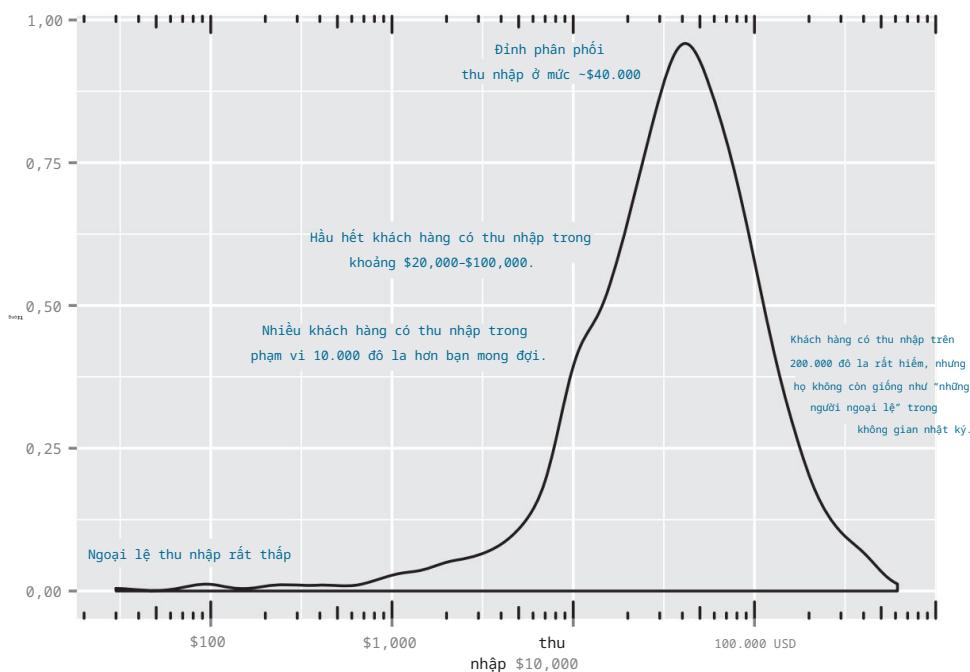
phân dữ liệu (hoặc tỷ lệ phân trăm dữ liệu, chia cho 100) đảm nhận một mục tiêu cụ thể giá trị. Phân số này thường rất nhỏ. Khi bạn nhìn vào biểu đồ mật độ, bạn sẽ quan tâm đến hình dạng tổng thể của đường cong hơn là các giá trị thực tế trên trục y. Bạn đã thấy biểu đồ mật độ tuổi; hình 3.4 thể hiện biểu đồ mật độ thu nhập. Bạn tạo hình 3.4 với lớp geom\_density , như trong danh sách sau.

#### Liệt kê 3.7 Tạo biểu đồ mật độ

```
thư viện (quy mô)
ggplot(custdata) + geom_density(aes(x=thu nhập)) +
  scale_x_continuity(nhãn=dô la)
```

Gói tỷ lệ mang đến ký hiệu  
tỷ lệ đô la.  
←———— Đặt nhãn trục x thành đô la.

Khi phạm vi dữ liệu rất rộng và khối lượng của phân bố tập trung nhiều về một phía, như phân bố trong hình 3.4, rất khó để xem chi tiết của nó. Hình dạng. Ví dụ, thật khó để nói giá trị chính xác mà phân phối thu nhập có đỉnh cao của nó. Nếu dữ liệu không âm, thì một cách để đưa ra chi tiết hơn là vẽ đồ thị phân phối theo thang logarit, như thể hiện trong hình 3.5. Điều này tương đương với vẽ đồ thị mật độ của  $\log_{10}(\text{thu nhập})$ .



Hình 3.5 Biểu đồ mật độ thu nhập trên thang log10 làm nổi bật các chi tiết về phân phối thu nhập khó thấy hơn trong biểu đồ mật độ thông thường.

Trong ggplot2, bạn có thể vẽ hình 3.5 với các lớp geom\_density và scale\_x\_log10 , chặng hạn như trong danh sách tiếp theo.

**Lệnh 3.8 Tạo biểu đồ mật độ theo tỷ lệ log**

```
Đặt trục x ở tỷ lệ log10, với các điểm đánh dấu và  
nhân được đặt theo cách thủ công là đồ la.  
ggplot(custdata) + geom_density(aes(x=thu nhập)) +  
scale_x_log10(breaks=c(100,1000,10000,100000), nhân=dô la) + annotation_logticks(sides="bt")  
←  
←  
Thêm các dấu tick theo tỷ lệ log vào  
trên cùng và dưới cùng của biểu đồ.
```

Khi bạn đưa ra lệnh trước đó, bạn cũng nhận được một thông báo cảnh báo:

Thông báo cảnh báo: 1:  
Trong scale\$trans\$trans(x) : NaN được tạo ra 2: Đã xóa 79 hàng  
chứa giá trị không hữu hạn (stat\_density).

Điều này cho bạn biết rằng ggplot2 đã bỏ qua các hàng có giá trị bằng 0 và âm (vì  $\log(0) = \text{Infinity}$ ), và có 79 hàng như vậy. Hãy ghi nhớ điều đó khi đánh giá biểu đồ.

Trong không gian nhật ký, thu nhập được phân phối dưới dạng một thứ trông giống như một phân phối “bình thường”, như sẽ được thảo luận trong phụ lục B. Nó không hoàn toàn là một phân phối bình thường (thực tế, có vẻ như ít nhất hai phân phối chuẩn được trộn lẫn với nhau).

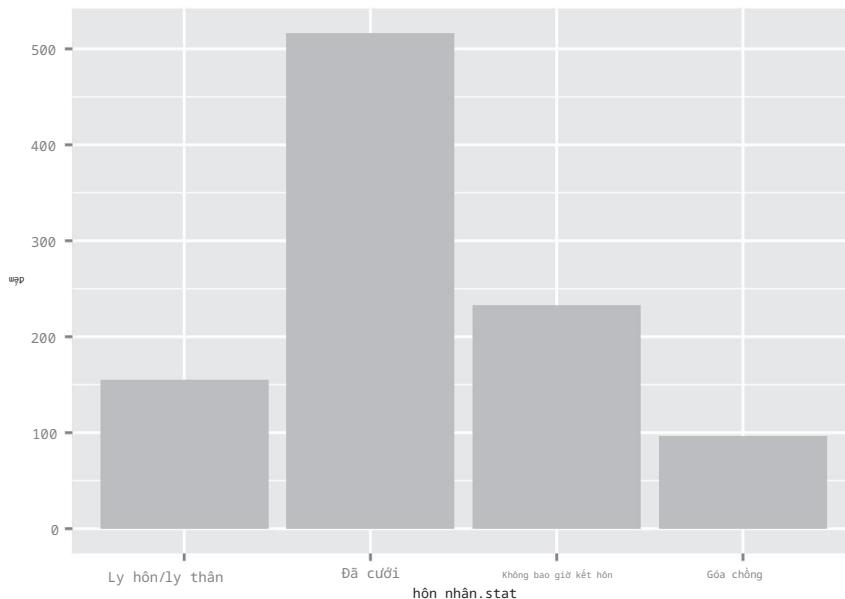
### Khi nào bạn nên sử dụng thang logarit?

Bạn nên sử dụng thang logarit khi thay đổi phần trăm hoặc thay đổi theo thứ tự độ lớn quan trọng hơn thay đổi theo đơn vị tuyệt đối. Bạn cũng nên sử dụng nhật ký chia tỷ lệ để trực quan hóa tốt hơn dữ liệu bị sai lệch nhiều.

Ví dụ: trong dữ liệu thu nhập, chênh lệch thu nhập năm nghìn đô la có nghĩa là một cái gì đó rất khác nhau trong dân số nơi thu nhập có xu hướng giảm trong hàng chục hàng ngàn đô la so với ở những quần thể có thu nhập rời vào hàng trăm hàng ngàn hoặc hàng triệu đô la. Nói cách khác, điều tạo nên "sự khác biệt đáng kể" phụ thuộc vào thứ tự độ lớn của thu nhập mà bạn đang xem xét. Tương tự, trong dân số như hình 3.5, một số ít người có thu nhập rất cao sẽ khiến phần lớn dữ liệu được nén vào một vùng tương đối nhỏ của đồ thị. Vì cả hai lý do đó, vẽ biểu đồ phân phối thu nhập theo thang logarit là một ý tưởng tốt.

#### BIỂU ĐỒ THANH

Biểu đồ thanh là một biểu đồ cho dữ liệu rời rạc: nó ghi lại tần suất của mọi giá trị của một biến phân loại. Hình 3.6 cho thấy sự phân bố tình trạng hôn nhân trong bộ dữ liệu khách hàng của bạn. Nếu bạn tin rằng tình trạng hôn nhân giúp dự đoán khả năng sức khỏe bảo hiểm, sau đó bạn muốn kiểm tra xem bạn có đủ khách hàng với các tình trạng hôn nhân khác nhau để giúp bạn khám phá mối quan hệ giữa việc kết hôn (hoặc không) và có bảo hiểm y tế.



Hình 3.6 Biểu đồ thanh hiển thị phân phối của các biến phân loại.

Lệnh ggplot2 để tạo hình 3.6 sử dụng geom\_bar:

```
ggplot(custdata) + geom_bar(aes(x=marital.stat), fill="grey")
```

Biểu đồ này không thực sự hiển thị bất kỳ thông tin nào nhiều hơn tóm tắt(custdata\$marital .stat) sẽ hiển thị, mặc dù một số người thấy biểu đồ dễ tiếp thu hơn văn bản. Biểu đồ thanh hữu ích nhất khi số lượng giá trị có thể có khá lớn, chẳng hạn như trạng thái cư trú. Trong tình huống này, chúng ta thường thấy rằng biểu đồ ngang dễ đọc hơn biểu đồ dọc.

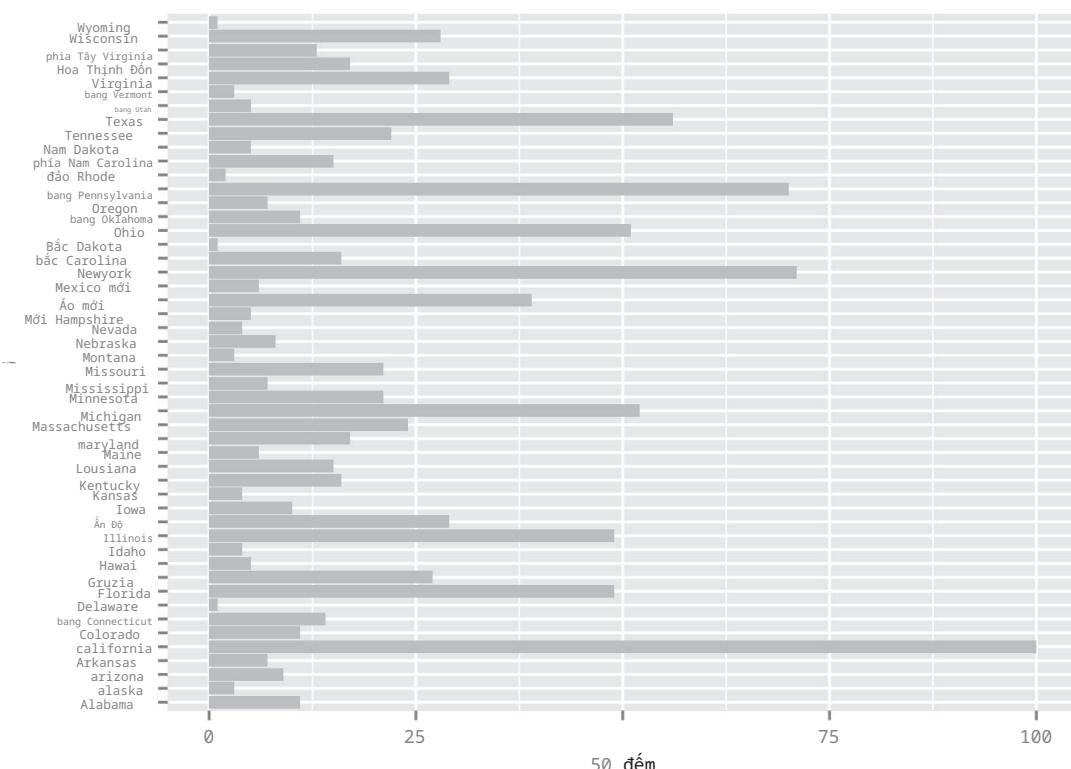
Lệnh ggplot2 để tạo hình 3.7 được hiển thị trong danh sách tiếp theo.

#### Liệt kê 3.9 Tạo biểu đồ thanh ngang

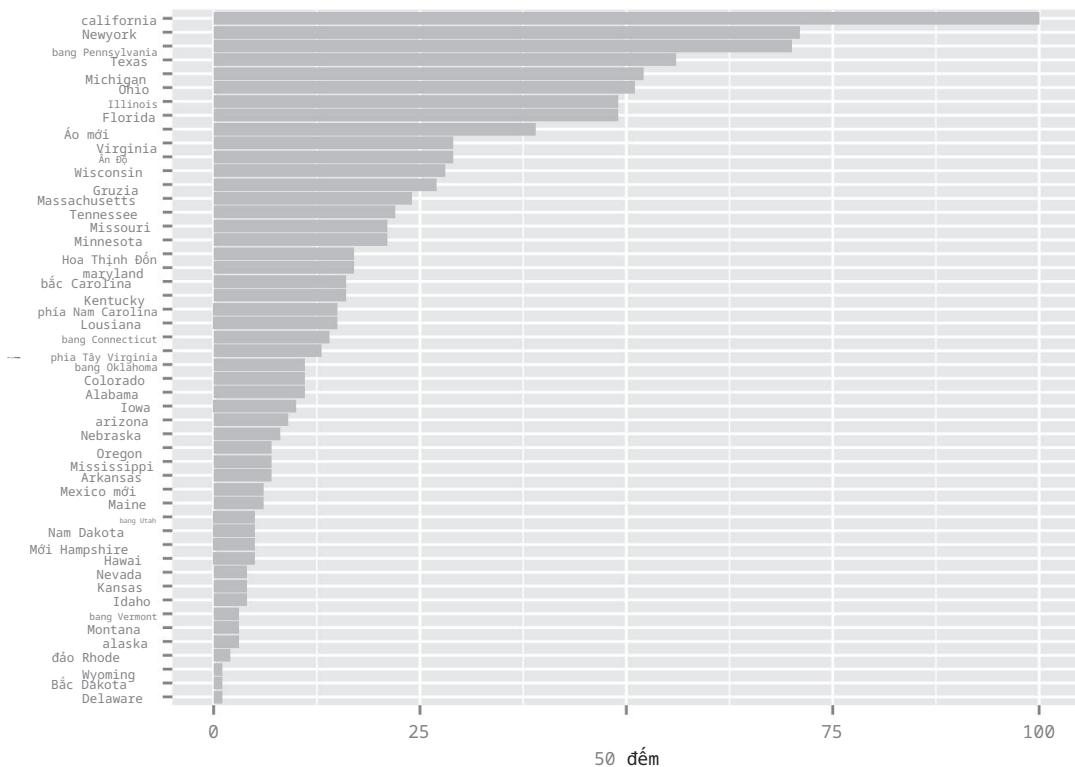
```
Lật x  
và y  
trục:  
state.of.res  
hiện nằm  
trên trục y.  
→  
ggplot(dữ liệu lưu ký) +  
  geom_bar(aes(x=state.of.res), fill="gray") + coord_flip() +  
  theme(axis.text.y=element_text(size=rel(0.8)))
```

Vẽ biểu đồ thanh như trước:  
trạng thái.of.res nằm trên  
trục x, số đếm nằm trên trục y.

Giảm kích thước của nhãn đánh dấu trục y xuống 80% kích thước  
mặc định để dễ đọc.



Hình 3.7 Biểu đồ thanh ngang có thể dễ đọc hơn khi có một số danh mục có tên dài.



Hình 3.8 Việc sắp xếp biểu đồ thanh theo số lượng giúp dễ đọc hơn.

Cleveland<sup>3</sup> khuyên nên sắp xếp dữ liệu trong biểu đồ thanh (hoặc trong biểu đồ chấm, hình ảnh trực quan ưa thích của Cleveland trong trường hợp này) để trích xuất thông tin chuyên sâu hiệu quả hơn từ dữ liệu. Điều này được thể hiện trong hình 3.8.

Hình dung này yêu cầu thao tác nhiều hơn một chút, ít nhất là trong ggplot2, bởi vì mặc định, ggplot2 sẽ vẽ các danh mục của biến nhân tố theo thứ tự bảng chữ cái. ĐẾN thay đổi điều này, chúng tôi phải chỉ định thứ tự của các danh mục theo cách thủ công-trong hệ số biến, không có trong ggplot2.

#### Liệt kê 3.10 Tạo biểu đồ thanh với các danh mục được sắp xếp

```
> stateums <- table(custdata$state.of.res)
đổi tên
cột cho
khả năng đọc.          ↓
> statef <- as.data.frame(stateums)
Chuyển đổi dữ liệu
tổng hợp dữ liệu
theo trạng thái
cú trú- chính xác
thông tin mà
biểu đồ thanh vẽ.

> colnames(statef)<-c("state.of.res", "count")
tóm tắt (trạng thái)          ↓
Lưu ý rằng thứ tự mặc định cho
biến state.of.res được sắp xếp theo thứ tự bảng chữ cái.

> Var1 và Freq.
```

<sup>3</sup> Xem William S. Cleveland, The Elements of Graphing Data, Hobart Press, 1994.

```

trạng thái.of.res      : đêm
Alabama : 1            : tối thiểu : 1,00
alaska      : 1          Qu. 1: 5,00
Arizona : 1             Trung bình : 12,00
Arkansas : 1            Ý nghĩa : 20,00
California: 1           Qu. 3: 26,25
Colorado : 1             tối đa.    : 100,00
(Khác)                 : 44
> statef <- biến đổi (statef,
  state.of.res=reorder(state.of.res, đêm))

> tóm tắt (trạng thái)
trạng thái.of.res      : đêm
Delaware : 1            : tối thiểu : 1,00
Bắc Dakota: 1           Qu. 1: 5,00
Wyoming      : 1          Trung bình : 12,00
Đảo Rhode: 1            Ý nghĩa : 20,00
alaska       : 1          Qu. 3: 26,25
Montana     : 1           tối đa.    : 100,00
(Khác)                 : 44
> ggplot(statef)+ geom_bar(aes(x=state.of.res,y=count),
  thống kê="danh tính",
  tô = "xám") +
  txa độ_flip() +
  chủ đề (axis.text.y=element_text(size=rel(0.8)))

```

Sử dụng hàm `reorder()` để đặt biến `state.of.res` thành thứ tự đêm. Sử dụng hàm `transform()` để áp dụng phép biến đổi cho khung dữ liệu `state.of.res`.

Biến `state.of.res` hiện được đếm theo thứ tự.

Vì dữ liệu đang được chuyển đến `geom_bar` được tổng hợp trước, hãy chỉ định cả hai biến `x` và `y`, đồng thời sử dụng `stat="identity"` để vẽ dữ liệu chính xác như đã cho.

Lật các trục và giảm kích thước của văn bản nhãn như trước.

Trước khi chuyển sang trực quan hóa cho hai biến, trong bảng 3.1, chúng tôi sẽ tóm tắt trực quan hóa mà chúng ta đã thảo luận trong phần này.

Bảng 3.1 Trực quan hóa cho một biến

loại đồ thị	công dụng
Biểu đồ hoặc biểu đồ mật độ	Kiểm tra phạm vi dữ liệu Kiểm tra số lượng chê độ Kiểm tra nếu phân phối là bình thường/lognormal Kiểm tra sự bất thường và ngoại lệ
Biểu đồ cột	So sánh tần số tương đối hoặc tuyệt đối của các giá trị của một biến phân loại

### 3.2.2 Kiểm tra trực quan mối quan hệ giữa hai biến

Ngoài việc kiểm tra các biến riêng lẻ, bạn sẽ thường muốn xem xét mối quan hệ giữa hai biến. Ví dụ: bạn có thể muốn trả lời các câu hỏi như sau:

Có mối quan hệ nào giữa hai yếu tố đầu vào tuổi và thu nhập trong dữ liệu của tôi không?  
 Loại mối quan hệ nào, và mạnh mẽ như thế nào? Có mối  
 quan hệ giữa tình trạng hôn nhân đầu vào và BHYT đầu ra? Khôe như thế nào?

Bạn sẽ định lượng chính xác các mối quan hệ này trong giai đoạn lập mô hình, nhưng việc khám phá chúng bây giờ mang lại cho bạn cảm giác về dữ liệu và giúp bạn xác định biến nào là ứng cử viên tốt nhất để đưa vào mô hình.

Đầu tiên, hãy xem xét mối quan hệ giữa hai biến liên tục. Nhiều nhất cách rõ ràng (mặc dù không phải lúc nào cũng tốt nhất) là biểu đồ đường.

#### VỊ TRÍ DÒNG

Biểu đồ đường hoạt động tốt nhất khi mối quan hệ giữa hai biến tương đối rõ ràng: mỗi giá trị x có một giá trị y duy nhất (hoặc gần như duy nhất), như trong hình 3.9. Bạn vẽ hình 3.9 bằng geom\_line.

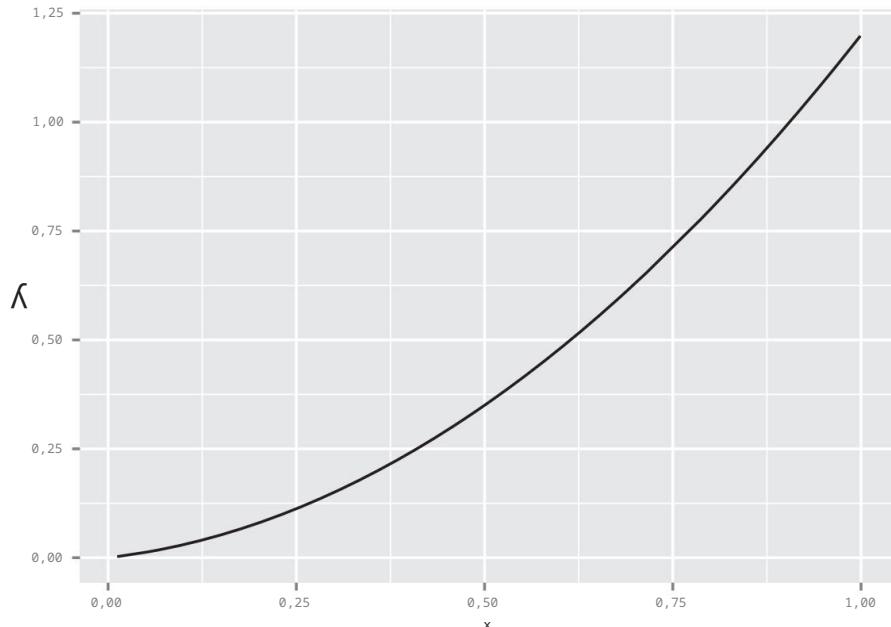
#### Lịch kê 3.11 Tạo biểu đồ đường

```
Vẽ
đường
kích bản. > ggplot(data.frame(x=x,y=y), aes(x=x,y=y)) + geom_line()

x <- runif(100)
y <- x^2 + 0,2*x
```

Đầu tiên, tạo dữ liệu cho ví dụ này. Biến x được phân  
phối ngẫu nhiên đều giữa 0 và 1.

Biến y là một hàm  
bậc hai của x.



Hình 3.9 Ví dụ về biểu đồ đường

Khi dữ liệu không liên quan rõ ràng, các biểu đồ đường không hữu ích; thay vào đó, bạn sẽ muốn sử dụng biểu đồ phân tán, như bạn sẽ thấy trong phần tiếp theo.

### VỊ TRÍ SCATER VÀ LÀM MỊN MÀNG ĐƯỜNG CỘNG

Bạn sẽ mong đợi có một mối quan hệ giữa tuổi tác và bảo hiểm y tế, cũng như mối quan hệ giữa thu nhập và bảo hiểm y tế. Nhưng mối quan hệ giữa tuổi tác và thu nhập là gì? Nếu chúng theo dõi lẫn nhau một cách hoàn hảo, thì bạn có thể không muốn sử dụng cả hai biến trong mô hình bảo hiểm y tế. Thông kê tóm tắt thích hợp là mối tương quan mà chúng tôi tính toán trên một tập hợp con an toàn của dữ liệu của chúng tôi.

#### Liệt kê 3.12 Kiểm tra mối tương quan giữa độ tuổi và thu nhập

```
custdata2 <- tập hợp con(custdata,
  (custdata$age>0 & custdata$age < 100 & custdata$ income > 0))

cor(custdata2$age, custdata2$thu nhập)
[1] -0,02240845
```

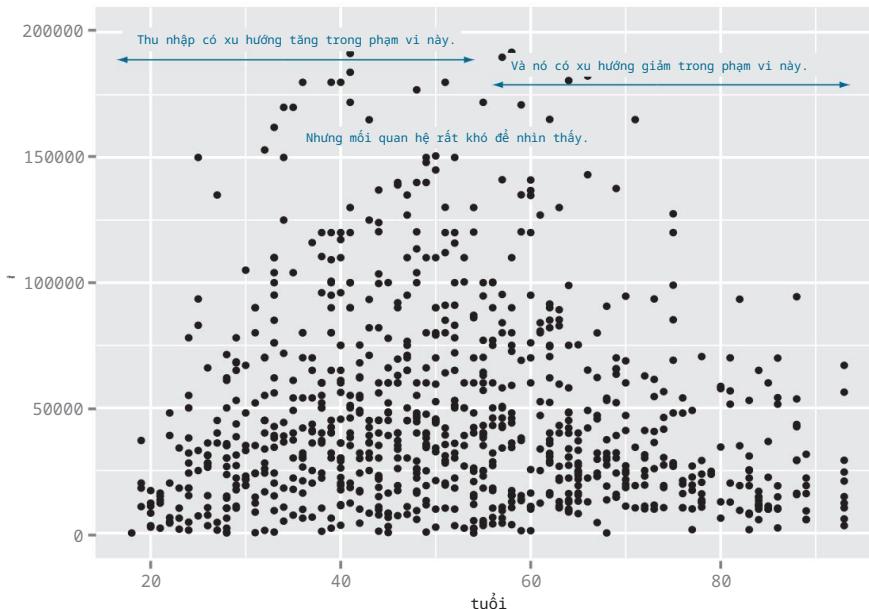
Chỉ xem xét một tập hợp con dữ liệu có giá trị độ tuổi và thu nhập hợp lý.

Lấy tương quan giữa tuổi và thu nhập.

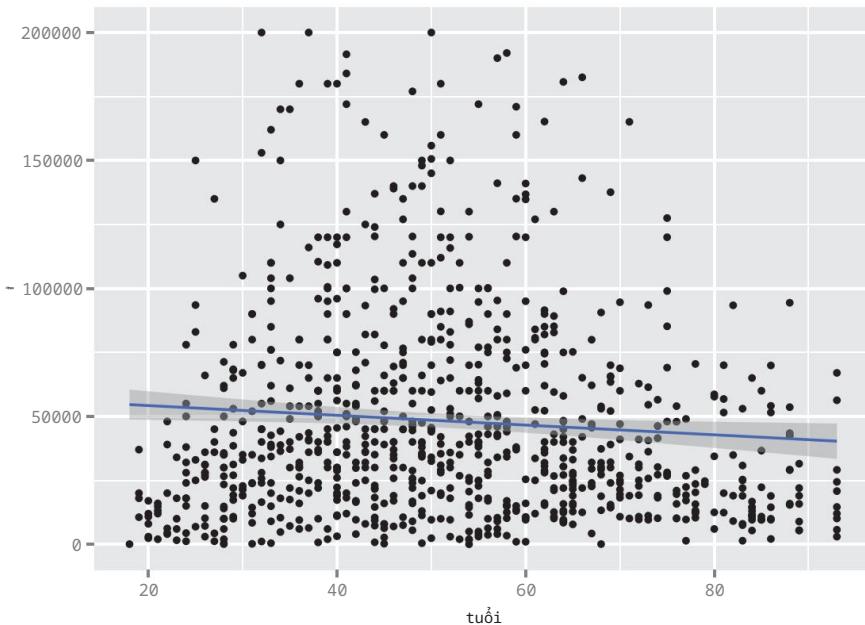
Kết quả tương quan.

Mối tương quan tiêu cực là đáng ngạc nhiên, vì bạn mong đợi rằng thu nhập sẽ tăng lên khi mọi người già đi. Hình ảnh trực quan cung cấp cho bạn thông tin chi tiết hơn về những gì đang diễn ra so với một con số duy nhất có thể. Trước tiên hãy thử một biểu đồ phân tán; bạn vui lòng hình 3.10 với geom\_point:

```
ggplot(custdata2, aes(x=tuổi, y=thu nhập)) + geom_point() +
  ylim(0, 200000)
```



Hình 3.10 Biểu đồ phân tán thu nhập theo độ tuổi



Hình 3.11 Biểu đồ phân tán thu nhập theo độ tuổi, với sự phù hợp tuyến tính

Mỗi quan hệ giữa tuổi tác và thu nhập không dễ nhận thấy. Bạn có thể cố gắng làm cho mỗi quan hệ rõ ràng hơn bằng cách vẽ đồ thị sự phù hợp tuyến tính thông qua dữ liệu, như thể hiện trong hình 3.11.

Bạn vẽ hình 3.11 bằng lớp stat\_smooth :

```
ggplot(custdata2, aes(x=tuổi, y=thu nhập)) + geom_point() + stat_smooth(method="lm") +
  ylim(0, 200000)
```

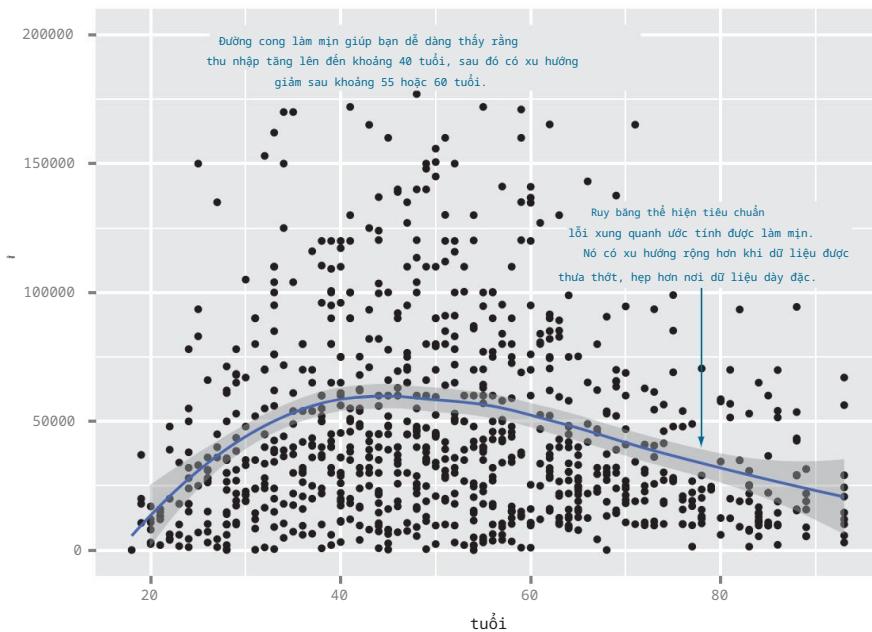
Trong trường hợp này, sự phù hợp tuyến tính không thực sự nắm bắt được hình dạng của dữ liệu. Thay vào đó, bạn có thể nắm bắt hình dạng tốt hơn bằng cách vẽ một đường cong làm mịn thông qua dữ liệu, như thể hiện trong hình 3.12.

Trong R, các đường cong làm mịn được khớp bằng cách sử dụng các **hàm hoàng thổ** (hoặc lowess), tính toán sự khớp tuyến tính cục bộ được làm mịn muộn của dữ liệu. Trong ggplot2, bạn có thể vẽ đường cong làm mịn cho dữ liệu bằng cách sử dụng geom\_smooth:

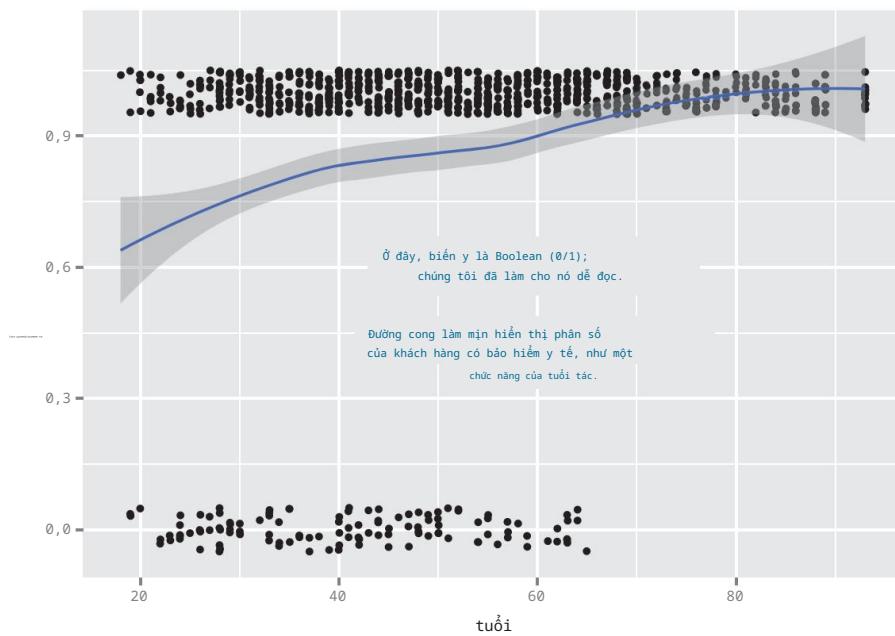
```
ggplot(custdata2, aes(x=tuổi, y=thu nhập)) + geom_point() +
  geom_smooth() + ylim(0, 200000)
```

Biểu đồ phân tán với đường cong làm mịn cũng giúp hình dung tốt về mối quan hệ giữa một biến liên tục và Boolean. Giả sử bạn đang cân nhắc sử dụng tuổi làm thông tin đầu vào cho mô hình bảo hiểm y tế của mình. Bạn có thể muốn lập kế hoạch bảo hiểm y tế

<sup>4</sup> Các lớp thống kê trong ggplot2 là các lớp thực hiện các phép biến đổi trên dữ liệu. Chúng thường được gọi dưới vỏ bọc bởi các lớp geom. Đôi khi bạn phải gọi chúng trực tiếp để truy cập các tham số không thể truy cập được từ các lớp địa lý. Trong trường hợp này, đường cong làm mịn mặc định đã sử dụng geom\_smooth, là đường cong hình hoàng thổ, như bạn sẽ thấy ngay sau đây. Để vẽ đồ thị phù hợp tuyến tính, chúng ta phải gọi trực tiếp stat\_smooth .



Hình 3.12 Biểu đồ phân tán thu nhập theo độ tuổi, với đường cong làm phẳng



Hình 3.13 Phân bố khách hàng có bảo hiểm y tế theo tuổi

phạm vi bảo hiểm như là một chức năng của tuổi tác, như thể hiện trong hình 3.13. Điều này sẽ cho bạn thấy khả năng có bảo hiểm y tế tăng lên khi tuổi của khách hàng tăng lên.  
Bạn vẽ hình 3.13 bằng lệnh được hiển thị trong danh sách tiếp theo.

#### Liệt kê 3.13 Vẽ biểu đồ phân phối của health.ins như một hàm của tuổi

```
Thêm vào  
làm mịn  
đường cong.
```

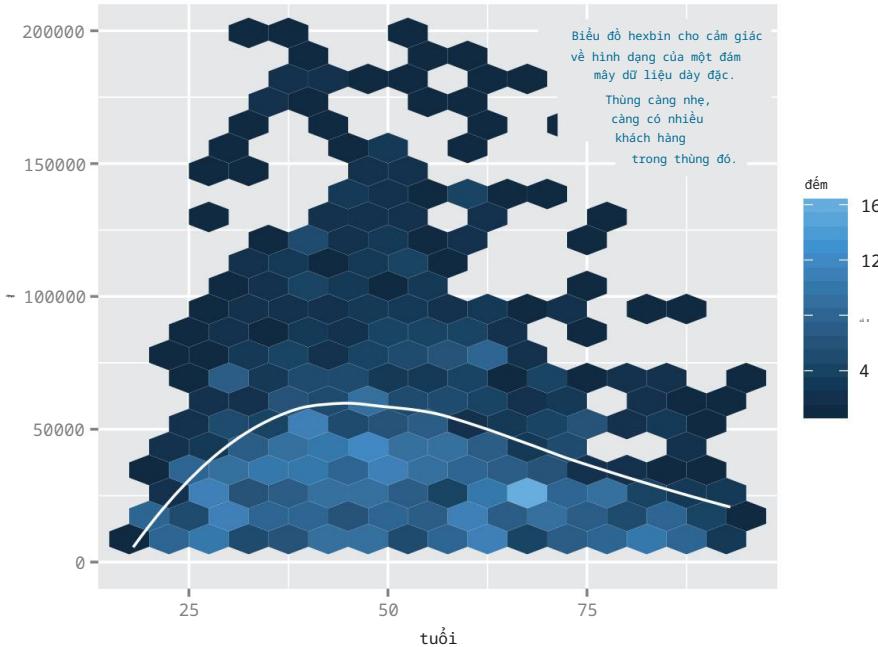
```
ggplot(custdata2, aes(x=tuổi, y=as.numeric(health.ins))) +  
  geom_point(position=position_jitter(w=0.05, h=0.05)) + geom_smooth()  
Vì các giá trị y chỉ có thể là 0 hoặc 1, hãy thêm một  
jitter nhỏ để hiểu được mật độ dữ liệu.
```

Biến Boolean health.ins phải được chuyển đổi thành biến 0/1 bằng cách sử dụng as.numeric.

Trong các ví dụ về bảo hiểm y tế của chúng tôi, tập dữ liệu đủ nhỏ để các biểu đồ phân tán mà bạn đã tạo vẫn có thể đọc được. Nếu tập dữ liệu lớn hơn gấp trăm lần, sẽ có rất nhiều điểm mà chúng sẽ bắt đầu vẽ đồ thị chồng lên nhau; âm mưu phân tán sẽ biến thành một vết bẩn khó đọc. Trong những tình huống có khối lượng lớn như thế này, hãy thử một biểu đồ tổng hợp, chẳng hạn như một biểu đồ hexbin.

#### VỊ TRÍ HEXBIN

Biểu đồ hexbin giống như biểu đồ hai chiều. Dữ liệu được chia thành các ngăn và số lượng điểm dữ liệu trong mỗi ngăn được thể hiện bằng màu hoặc bóng. Hãy quay trở lại ví dụ về thu nhập so với độ tuổi. Hình 3.14 cho thấy biểu đồ dữ liệu hexbin. Lưu ý cách đường cong làm mịn vạch ra hình dạng được tạo bởi vùng dữ liệu dày đặc nhất.



Hình 3.14 Biểu đồ thu nhập theo độ tuổi trong biểu đồ Hexbin, với đường cong làm mịn được tô màu trắng

Để tạo biểu đồ hexbin trong R, bạn phải cài đặt gói hexbin . chúng ta sẽ thảo luận cách cài đặt các gói R trong phụ lục A. Sau khi cài đặt hexbin và thư viện được tải, bạn tạo các ô bằng cách sử dụng lớp geom\_hex .

#### Lý thuyết 3.14 Tạo biểu đồ hexbin

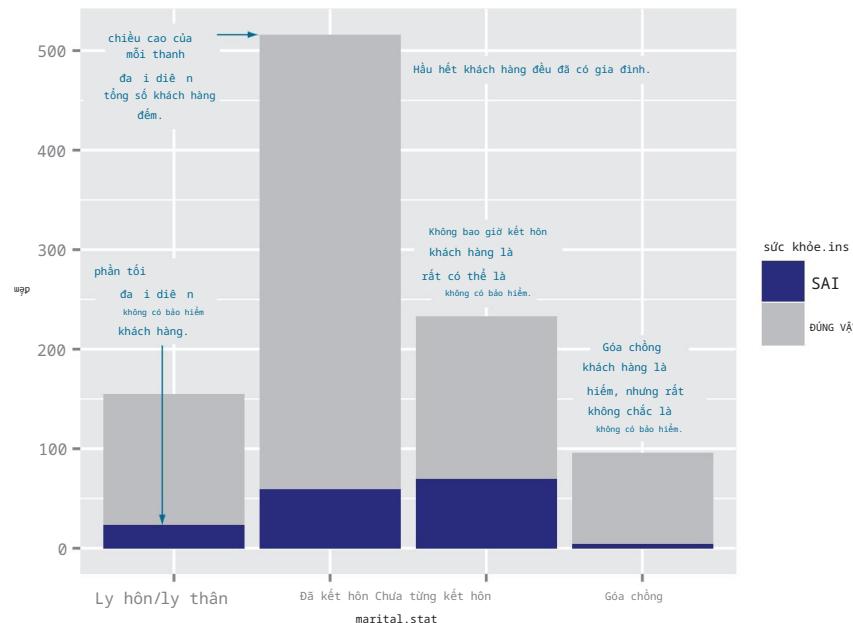
```
thư viện (hexbin)          ← Tải thư viện hexbin.
ggplot(custdata2, aes(x=tuổi, y=thu nhập)) +
  geom_hex(binwidth=c(5, 10000)) +
  geom_smooth(color="white", se=F) +
  ylim(0,200000)           ← Thêm đường cong
                           làm mịn màu
                           trắng;
                           loại bỏ dài băng
                           lõi tiêu chuẩn (se=F).
```

Tạo hexbin với độ tuổi  
được chia thành  
các giá số 5 năm, thu  
nhập theo giá số \$10.000.

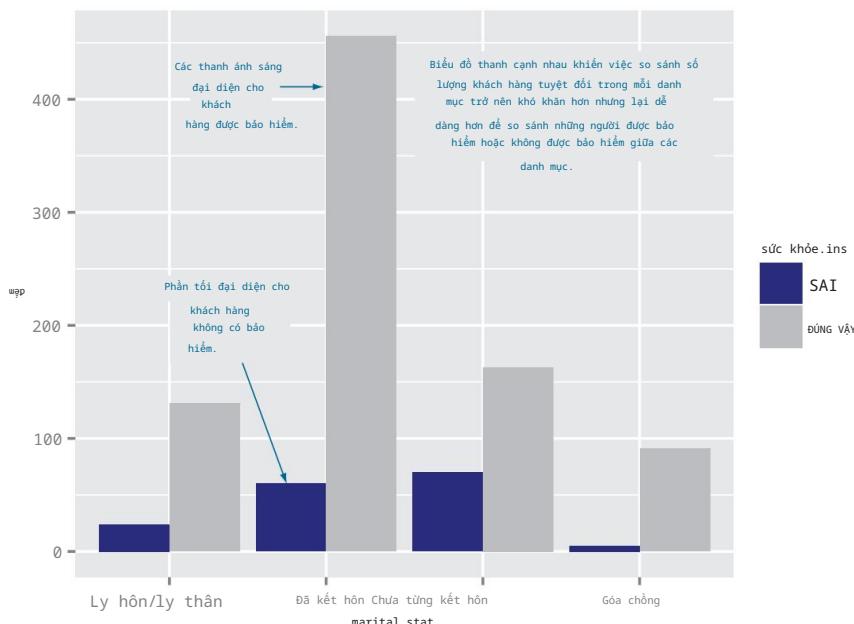
Trong phần này và phần trước, chúng ta đã xem xét các ô có ít nhất một trong các biến là số. Nhưng trong ví dụ về bảo hiểm y tế của chúng ta, đều ra là rõ ràng, và nhiều biến đều vào cùng vậy. Tiếp theo, chúng ta sẽ xem xét các cách để trực quan hóa mối quan hệ giữa hai biến phân loại.

#### BIỂU ĐỒ THANH CHO HAI BIẾN THỂ PHÂN LOẠI

Hãy kiểm tra mối quan hệ giữa tình trạng hôn nhân và xác suất sức khỏe phạm vi bảo hiểm. Cách đơn giản nhất để hình dung điều này là với một thanh xếp chồng lên nhau biểu đồ, như thể hiện trong hình 3.15.



Hình 3.15 Bảo hiểm y tế so với tình trạng hôn nhân: biểu đồ thanh xếp chồng lên nhau



Hình 3.16 Bảo hiểm y tế so với tình trạng hôn nhân: biểu đồ thanh song song

Một số người thích biểu đồ thanh cạnh nhau, thể hiện trong hình 3.16, giúp dễ dàng so sánh số lượng người có bảo hiểm và không có bảo hiểm giữa các danh mục.

Thiếu sót chính của cả biểu đồ thanh xếp chồng lên nhau và cạnh nhau là bạn không thể dễ dàng so sánh tỷ lệ người được bảo hiểm và không được bảo hiểm giữa các danh mục, đặc biệt là đối với các danh mục hiếm như Góa phụ. Bạn có thể sử dụng cái mà ggplot2 gọi là biểu đồ thanh đầy đủ để trực tiếp về biểu đồ trực quan hóa các tỷ lệ, như trong hình 3.17.

Biểu đồ thanh đầy đủ cho thấy rõ ràng những khách hàng đã ly hôn có nhiều khả năng không được bảo hiểm hơn những người đã kết hôn. Nhưng bạn đã đánh mất thông tin rằng góá bụa, mặc dù có tính dự đoán cao về bảo hiểm, là một thể loại hiếm gặp.

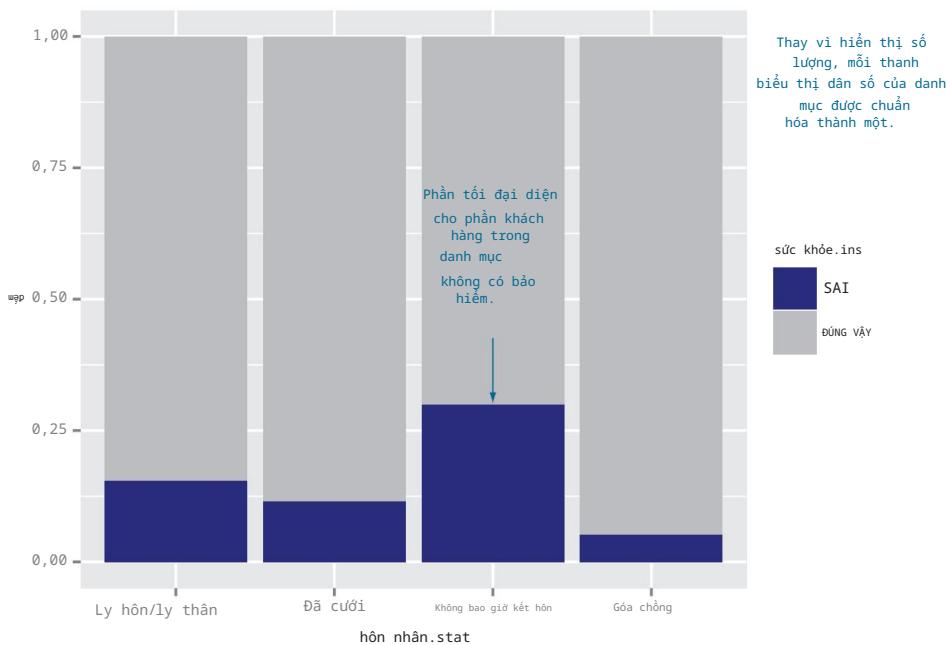
Biểu đồ nào bạn sử dụng tùy thuộc vào thông tin nào là quan trọng nhất để bạn truyền đạt. Các lệnh ggplot2 cho từng ô này được đưa ra tiếp theo. Lưu ý việc sử dụng `thảm mý` điền vào ; điều này yêu cầu ggplot2 tô màu (điền) các thanh theo giá trị của biến `health.ins`. Đổi số vị trí cho `geom_bar` chỉ định kiểu biểu đồ thanh.

#### Liệt kê 3.15 Chi tiết các kiểu khác nhau của biểu đồ thanh

```
ggplot(custdata) + geom_bar(aes(x=marital.stat, fill=health.ins))
                                         ┏━━━━━ Biểu đồ thanh xếp chồng, mặc định

ggplot(custdata) + geom_bar(aes(x=marital.stat, fill=health.ins),
                           position="dodge")
                                         ┏━━━━━ Biểu đồ thanh cạnh nhau

ggplot(custdata) + geom_bar(aes(x=marital.stat, fill=health.ins),
                           position="fill")
                                         ┏━━━━━ Biểu đồ thanh đầy
```



Hình 3.17 Bảo hiểm y tế so với tình trạng hôn nhân: biểu đồ thanh đầy

Để hiểu đồng thời cả dân số trong mỗi danh mục và tỷ lệ người được bảo hiểm so với người không được bảo hiểm, bạn có thể thêm cái được gọi là tẩm thản vào biểu đồ thanh đã diễn. Một tẩm thản là một loạt dấu tích hoặc điểm trên trục x, một dấu tích trên mỗi mốc. Tẩm thản dày đặc ở nơi bạn có nhiều dữ liệu và thưa thớt ở nơi bạn có ít dữ liệu. Điều này được thể hiện trong hình 3.18. Bạn tạo biểu đồ này bằng cách thêm lớp geom\_point vào biểu đồ.

#### Lịch kê 3.16 Vẽ dữ liệu bằng một tẩm thản

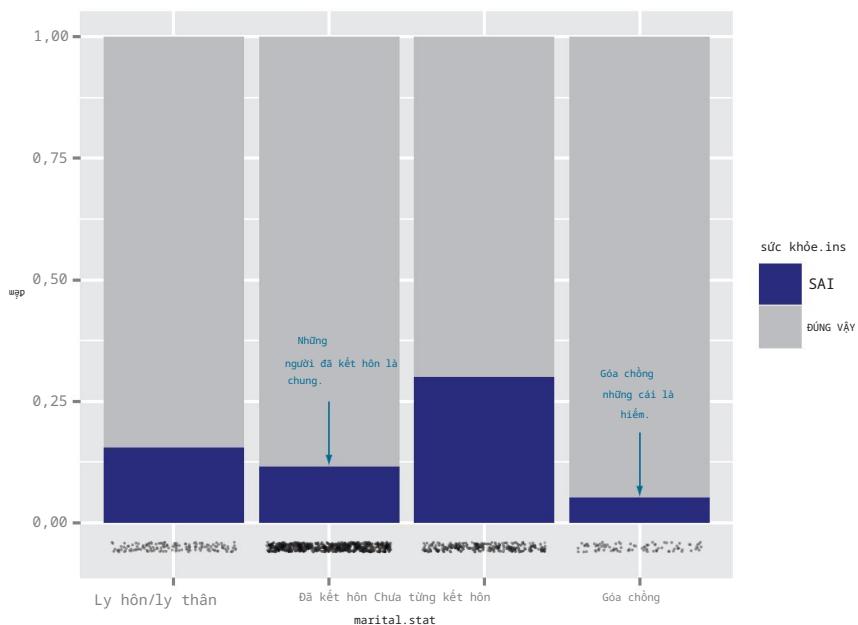
```
ggplot(custdata, aes(x=marital.stat)) +
  geom_bar(aes(fill=health.ins), position="fill") +
  geom_point(aes(y=-0,05), size=0,75, alpha=0,3,
             vi tri=vi tri_jitter(h=0,01))
```

Đặt các điểm ngay dưới trục y, ba phần tư kích thước mặc định và làm cho chúng hơi trong suốt với tham số alpha.

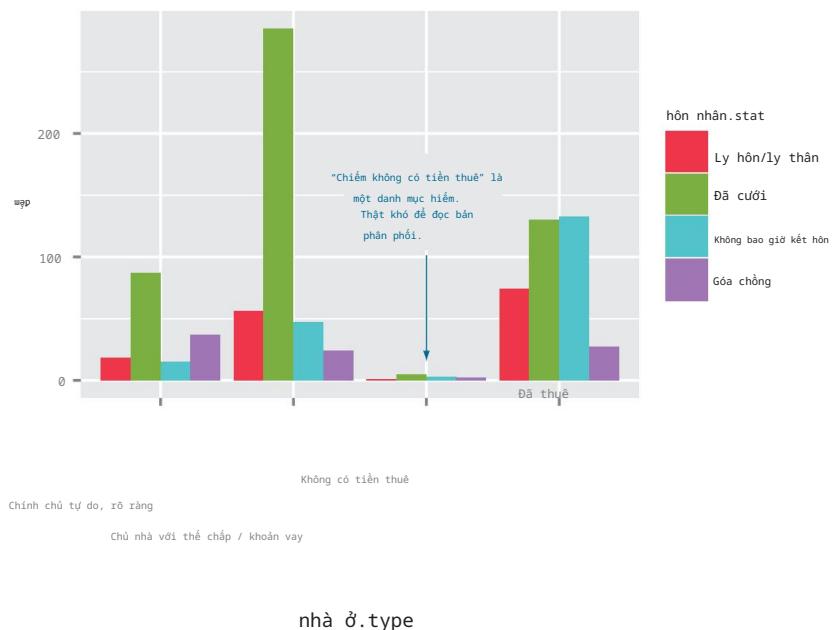
← Jitter các điểm một chút cho dễ đọc.

Trong các ví dụ trước, một trong các biến là nhị phân; các biểu đồ giống nhau có thể được áp dụng cho hai biến mà mỗi biến có một số danh mục, nhưng kết quả khó đọc hơn. Giả sử bạn quan tâm đến sự phân bố tình trạng hôn nhân giữa các loại nhà ở. Một số người thấy biểu đồ thanh cạnh nhau dễ đọc nhất trong tình huống này, nhưng nó không hoàn hảo, như bạn thấy trong hình 3.19.

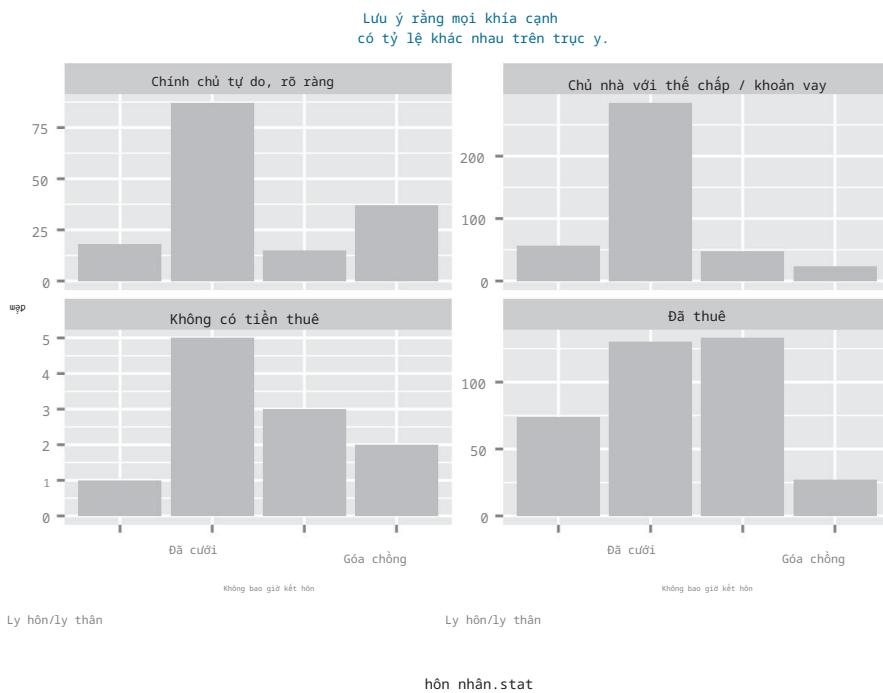
Một biểu đồ như hình 3.19 trở nên lộn xộn nếu một trong hai biến có số lượng danh mục lớn. Một giải pháp thay thế tốt hơn là chia các bản phân phối thành các biểu đồ khác nhau, mỗi biểu đồ cho từng loại nhà ở. Trong ggplot2, thao tác này được gọi là tạo mặt biểu đồ và bạn sử dụng lớp facet\_wrap . Kết quả ở hình 3.20.



Hình 3.18 Bảo hiểm y tế so với tình trạng hôn nhân: biểu đồ thanh đầy những tám thẳm



Hình 3.19 Phân bố tình trạng hôn nhân theo loại nhà ở: biểu đồ thanh cạnh nhau



Hình 3.20 Phân bố tình trạng hôn nhân theo loại hình nhà ở: biểu đồ cột cạnh nhau

Mã cho các hình 3.19 và 3.20 trông giống như danh sách tiếp theo.

**Liệt kê 3.17 Vẽ biểu đồ thanh có và không có các khía cạnh**

```
Biểu đồ thanh cạnh nhau.
    ggplot(custdata2) +
      geom_bar(aes(x=housing.type, fill=marital.stat),
               vị trí = "né") +
      chủ đề(axis.text.x = element_text(angle = 45, hjust = 1))
```

Nghiêng nhãn trục x để chúng không trùng nhau. Bạn cũng có thể sử dụng coord\_flip() để xoay biểu đồ, như chúng ta đã thấy trước đây. Một số thích coord\_flip() hơn vì lớp chủ đề() phức tạp để sử dụng.

```
Các mặt biểu đồ thanh.
    ggplot(custdata2) +
      geom_bar(aes(x=marital.stat), vị trí="né",
               diền = "xám đậm") +
      facet_wrap(~housing.type,scale="free_y") +
      chủ đề(axis.text.x = element_text(angle = 45, hjust = 1))
```

Tạo khía cạnh cho biểu đồ bằng housing.type. Đối số scale="free\_y" chỉ định rằng mỗi khía cạnh có một trục y được chia tỷ lệ độc lập (mặc định là tất cả các khía cạnh có cùng tỷ lệ trên cả hai trục). Đối số free\_x sẽ giải phóng tỷ lệ trục x và đối số free giải phóng cả hai trục.

Khi viết bài này, facet\_wrap không tương thích với coord\_flip, vì vậy chúng tôi phải nghiêng nhãn trục x.

Bảng 3.2 tóm tắt các trực quan hóa cho hai biến mà chúng tôi đã đề cập.

Bảng 3.2 Trực quan hóa cho hai biến

loại đồ thị	công dụng
cột truyền động	Thể hiện mối quan hệ giữa hai biến liên tục. Tốt nhất khi mối quan hệ đó đang hoạt động, hoặc gần như vậy.
âm mưu phân tán	Thể hiện mối quan hệ giữa hai biến liên tục. Tốt nhất khi mối quan hệ quá lỏng lẻo hoặc giống như đám mây để có thể dễ dàng nhìn thấy trên biểu đồ đường.
Làm mịn đường cong	Hiển thị mối quan hệ hoặc xu hướng "trung bình" cơ bản giữa hai biến liên tục. Cũng có thể được sử dụng để hiển thị mối quan hệ giữa biến liên tục và biến nhị phân hoặc biến Boolean: tỷ lệ giá trị thực của biến rời rạc dưới dạng hàm của biến liên tục.
cột truyền hexbin	Cho thấy mối quan hệ giữa hai biến liên tục khi dữ liệu rất dày đặc.
Biểu đồ thanh xếp chồng lên nhau	Hiển thị mối quan hệ giữa hai biến phân loại (var1 và var2). Làm nổi bật tần suất của từng giá trị của var1.
Biểu đồ thanh cạnh nhau	Hiển thị mối quan hệ giữa hai biến phân loại (var1 và var2). Tốt để so sánh tần số của từng giá trị của var2 trên các giá trị của var1. Hoạt động tốt nhất khi var2 là nhị phân.
Biểu đồ thanh đầy	Hiển thị mối quan hệ giữa hai biến phân loại (var1 và var2). Tốt để so sánh tần số tương đối của từng giá trị của var2 trong mỗi giá trị của var1. Hoạt động tốt nhất khi var2 là nhị phân.
Biểu đồ thanh có mặt	Hiển thị mối quan hệ giữa hai biến phân loại (var1 và var2). Tốt nhất để so sánh tần suất tương đối của từng giá trị của var2 trong mỗi giá trị của var1 khi var2 nhận nhiều hơn hai giá trị.

Có nhiều biến thể và hình ảnh trực quan khác mà bạn có thể sử dụng để khám phá dữ liệu; tập hợp trước bao gồm một số đồ thị cơ bản và hữu ích nhất. Bạn nên thử các loại biểu đồ khác nhau để có được những hiểu biết khác nhau từ dữ liệu. Đó là một quá trình tương tác. Một biểu đồ sẽ đưa ra các câu hỏi mà bạn có thể thử trả lời bằng cách vẽ lại dữ liệu một lần nữa, với một hình dung khác nhau.

Cuối cùng, bạn sẽ khám phá dữ liệu của mình đủ để hiểu về nó và phát hiện ra hầu hết vấn đề và vấn đề lớn. Trong chương tiếp theo, chúng ta sẽ thảo luận về một số cách để giải quyết các vấn đề phổ biến mà bạn có thể phát hiện ra trong dữ liệu.

### 3.3 Tóm tắt

Tại thời điểm này, bạn đã cảm nhận được dữ liệu của mình. Bạn đã khám phá nó thông qua tóm tắt và hình dung; bây giờ bạn có ý thức về chất lượng dữ liệu của mình và về mối quan hệ giữa các biến của bạn. Bạn đã nắm bắt và sẵn sàng để sửa một số loại các vấn đề về dữ liệu—mặc dù bạn có thể sẽ gặp phải nhiều vấn đề hơn khi tiến triển.

Có thể một số điều bạn đã khám phá đã khiến bạn phải đánh giá lại câu hỏi mà bạn đang cố gắng trả lời hoặc sửa đổi các mục tiêu của mình. Có lẽ bạn đã quyết định rằng bạn

cần nhiều hơn hoặc các loại dữ liệu khác nhau để đạt được mục tiêu của mình. Đây là tất cả tốt. Quy trình khoa học dữ liệu được tạo thành từ các vòng lặp trong các vòng lặp. Giai đoạn khám phá dữ liệu và làm sạch dữ liệu là hai trong số các giai đoạn tồn tại lâu hơn và cũng là giai đoạn quan trọng nhất của quy trình. Không có dữ liệu tốt, bạn không thể xây dựng các mô hình tốt. Thời gian bạn dành ở đây là thời gian bạn không lãng phí ở nơi khác.

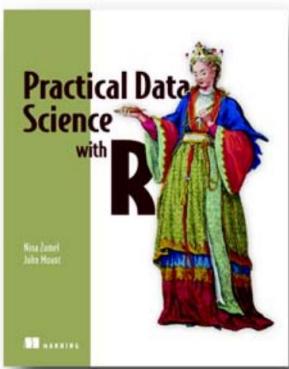
### điểm chính

Dành thời gian để kiểm tra dữ liệu của bạn trước khi đi sâu vào mô hình hóa. Lệnh tóm tắt

giúp bạn phát hiện các vấn đề với phạm vi dữ liệu, đơn vị, kiểu dữ liệu, và các giá trị bị thiếu hoặc không hợp lệ.

Trực quan hóa bổ sung cho bạn cảm giác phân phối dữ liệu và mối quan hệ giữa các biến. Trực quan hóa

là một quá trình lặp đi lặp lại và giúp trả lời các câu hỏi về dữ liệu. Thời gian ở đây là thời gian không bị lãng phí trong quá trình lập mô hình.



Các nhà phân tích và nhà phát triển kinh doanh đang ngày càng thu thập, quản lý, phân tích và báo cáo về dữ liệu kinh doanh quan trọng. Ngôn ngữ R và các công cụ liên quan của nó cung cấp một cách đơn giản để giải quyết các nhiệm vụ khoa học dữ liệu hàng ngày mà không cần nhiều lý thuyết hàn lâm hoặc toán học nâng cao.

[Khoa học dữ liệu thực tế với R](#) chỉ cho bạn cách áp dụng ngôn ngữ lập trình R và các kỹ thuật thống kê hữu ích vào các tình huống kinh doanh hàng ngày. Sử dụng các ví dụ về tiếp thị, kinh doanh thông minh và hỗ trợ quyết định, nó chỉ cho bạn cách thiết kế thử nghiệm (chẳng hạn như thử nghiệm A/B), xây dựng mô hình dự đoán và trình bày kết quả cho khán giả ở mọi cấp độ.

### Có gì bên trong

Khoa học dữ liệu dành cho doanh nghiệp chuyên nghiệp  
Phân tích thống kê bằng ngôn ngữ R  
Vòng đời dự án, từ lập kế hoạch đến phân phối  
Nhiều trường hợp sử dụng quen thuộc ngay lập tức  
Chia khóa để trình bày dữ liệu hiệu quả

Cuốn sách này có thể truy cập được cho độc giả không có nền tảng về khoa học dữ liệu. Một số quen thuộc với số liệu thống kê cơ bản, R hoặc ngôn ngữ kịch bản khác được giả định.

## Chuỗi thời gian

Chuỗi thời gian là cách bạn tổ chức dữ liệu khi thời gian là một yếu tố quan trọng. Ví dụ bao gồm dự báo giá cổ phiếu, lập mô hình mô trường và dự đoán nhu cầu sản phẩm trong tương lai. Trong mô hình dự đoán cổ điển, học một mối quan hệ giữa các đầu vào và kết quả giả định, cả hai đều được lấy mẫu từ cùng một thời gian, là đủ. Ngược lại, đối với chuỗi thời gian, bạn được yêu cầu dự báo một hoặc nhiều số lượng hơn cho một loạt thời gian trong tương lai, chỉ dựa trên các phép đo từ qua khư. Các mô hình chuỗi thời gian có nguy cơ khớp sai cao, vì vậy việc sử dụng các kỹ thuật được ký tự tốt là rất quan trọng. Chương sau minh họa rõ nhất các thành phần phổ biến của dự đoán chuỗi thời gian sử dụng R: làm mịn các xu hướng, ước tính các đường trung bình động, xác định các dao động theo mùa và ước tính các mối quan hệ lũy tiến tự động.

Chương 15 từ R in Action, Phiên bản thứ hai  
của Robert I. Kabacoff

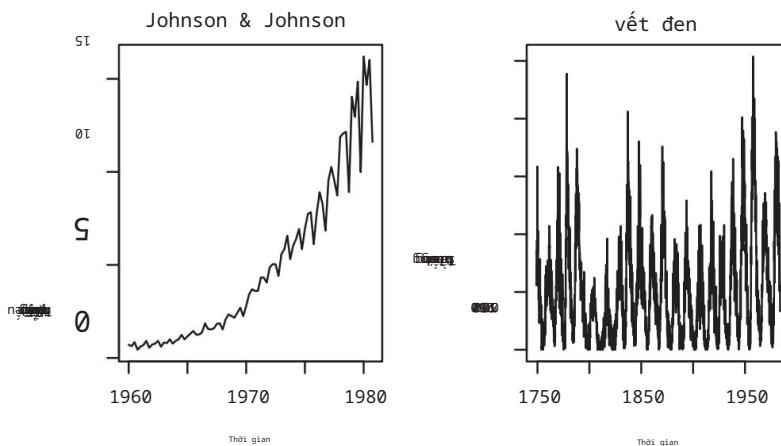
## Chuỗi thời gian

Chương này đề cập đến  
Tạo chuỗi thời gian  
Phân tách chuỗi thời gian thành các thành phần  
Phát triển các mô hình dự đoán  
Dự đoán các giá trị trong tương lai

Sự nóng lên toàn cầu đang diễn ra nhanh như thế nào và tác động sẽ như thế nào sau 10 năm nữa? Ngoại trừ các biện pháp lặp lại ANOVA trong phần 9.6, mỗi chương trước đã tập trung vào dữ liệu cắt ngang. Trong tập dữ liệu chéo, các biến được đo tại một thời điểm duy nhất. Ngược lại, dữ liệu theo chiều dọc liên quan đến việc đo lường các biến lặp đi lặp lại theo thời gian. Bằng cách theo dõi một hiện tượng theo thời gian, bạn có thể học được rất nhiều điều về nó.

Trong chương này, chúng ta sẽ xem xét các quan sát đã được ghi lại ở các khoảng thời gian cách nhau trong một khoảng thời gian nhất định. Chúng ta có thể sắp xếp các quan sát như vậy thành một chuỗi thời gian có dạng  $Y_1, Y_2, Y_3, \dots, Y_t, \dots, Y_T$ , trong đó  $Y_t$  đại diện cho giá trị của  $Y$  tại thời điểm  $t$  và  $T$  là tổng số quan sát trong chuỗi.

Hãy xem xét hai chuỗi thời gian rất khác nhau được hiển thị trong hình 15.1. Chuỗi bên trái chứa thu nhập hàng quý (đô la) trên mỗi cổ phiếu của Johnson & Johnson từ năm 1960 đến năm 1980. Có 84 quan sát: một quan sát cho mỗi quý trên 21



Hình 15.1 Biểu đồ chuỗi thời gian cho (a) thu nhập hàng quý trên mỗi cổ phiếu của Johnson & Johnson (bằng đô la) từ năm 1960 đến năm 1980 và (b) số vết đèn mặt trời tương đối trung bình hàng tháng được ghi lại từ năm 1749 đến năm 1983

năm. Chuỗi bên phải mô tả số lượng vết đèn mặt trời tương đối trung bình hàng tháng từ 1749 đến 1983 được ghi lại bởi Đài thiên văn Liên bang Thụy Sĩ và Đài thiên văn Tokyo Astronomical. Chuỗi thời gian vết đèn mặt trời dài hơn nhiều, với 2.820 lần quan sát-1 lần mỗi tháng trong 235 năm.

Các nghiên cứu về dữ liệu chuỗi thời gian liên quan đến hai câu hỏi cơ bản: điều gì đã xảy ra (mô tả) và điều gì sẽ xảy ra tiếp theo (dự báo)? Đối với Johnson & Johnson dữ liệu, bạn có thể hỏi

Giá cổ phiếu Johnson & Johnson có thay đổi theo thời gian không? Có hiệu ứng hàng quý nào không, với giá cổ phiếu tăng và giảm theo chu kỳ đều đặn trong suốt cả năm? Bạn có thể dự báo giá cổ phiếu trong tương lai sẽ như thế nào và nếu có thì mức độ sự chính xác?

Đối với dữ liệu vết đèn mặt trời, bạn có thể hỏi

Mô hình thống kê nào mô tả đúng nhất hoạt động của vết đèn mặt trời?  
Có phải một số mô hình phù hợp với dữ liệu hơn những mô hình khác không?

Số lượng vết đèn tại một thời điểm nhất định có thể dự đoán được không và nếu có thì ở mức độ nào?

Khả năng dự đoán chính xác giá cổ phiếu có liên quan đến (hy vọng) sớm của tôi  
nghỉ hưu ở một hòn đảo nhiệt đới, trong khi khả năng dự đoán hoạt động của vết đèn mặt trời có liên quan đến việc tiếp nhận điện thoại di động của tôi trên hòn đảo nói trên.

Dự đoán các giá trị tương lai của một chuỗi thời gian, hoặc dự báo, là một kỹ năng cơ bản của con người  
hoạt động và nghiên cứu về dữ liệu chuỗi thời gian có các ứng dụng quan trọng trong thế giới thực. Sáng mù kinh tế  
sử dụng dữ liệu chuỗi thời gian để có gắng hiểu và dự đoán điều gì sẽ xảy ra trong

thị trường tài chính. Các nhà quy hoạch thành phố sử dụng dữ liệu chuỗi thời gian để dự đoán giao thông trong tương lai yêu cầu. Các nhà khoa học khí hậu sử dụng dữ liệu chuỗi thời gian để nghiên cứu biến đổi khí hậu toàn cầu. Các công ty sử dụng chuỗi thời gian để dự đoán nhu cầu sản phẩm và doanh số bán hàng trong tương lai. Các quan chức chăm sóc sức khỏe sử dụng dữ liệu chuỗi thời gian để nghiên cứu sự lây lan của bệnh và dự đoán số lượng các trường hợp trong tương lai trong một khu vực nhất định. Các nhà địa chất học nghiên cứu dữ liệu chuỗi thời gian để dự đoán động đất. Trong mỗi trường hợp, việc nghiên cứu chuỗi thời gian lịch sử là một phần không thể thiếu của quá trình. Bởi vì các cách tiếp cận khác nhau có thể hoạt động tốt nhất với các loại thời gian khác nhau loạt, chúng ta sẽ điều tra nhiều ví dụ trong chương này.

Có nhiều phương pháp để mô tả dữ liệu chuỗi thời gian và dự báo những giá trị trong tương lai. Nếu bạn làm việc với dữ liệu chuỗi thời gian, bạn sẽ thấy rằng R có một số khả năng phân tích toàn diện có sẵn ở bất cứ đâu. Chương này khám phá một số các phương pháp mô tả và dự báo phổ biến nhất và các hàm R được sử dụng để thực hiện chúng. Các chức năng được liệt kê trong bảng 15.1 theo thứ tự xuất hiện trong chương.

Bảng 15.1 Hàm phân tích chuỗi thời gian

Chức năng	Bưu kiện	Sử dụng
ts()	số liệu thống kê	Tạo một đối tượng chuỗi thời gian.
kịch bản()	dữ liệu Vẽ một chuỗi thời gian.	
bắt đầu()	số liệu thống kê	Trả về thời gian bắt đầu của một chuỗi thời gian.
kết thúc()	số liệu thống kê	Trả về thời gian kết thúc của một chuỗi thời gian.
Tính thường xuyên()	số liệu thống kê	Trả về khoảng thời gian của một chuỗi thời gian.
cửa sổ()	số liệu thống kê	Tập hợp con một đối tượng chuỗi thời gian.
ma()	dự báo Phù hợp với mô hình trung bình động đơn giản.	
stl()	số liệu thống kê	Phân tách chuỗi thời gian thành các thành phần theo mùa, xu hướng và bắt thường bằng cách sử dụng hoàng thoả.
biểu đồ tháng()	số liệu thống kê	Biểu đồ các thành phần theo mùa của một chuỗi thời gian.
cốt truyện theo mùa()	dự báo Tạo ra một cốt truyện theo mùa.	
thống kê HoltWinters()		Phù hợp với một mô hình làm mịn theo cấp số nhân.
dự báo()	dự báo Dự báo các giá trị trong tương lai của một chuỗi thời gian.	
sự chính xác()	dự báo Báo cáo phù hợp với các biện pháp cho một mô hình chuỗi thời gian.	
ets()	dự báo Phù hợp với mô hình làm mịn hàm mũ. Bao gồm khả năng tự động hóa việc lựa chọn một mô hình.	
lỗi()	số liệu thống kê	Trả về phiên bản trễ của chuỗi thời gian.
Acf()	dự báo Ước lượng hàm tự tương quan.	
Gói()	dự báo Ước lượng hàm tự tương quan từng phần.	
khác biệt()	căn cứ	Trả về sự khác biệt bị trễ và lặp đi lặp lại.

Bảng 15.1 Hàm phân tích chuỗi thời gian

Chức năng	Bưu kiện	Sử dụng
khác biệt()	dự báo Xác định	mức độ khác biệt cần thiết để loại bỏ các xu hướng trong một chuỗi thời gian.
adf.test()	hàng loạt	Tính toán một phép thử Augmented Dickey-Fuller rằng một chuỗi thời gian là cố định.
arima()	số liệu thống kê	Phù hợp với các mô hình trung bình động tích hợp tự hồi quy.
Hop.test()	số liệu thống kê	Tính toán kiểm định Ljung-Box rằng phần dư của một chuỗi thời gian là độc lập.
bds.test()	hàng loạt	Tính toán kiểm tra BDS mà một chuỗi bao gồm các biến ngẫu nhiên độc lập, được phân phối giống nhau.
auto.arima()	dự báo Tự động hóa việc lựa chọn một mô hình ARIMA.	

Bảng 15.2 liệt kê dữ liệu chuỗi thời gian mà bạn sẽ phân tích. Chúng có sẵn với cơ sở cài đặt R. Các bộ dữ liệu rất khác nhau về đặc điểm và mô hình phù hợp với họ nhất.

Bảng 15.2 Bộ dữ liệu được sử dụng trong chương này

Chuỗi thời gian	Sự miêu tả
hành khách hàng không	Số hành khách hàng tháng của các hãng hàng không từ 1949-1960
JohnsonJohnson Thu nhập hàng quý trên mỗi cổ phiếu của Johnson & Johnson	
nhtemp	Nhiệt độ trung bình hàng năm ở New Haven, Connecticut, từ 1912-1971
sông Nile	Dòng chảy của sông Nile
vết đen	Số vết đen mặt trời hàng tháng từ 1749-1983

Chúng ta sẽ bắt đầu với các phương pháp tạo và thao tác chuỗi thời gian, mô tả và vẽ sơ đồ chúng và phân tách chúng thành cấp độ, xu hướng, theo mùa và bắt thường (lỗi) thành phần. Sau đó, chúng tôi sẽ chuyển sang dự báo, bắt đầu với số mũ phô biến phương pháp mô hình hóa sử dụng trung bình có trọng số của các giá trị chuỗi thời gian để dự đoán những giá trị trong tương lai. Tiếp theo, chúng ta sẽ xem xét một tập hợp các kỹ thuật dự báo được gọi là tự hồi quy các mô hình đường trung bình động tích hợp (ARIMA) sử dụng các mối tương quan giữa các dữ liệu gần đây để tìm hiểu thêm về những chủ đề này.

## 15.1 Tạo đối tượng chuỗi thời gian trong R

Để làm việc với một chuỗi thời gian trong R, bạn phải đặt nó vào một đối tượng chuỗi thời gian-một Cấu trúc R chứa các quan sát, thời gian bắt đầu và kết thúc của chuỗi,

và thông tin về chu kỳ của nó (ví dụ: hàng tháng, hàng quý hoặc hàng năm dữ liệu). Khi dữ liệu nằm trong một đối tượng chuỗi thời gian, bạn có thể sử dụng nhiều chức năng để thao tác, mô hình hóa và vẽ đồ thị dữ liệu.

Một vectơ số hoặc một cột trong khung dữ liệu có thể được lưu dưới dạng chuỗi thời gian đối tượng bằng hàm `ts()`. định dạng là

```
myseries <- ts(dữ liệu, bắt đầu=, kết thúc=, tần suất=)
```

trong đó myseries là đối tượng chuỗi thời gian, dữ liệu là một vectơ số chứa quan sát, bắt đầu chỉ định thời gian bắt đầu chuỗi, kết thúc chỉ định thời gian kết thúc (tùy chọn) và tần suất biểu thị số lượng quan sát trên một đơn vị thời gian (đối với ví dụ, tần suất=1 cho dữ liệu hàng năm, tần suất=12 cho dữ liệu hàng tháng và tần suất = 4 cho dữ liệu hàng quý).

Một ví dụ được đưa ra trong danh sách sau đây. Dữ liệu bao gồm số liệu bán hàng hàng tháng trong hai năm, bắt đầu từ tháng 1 năm 2003.

#### Liệt kê 15.1 Tạo đối tượng chuỗi thời gian

```
> doanh số <- c(18, 33, 41, 7, 34, 35, 24, 25, 24, 21, 25, 20, 22, 31, 40, 29, 25, 21, 22, 54,
```

```
31, 25, 26, 35)
```

```
> tsales <- ts(sales, start=c(2003, 1), frequency=12) > tsales
```

←  
Tạo chuỗi  
thời gian  
đối tượng b

```
Tháng 1 Tháng 2 Tháng 4 Tháng 5 Tháng 6 Tháng 7 Tháng 9 Tháng 10 Tháng 11 Tháng 12  
2003 18 33 41 7 34 35 24 25 24 21 25 20  
2004 22 31 40 29 25 21 22 54 31 25 26 35
```

```
> cốt truyện (tsales)
```

←  
Lấy thông tin  
c về đối tượng

```
> bắt đầu (tsales)  
[1] 2003 1
```

```
> kết thúc (tsales)
```

```
[1] 2004 12
```

```
> tần suất (tsales)
```

```
[1] 12
```

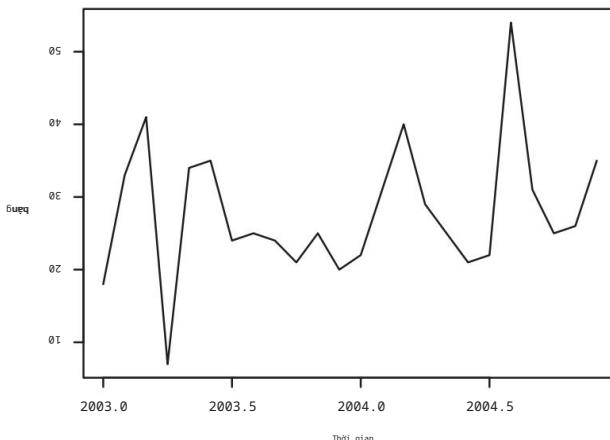
Tập hợp con đối tượng d

```
> tsales.subset <- window(tsales, start=c(2003, 5), end=c(2004, 6)) > tsales.subset
```

←

```
Tháng 1 Tháng 2 Tháng 4 Tháng 5 Tháng 6 Tháng 7 Tháng 9 Tháng 10 Tháng 11 Tháng 12  
2003 34 35 24 25 24 21 25 20  
2004 22 31 40 29 25 21
```

Trong danh sách này, hàm `ts()` được sử dụng để tạo đối tượng chuỗi thời gian b. Một khi nó là được tạo, bạn có thể in và vẽ nó; đồ thị được cho trong hình 15.2. Bạn có thể sửa đổi biểu đồ bằng cách sử dụng các kỹ thuật được mô tả trong chương 3. Ví dụ: `pilot(tsales, type="o", pch=19)` sẽ tạo ra một biểu đồ chuỗi thời gian với các vòng tròn được lấp đầy, được kết nối.



Hình 15.2 Biểu đồ chuỗi thời gian cho dữ liệu bán hàng trong danh sách 15.1. Ký hiệu thập phân về thời gian thứ nguyên được sử dụng để đại diện cho phần của một năm. Ví dụ: 2003.5 đại diện cho ngày 1 tháng 7 (nửa năm 2003).

Khi bạn đã tạo đối tượng chuỗi thời gian, bạn có thể sử dụng các hàm như `start()`, `end()`, và `tần số ()` để trả về các thuộc tính của nó c. Bạn cũng có thể sử dụng hàm `window()` để tạo một chuỗi thời gian mới là tập hợp con của chuỗi thời gian gốc d.

## 15.2 Làm mịn và phân hủy theo mùa

Giống như các nhà phân tích khám phá một tập dữ liệu với các số liệu thống kê và biểu đồ mô tả trước khi cố gắng lập mô hình dữ liệu, việc mô tả một chuỗi thời gian bằng số và trực quan nên là bước đầu tiên trước khi cố gắng xây dựng các mô hình phức tạp. Trong phần này, chúng ta sẽ xem xét làm mịn chuỗi thời gian để làm rõ xu hướng chung của nó và phân tách chuỗi thời gian thành để quan sát bất kỳ hiệu ứng theo mùa.

### 15.2.1 Làm mịn với các đường trung bình động đơn giản

Bước đầu tiên khi điều tra một chuỗi thời gian là vẽ đồ thị của nó, như trong danh sách 15.1. Coi như chuỗi thời gian sông Nile. Nó ghi lại dòng chảy hàng năm của sông Nile tại Ashwan từ năm 1871-Năm 1970. Có thể nhìn thấy cốt truyện của bộ truyện ở bảng phía trên bên trái của hình 15.3. Thời gian sê-ri dường như đang giảm, nhưng có rất nhiều biến thể từ năm này sang năm khác.

Chuỗi thời gian thường có một thành phần bất thường hoặc lỗi đáng kể. Để phân biệt bất kỳ mẫu nào trong dữ liệu, bạn sẽ thường xuyên muốn vẽ một đường cong được làm nhẵn giảm bớt những dao động này. Một trong những phương pháp đơn giản nhất để làm mịn thời gian là sử dụng các đường trung bình động đơn giản. Ví dụ, mỗi điểm dữ liệu có thể được thay thế với giá trị trung bình của quan sát đó và một quan sát trước và sau nó. Đây là

gọi là đường trung bình động định tâm. Một đường trung bình di chuyển trung tâm được định nghĩa là

$$St = (Y_{t-q} + \dots + Y_t + \dots + Y_{t+q}) / (2q + 1)$$

trong đó  $St$  là giá trị được làm mịn tại thời điểm  $t$  và  $k = 2q + 1$  là số lượng quan sát được tính trung bình. Giá trị  $k$  thường được chọn là số lẻ (3 trong ví dụ). Do cần thiết, khi sử dụng đường trung bình động ở giữa, bạn sẽ mất  $(k - 1) / 2$  quan sát ở mỗi đầu của chuỗi.

Một số hàm trong R có thể cung cấp đường trung bình động đơn giản, bao gồm SMA() trong gói TTR , rollmean() trong gói sở thú và ma() trong gói dự báo .

Tại đây, bạn sẽ sử dụng hàm ma() để làm mịn chuỗi thời gian sông Nile đi kèm với cài đặt cơ sở R.

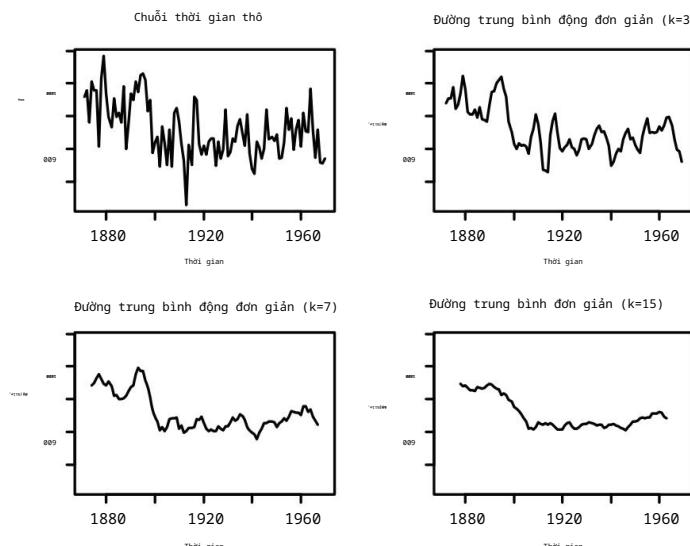
Mã trong danh sách tiếp theo vẽ sơ đồ chuỗi thời gian thô và các phiên bản được làm mịn bằng cách sử dụng k bằng 3, 7 và 15. Đồ thị được cho trong hình 15.3.

#### Lịch kê 15.2 Đường trung bình động đơn giản

```
thư viện (dự báo)
opar <- par(no.readonly=TRUE)
mệnh (mfrow=c(2,2))
ylim <- c(min(Nile), max(Nile))
cốt truyện(Nile, main="Chuỗi thời gian thô")
plot(ma(Nile, 3), main="Trung bình Động Đơn giản (k=3)", ylim=ylim)
plot(ma(Nile, 7), main="Trung bình Động Đơn giản (k=7)", ylim=ylim)
plot(ma(Nile, 15), main="Trung bình Động Đơn giản (k=15)", ylim=ylim)
mệnh (opar)
```

Khi k tăng, cốt truyện ngày càng trở nên mượt mà. Thách thức là tìm ra giá trị của k làm nổi bật các mẫu chính trong dữ liệu mà không làm mịn hoặc làm mịn quá mức. Đây là nghệ thuật hơn là khoa học, và có lẽ bạn sẽ muốn thử một số giá trị của k trước khi chọn một giá trị. Từ các đồ thị trong hình 15.3, chắc chắn xuất hiện là sự suy giảm dòng chảy của sông từ năm 1892 đến năm 1900. Những thay đổi khác được mở ra cho diễn dịch. Ví dụ, có thể có một xu hướng gia tăng nhỏ giữa 1941 và 1961, nhưng đây cũng có thể là một biến thể ngẫu nhiên.

Đối với dữ liệu chuỗi thời gian có chu kỳ lớn hơn một (nghĩa là có chu kỳ theo mùa thành phần), bạn sẽ muốn vượt ra ngoài mô tả về xu hướng tổng thể. theo mùa phân tích có thể được sử dụng để kiểm tra cả xu hướng theo mùa và xu hướng chung.



Hình 15.3 Chuỗi thời gian sông Nile đo dòng chảy hàng năm của sông tại Ashwan từ 1871-1970 (phía trên bên trái). Các biểu đồ khác là các phiên bản được làm trơn bằng cách sử dụng các đường trung bình động đơn giản ở ba mức làm mịn (k=3, 7 và 15).

### 15.2.2 Phân hủy theo mùa

Dữ liệu chuỗi thời gian có khía cạnh theo mùa (chẳng hạn như dữ liệu hàng tháng hoặc hàng quý) có thể được phân tách thành một thành phần xu hướng, một thành phần theo mùa và một thành phần bất thường thành phần. Thành phần xu hướng ghi lại những thay đổi về cấp độ theo thời gian. Thành phần theo mùa nắm bắt các hiệu ứng theo chu kỳ do thời gian trong năm. Thành phần bất thường (hoặc lỗi) nắm bắt những ảnh hưởng không được mô tả bởi xu hướng và hiệu ứng theo mùa.

Sự phân tách có thể là cộng hoặc nhân. Trong một mô hình phụ gia, tổng các thành phần để đưa ra các giá trị của chuỗi thời gian. Đặc biệt,

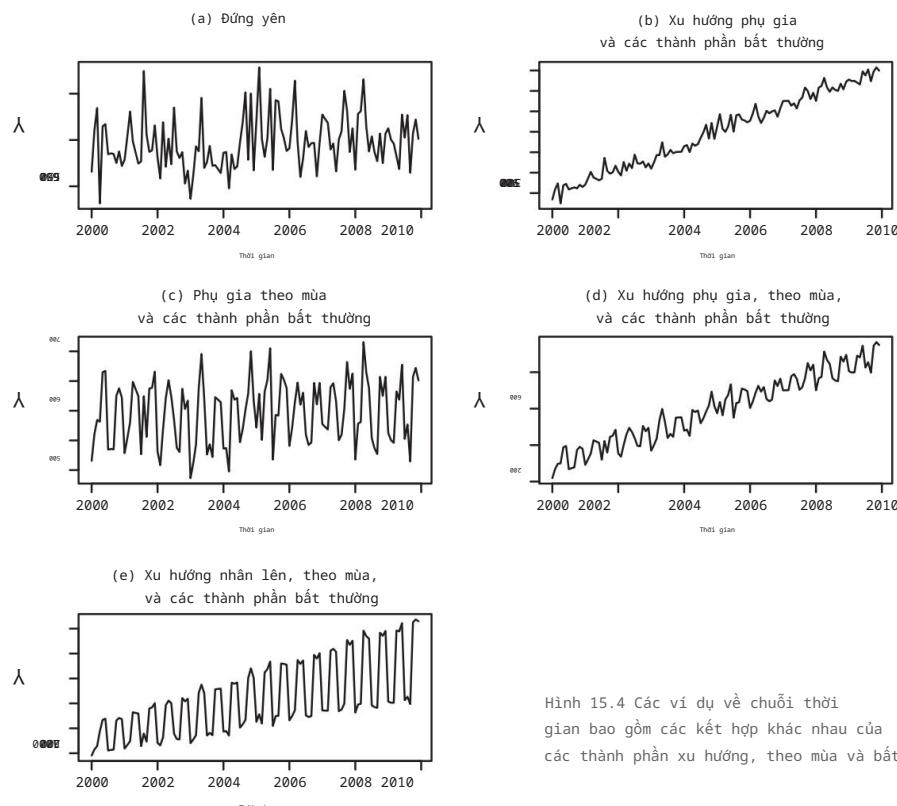
$$Y_t = X_t \text{ xu hướng} + \text{Thời vụ} + \text{Bất thường}$$

trong đó quan sát tại thời điểm  $t$  là tổng các đóng góp của xu hướng tại thời điểm  $t$ , tác động theo mùa tại thời điểm  $t$  và tác động bất thường tại thời điểm  $t$ .

Trong một mô hình nhân, được đưa ra bởi phương trình

$$Y_t = X_t \text{ xu hướng} * \text{Theo mùa} * \text{Không thường xuyên}$$

xu hướng, theo mùa và ảnh hưởng bất thường được nhân lên. Các ví dụ được đưa ra trong hình 15.4.



Hình 15.4 Các ví dụ về chuỗi thời gian bao gồm các kết hợp khác nhau của các thành phần xu hướng, theo mùa và bất thường

Trong biểu đồ đầu tiên (a), không có xu hướng cũng như thành phần theo mùa. Ảnh hưởng duy nhất là sự dao động ngẫu nhiên xung quanh một mức nhất định. Trong ô thứ hai (b), có một xu hướng tăng dần theo thời gian, cũng như những biến động ngẫu nhiên. Trong ô thứ ba (c), có hiệu ứng theo mùa và biến động ngẫu nhiên, nhưng không có xu hướng tổng thể nào khác với chiều ngang đường kẻ. Trong biểu đồ thứ tư (d), cả ba thành phần đều có mặt: xu hướng tăng, hiệu ứng âm thanh trên biển và dao động ngẫu nhiên. Bạn cũng thấy cả ba thành phần trong phần cuối cùng cốt truyện (e), nhưng ở đây chúng kết hợp theo cách nhân lên. Lưu ý cách thay đổi là tỷ lệ thuận với mức độ: khi mức độ tăng lên, độ biến thiên cũng vậy. Sự khuếch đại này (hoặc khả năng giảm chấn) dựa trên mức độ hiện tại của chuỗi gợi ý mạnh mẽ về một mô hình nhân.

Một ví dụ có thể tạo ra sự khác biệt giữa các mô hình cộng và nhân rõ ràng hơn. Hãy xem xét một chuỗi thời gian ghi lại doanh số bán xe máy hàng tháng trong khoảng thời gian 10-khoảng thời gian trong năm. Trong một mô hình có hiệu ứng phụ gia theo mùa, số lượng xe máy bán có xu hướng tăng 500 vào tháng 11 và tháng 12 (do mùa Giáng sinh vội vàng) và giảm 200 vào tháng 1 (khi doanh số có xu hướng giảm). Sự gia tăng theo mùa hoặc giảm không phụ thuộc vào khối lượng bán hàng hiện tại.

Trong một mô hình có hiệu ứng theo mùa nhân lên, doanh số bán xe máy trong tháng 11 và tháng 12 có xu hướng tăng 20% và giảm 10% trong tháng 1. Trong trường hợp nhân lên, tác động của hiệu ứng theo mùa tỷ lệ thuận với khối lượng bán hàng hiện tại.

Đây không phải là trường hợp trong một mô hình phụ gia. Trong nhiều trường hợp, mô hình nhân là thực tế hơn.

Một phương pháp phổ biến để phân tách chuỗi thời gian thành các thành phần xu hướng, theo mùa và bắt thường là phân tách theo mùa bằng cách làm mịn hoàng thổ. Trong R, đây có thể là được thực hiện với hàm `stl()`. định dạng là

```
stl(ts, s.window=, t.window=)
```

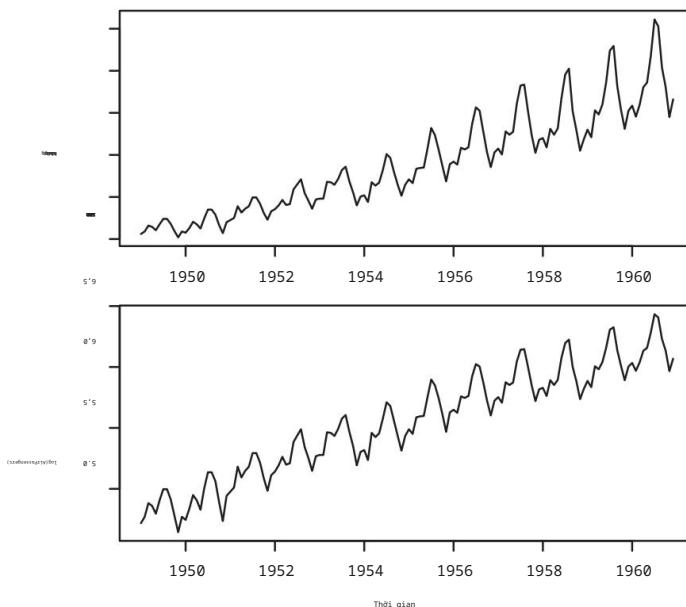
trong đó `ts` là chuỗi thời gian được phân tách, `s.window` kiểm soát tốc độ theo mùa các hiệu ứng có thể thay đổi theo thời gian và `t.window` kiểm soát tốc độ thay đổi của xu hướng tăng cao. Giá trị nhỏ hơn cho phép thay đổi nhanh hơn. Đặt `s.window="periodic"` buộc các hiệu ứng theo mùa giống hệt nhau qua các năm. Chỉ các tham số `ts` và `s.window` là bắt buộc. Xem trợ giúp (`stl`) để biết chi tiết.

Hàm `stl()` chỉ có thể xử lý các mô hình cộng, nhưng đây không phải là hạn chế nghiêm trọng. Các mô hình nhân có thể được chuyển đổi thành các mô hình cộng bằng cách sử dụng `log trans` sự hình thành:

```
log(Yt) = log(Xu hướng * Theo mùa * Không đều)
        = nhật ký (Xu hướng) + nhật ký (Theo mùa) + nhật ký (Không thường xuyên)
```

Sau khi khớp mô hình phụ gia với chuỗi biến đổi nhật ký, kết quả có thể được chuyển đổi trở lại tỷ lệ ban đầu. Hãy xem một ví dụ.

Chuỗi thời gian `AirPassengers` đi kèm với cài đặt cơ sở R và mô tả tổng số hàng tháng (tính bằng nghìn) hành khách của các hãng hàng không quốc tế từ năm 1949 đến Năm 1960. Một biểu đồ dữ liệu được đưa ra ở trên cùng của hình 15.5. Từ biểu đồ, nó xuất hiện độ biến thiên của chuỗi tăng theo cấp độ, gợi ý một mô hình cấp số nhân.



Hình 15.5 Biểu đồ chuỗi thời gian của AirPassengers (trên cùng). Chuỗi thời gian chứa tổng số hàng tháng (tính bằng nghìn) hành khách của hãng hàng không quốc tế từ năm 1949 đến năm 1960. Chuỗi thời gian chuyển đổi nhật ký (phía dưới) ổn định phương sai và phù hợp với phụ gia mô hình phân hủy theo mùa tốt hơn.

Biểu đồ ở phần dưới của hình 15.5 hiển thị chuỗi thời gian được tạo bằng cách lấy nhật ký của mỗi lần quan sát. Phương sai đã ổn định và chuỗi nhật ký trông giống như một ứng cử viên thích hợp cho sự phân hủy phụ gia. Điều này được thực hiện bằng cách sử dụng hàm `stl()` trong danh sách sau.

#### Liệt kê 15.3 Phân tích theo mùa sử dụng `stl()`

```
> cốt truyện(AirPassengers) >
lAirPassengers <- log(AirPassengers)
> cốt truyện(lAirPassengers, ylab="log(AirPassengers)")

> vửa vặn <- stl(lAirPassengers, s.window="period") > cốt truyện(vừa vặn)

> fit$time.series
      ←
      b Vẽ sơ đồ chuỗi thời gian
      ←
      c Phân tích chuỗi thời gian
      ←
      Các thành phần
      cho d mỗi quan sát
      ←
      phần còn lại của xu hướng theo mùa
Tháng 1 năm 1949 -0,09164 4,829 -0,0192494
Tháng 2 năm 1949 -0,11403 4,830 0,0543448
Tháng 3 năm 1949 0,01587 4,831 0,0355884
Tháng 4 năm 1949 -0,01403 4,833 0,0404633
Tháng 5 năm 1949 -0,01502 4,835 -0,0245905
Tháng 6 năm 1949 0,10979 4,838 -0,0426814
Tháng 7 năm 1949 0,21640 4,841 -0,0601152
Tháng 8 năm 1949 0,20961 4,843 -0,0558625
Tháng 9 năm 1949 0,06747 4,846 -0,0008274
Tháng 10 năm 1949 -0,07025 4,851 -0,0015113
Tháng 11 năm 1949 -0,21353 4,856 0,0021631
```

```
Tháng 12 năm 1949 -0,10064 4,865 0,0067347
```

... đầu ra bị bỏ qua ...

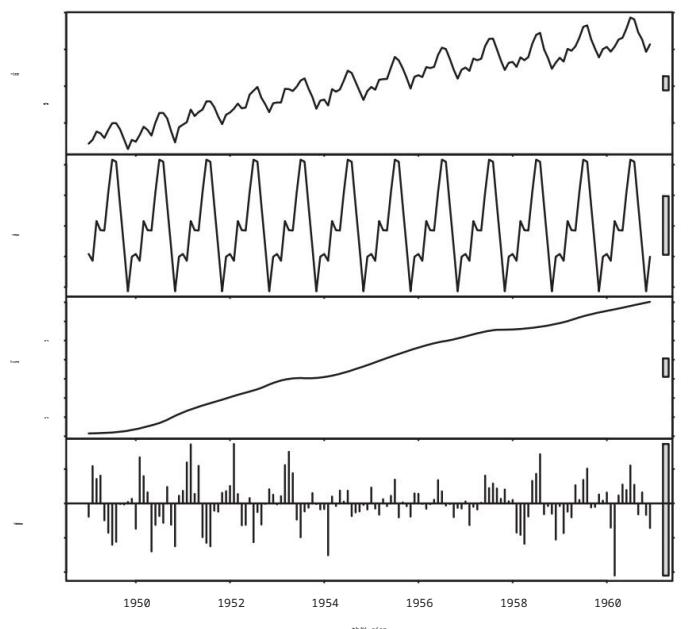
```
> exp(fit$time.series)
```

phần còn lại của xu hướng theo mùa

Tháng 1 năm 1949	0,9124	125,1	0,9809
Tháng 2 năm 1949	0,8922	125,3	1.0558
Tháng 3 năm 1949	1,0160	125,4	1.0362
Tháng 4 năm 1949	0,9861	125,6	1.0413
5 năm 1949	0,9851	125,9	Tháng 6 năm
1949 1,1160	126,2		0,9582
Tháng 7 năm 1949	1,2416	126,6	0,9417
Tháng 8 năm 1949	1,2332	126,9	Tháng 0,9457
9 năm 1949	1,0698	127,2	Tháng 10 năm
1949 0,9322	127,9		0,9992
Tháng 11 năm 1949	0,8077	128,5	0,9985
Tháng 12 năm 1949	0,9043	129,6	1.0022

... đầu ra bị bỏ qua ...

Đầu tiên, chuỗi thời gian được vẽ và biến đổi b. Một sự phân tách theo mùa được hình thành và lưu trong một đối tượng có tên là fit c. Vẽ kết quả cho đồ thị trong hình 15.6. Biểu đồ hiển thị chuỗi thời gian, theo mùa, xu hướng và các thành phần bất thường từ năm 1949 đến năm 1960. Lưu ý rằng các thành phần theo mùa đã bị hạn chế



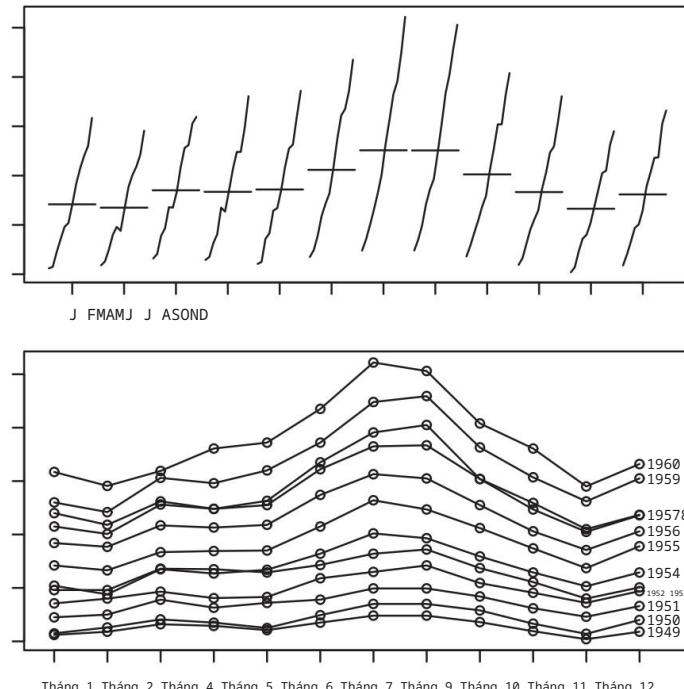
Hình 15.6 Phân tích theo mùa của AirPassengers **đã ghi**  
chuỗi thời gian sử dụng hàm `stl()`. Chuỗi thời gian (dữ liệu) được  
phân tách thành các thành phần theo mùa, xu hướng và bất thường.

giữ nguyên qua mỗi năm (sử dụng `tùy chọn s.window="period"` ). Xu hướng đang tăng lên một cách đơn điệu và hiệu ứng theo mùa cho thấy nhiều hành khách hơn trong mùa hè (có lẽ trong các kỳ nghỉ). Các thanh màu xám bên phải là độ lớn hướng dẫn – mỗi thanh đại diện cho cùng một cường độ. Điều này rất hữu ích vì các trục y là khác nhau đối với mỗi biểu đồ.

Đối tượng được trả về bởi `hàm stl()` chứa một thành phần được gọi là chuỗi thời gian chứa xu hướng, mùa và phần bắt thường của mỗi quan sát d. Trong trường hợp này, `fit$time.series` dựa trên chuỗi thời gian đã ghi. `exp(fit$time.series)` chuyển đổi phân tách trở lại số liệu ban đầu. Kiểm tra các hiệu ứng theo mùa cho thấy rằng số lượng hành khách tăng theo 24% trong tháng 7 (hệ số nhân là 1,24) và giảm 20% trong tháng 11 (với hệ số nhân là 0,80).

Hai biểu đồ bổ sung có thể giúp hình dung sự phân rã theo mùa. Chúng được tạo bởi `hàm monthplot()` đi kèm với cơ sở R và `hàm seasonplot()` được cung cấp trong gói dự báo . Mật mã

```
mệnh (mfrow=c(2,1))
thư viện (dự báo)
monthplot(AirPassengers, xlab="", ylab="")
seasonplot(AirPassengers, year.labels="TRUE", main="")
tạo ra các đồ thị trong hình 15.7.
```



Hình 15.7 Biểu đồ tháng (trên cùng) và biểu đồ mùa (dưới) cho chuỗi thời gian của AirPassengers . Mỗi cái cho thấy một xu hướng ngày càng tăng và mô hình theo mùa tương tự từ năm này sang năm khác.

Biểu đồ tháng (hình trên cùng) hiển thị các chuỗi con cho mỗi tháng (tất cả các giá trị tháng 1 được kết nối, tất cả các giá trị của tháng Hai được kết nối, v.v.), cùng với giá trị trung bình của mỗi dãy con. Từ biểu đồ này, có vẻ như xu hướng đang tăng lên mỗi tháng trong một cách đại khái thống nhất. Ngoài ra, số lượng hành khách lớn nhất diễn ra vào tháng 7 và tháng 8. Cốt truyện theo mùa (hình dưới) hiển thị các chuỗi con theo năm. Một lần nữa bạn thấy một mô hình tương tự, với lượng hành khách tăng lên mỗi năm và cùng một mô hình theo mùa.

Lưu ý rằng mặc dù bạn đã mô tả chuỗi thời gian, nhưng bạn chưa dự đoán bất kỳ những giá trị trong tương lai. Trong phần tiếp theo, chúng ta sẽ xem xét việc sử dụng các mô hình hàm mũ cho dự đoán ngoài dữ liệu có sẵn.

### 15.3 Các mô hình dự báo hàm mũ

Các mô hình hàm mũ là một số cách tiếp cận phổ biến nhất để dự báo giá trị tương lai của một chuỗi thời gian. Chúng đơn giản hơn nhiều loại mô hình khác, nhưng chúng có thể mang lại những dự đoán ngắn hạn tốt trong nhiều ứng dụng. Họ khác nhau nhau trong các thành phần của chuỗi thời gian được mô hình hóa. đơn giản mô hình hàm mũ (còn được gọi là mô hình hàm mũ đơn lẻ) phù hợp với chuỗi thời gian có mức không đổi và thành phần bất thường tại thời điểm i nhưng không có xu hướng cũng như thành phần âm biến. Mô hình hàm mũ kép (còn được gọi là làm mịn hàm mũ Holt) phù hợp với một chuỗi thời gian với cả cấp độ và xu hướng. Cuối cùng, một mô hình mũ ba (cũng được gọi là làm trơn theo cấp số nhân Holt-Winters) phù hợp với một chuỗi thời gian với các thành phần mức độ, xu hướng và âm thanh biến.

Các mô hình hàm mũ có thể phù hợp với hàm HoltWinters() trong cơ sở cài đặt hoặc chức năng ets() đi kèm với gói dự báo . Các et() chức năng có nhiều tùy chọn hơn và thường mạnh hơn. Chúng tôi sẽ tập trung vào ets() chức năng trong phần này.

Định dạng của hàm ets() là

```
ets(ts, model="ZZZ")
```

trong đó ts là một chuỗi thời gian và mô hình được chỉ định bằng ba chữ cái. lá thư đầu tiên biểu thị loại lỗi, chữ cái thứ hai biểu thị loại xu hướng và chữ cái thứ ba biểu thị loại theo mùa. Các chữ cái được phép là A cho phép cộng, M cho phép nhân, N cho không, và Z cho được chọn tự động. Ví dụ về các mô hình phổ biến được đưa ra trong bảng 15.3.

Bảng 15.3 Các hàm điều chỉnh mô hình dự báo hàm mũ đơn giản, gấp đôi và gấp ba

Kiểu	thông số phù hợp	Chức năng
đơn giản	mức độ	ets(ts, model="ANN") ses(ts)
gấp đôi	mức độ, độ dốc	ets(ts, model="AAN") holt(ts)
gấp ba lần	cấp độ, độ dốc, theo mùa (ts, model="AAA") hw(ts)	

Các hàm `ses()`, `holt()` và `hw()` là các hàm bao thuận tiện cho hàm `ets()` với các giá trị mặc định được chỉ định trước.

Đầu tiên chúng ta sẽ xem xét mô hình hàm mũ cơ bản nhất: phép làm tròn hàm mũ đơn giản. Đầu tiên cài đặt gói dự báo (`install.packages("forecast")`) trước khi tiến hành.

### 15.3.1 Làm mịn hàm mũ đơn giản

Làm mịn hàm mũ đơn giản sử dụng trung bình có trọng số của các giá trị chuỗi thời gian hiện có để đưa ra một dự đoán ngắn hạn về các giá trị trong tương lai. Các trọng số được chọn sao cho các quan sát có tác động giảm dần theo cấp số nhân đối với mức trung bình khi bạn quay ngược thời gian.

Mô hình làm mịn hàm mũ đơn giản giả định rằng một quan sát trong thời gian loạt có thể được mô tả bởi

$$Y_t = \text{mức độ} + \text{bất thường}$$

Dự đoán tại thời điểm  $Y_{t+1}$  (được gọi là dự báo trước 1 bước) được viết là

$$Y_{t+1} = c_0 Y_t + c_1 Y_{t-1} + c_2 Y_{t-2} + c_3 Y_{t-3} + \dots$$

trong đó  $c_i = \alpha(1 - \alpha)^i$ ,  $i = 0, 1, 2, \dots$  và  $0 \leq \alpha \leq 1$ . **Tổng trọng số**  $c_i$  bằng một và Dự báo trước 1 bước có thể được coi là trung bình có trọng số của giá trị hiện tại và tất cả giá trị quá khứ của chuỗi thời gian. Tham số alpha ( $\alpha$ ) kiểm soát tốc độ phân rã cho quả cân. Alpha càng gần 1 thì các quan sát gần đây càng có trọng số.

Alpha càng gần 0 thì các quan sát trong quá khứ càng có trọng số. Thực tế giá trị của alpha thường được máy tính chọn để tối ưu hóa tiêu chí phù hợp. Một tiêu chí phù hợp chung là tổng các lỗi bình phương giữa thực tế và dự đoán các giá trị. Một ví dụ sẽ giúp làm rõ những ý tưởng này.

Chuỗi thời gian `nhtemp` chứa nhiệt độ trung bình hàng năm tính bằng độ F ở New Haven, Connecticut, từ năm 1912 đến năm 1971. Biểu đồ của chuỗi thời gian có thể là được xem như đường thẳng trong hình 15.8.

Không có xu hướng rõ ràng và dữ liệu hàng năm thiếu thành phần theo mùa, vì vậy mô hình hàm mũ đơn giản là một nơi hợp lý để bắt đầu. Mã để thực hiện 1 bước dự báo trước bằng cách sử dụng `chức năng ses()` được đưa ra tiếp theo.

#### Liệt kê 15.4 Làm mịn hàm mũ đơn giản

```
> thư viện (dự báo)
> phù hợp <- ets(nhtemp, model="ANN") >
phù hợp
ET(A,N,N)

Gọi:
ets(y = nhtemp, model = "ANN")

Thông số làm mịn:
alpha = 0,182

Trạng thái ban đầu:
l = 50,2759
```

b Phù hợp với mô hình

```

sigma: 1.126

AIC AICc BIC
263,9 264,1 268,1

> dự báo (phù hợp, 1)                                     c Dự báo trước 1 bước
                                                               ↘
                                                               c Dự Báo Điểm Lo 80 Hi 80 Lo 95 Hi 95
1972          51,87 50,43 53,31 49,66 54,08

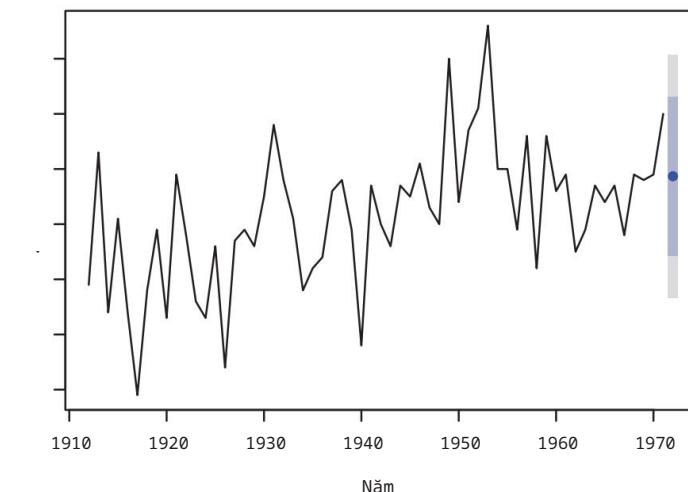
> cốt truyện(dự báo(phù hợp, 1), xlab="Năm",
ylab=expression(paste("Nhiệt độ (", °F,")")), main="Nhiệt độ trung bình hàng năm của
New Haven")

> độ chính xác (phù hợp)                                     d In các thước đo độ chính xác
                                                               ↘
                                                               d Tập huấn luyện RMSE MAE MPE MAPE MASE
                                                               ↑
Tập huấn luyện 0,146 1,126 0,8951 0,2419 1,749 0,9228

Câu lệnh ets(mode="ANN") phù hợp với mô hình hàm mũ đơn giản với thời gian nhtemp
lợi b. A chỉ ra rằng các lỗi là bổ sung và NN chỉ ra rằng có
không có xu hướng và không có thành phần theo mùa. Giá trị tương đối thấp của alpha (0,18) cho biết
rằng các quan sát xa cũng như gần đây đang được xem xét trong dự báo. Cái này
giá trị được chọn tự động để tối đa hóa mức độ phù hợp của mô hình với tập dữ liệu đã cho.

Hàm dự báo () được sử dụng để dự đoán chuỗi thời gian k bước trong tương lai.
Định dạng là dự báo (phù hợp, k). Dự báo trước 1 bước cho loạt bài này là 51,9°F
với khoảng tin cậy 95% (49,7°F đến 54,1°F) c. Chuỗi thời gian, dự báo
giá trị, và khoảng tin cậy 80% và 95% được vẽ trong hình 15.8 d.
```

New Haven Nhiệt độ trung bình hàng năm



Hình 15.8 Nhiệt độ trung bình hàng năm ở New Haven, Connecticut; và dự đoán trước 1 bước
từ dự báo hàm mũ đơn giản bằng cách sử dụng hàm ets()

Gói dự báo cũng cung cấp một hàm `precision()` hiển thị nhiều nhất  
đo lường độ chính xác dự đoán phổ biến cho dự báo chuỗi thời gian d. Một mô tả của  
từng được đưa ra trong bảng 15.4. et đại diện cho lỗi hoặc thành phần bất thường của mỗi  
quan sát ( $Y_t$ ).  $\hat{Y}_t$

Bảng 15.4 Đo lường độ chính xác dự báo

Đo lường	viết tắt	Sự định nghĩa
có nghĩa là lỗi	TOI	có nghĩa là ( và )
Lỗi bình phương trung bình gốc	RMSE	$\text{sqrt}(\text{cô nghĩa là}(\text{v}^2))$
Cô nghĩa là sai số tuyệt đối	MAE	cô nghĩa là (   et   )
Sai số phần trăm trung bình	MPE	trung bình( $100 * et / Y_t$ )
Lỗi phần trăm tuyệt đối trung bình MAPE		cô nghĩa là (   $100 * et / Y_t$   )
Cô nghĩa là lỗi tỷ lệ tuyệt đối	MASE	nghĩa là (   qt   ) trong đó $qt = et / (1/(T-1) * \text{tổng}( yt - yt-1 ))$ , T là số quan sát và tổng đi từ $t=2$ đến $t=t$

Sai số trung bình và sai số phần trăm trung bình có thể không hữu ích, bởi vì giá trị dương  
và lỗi tiêu cực có thể hủy bỏ. RMSE cho căn bậc hai của giá trị trung bình  
sai số bình phương, trong trường hợp này là  $1,13^\circ F$ . Báo cáo lỗi phần trăm tuyệt đối trung bình  
lỗi dưới dạng phần trăm của các giá trị chuỗi thời gian. Nó ít đơn vị hơn và có thể được sử dụng để  
so sánh độ chính xác dự đoán qua chuỗi thời gian. Nhưng nó giả định một thang đo  
với điểm 0 thực (ví dụ: số hành khách mỗi ngày). Vì thang đo Fahr enheit không có số không thực, bạn  
không thể sử dụng nó ở đây. Lỗi tỷ lệ tuyệt đối trung bình là  
phép đo độ chính xác gần đây nhất và được sử dụng để so sánh độ chính xác dự báo giữa  
chuỗi thời gian trên các quy mô khác nhau. Không có thước đo tốt nhất về độ chính xác dự đoán.  
RMSE chắc chắn là được biết đến nhiều nhất và thường được trích dẫn.

Làm mịn theo cấp số nhân đơn giản giả định không có xu hướng hoặc tổng hợp theo mùa  
nents. Phần tiếp theo xem xét các mô hình hàm mũ có thể chứa cả hai.

### 15.3.2 Làm trơn theo hàm mũ Holt và Holt-Winters

Phương pháp làm mịn hàm mũ Holt có thể phù hợp với chuỗi thời gian có tổng thể  
mức độ và một xu hướng (độ dốc). Mô hình cho một quan sát tại thời điểm t là

$$Y_t = \text{cao độ} + \text{độ dốc}*t + \text{bất thường}$$

Tham số làm mịn alpha kiểm soát sự phân rã theo hàm mũ cho cấp độ và tham số beta  
tham số làm mịn kiểm soát sự phân rã theo cấp số nhân cho độ dốc. Một lần nữa, mỗi tham số nằm trong  
khoảng từ 0 đến 1, với các giá trị lớn hơn mang lại nhiều trọng số hơn cho các quan sát gần đây.

Phương pháp làm mịn theo cấp số nhân Holt-Winters có thể được sử dụng để phù hợp với chuỗi thời gian  
có mức độ tổng thể, xu hướng và thành phần theo mùa. Ở đây, mô hình là

$$Y_t = \text{mức} + \text{độ dốc}*t + st + \text{bất thường}$$

trong đó số đại diện cho ảnh hưởng theo mùa tại thời điểm t. Ngoài alpha và beta tham số, tham số làm mịn gamma kiểm soát sự phân rã theo cấp số nhân của thành phần âm thanh biến. Giống như các giá trị khác, nó nằm trong khoảng từ 0 đến 1 và các giá trị lớn hơn cho nhiều quan trọng đối với các quan sát gần đây trong việc tính toán hiệu ứng theo mùa.

Trong phần 15.2, bạn đã phân tách một chuỗi thời gian mô tả tổng số hành tháng (trong nhật ký nghìn) hành khách của các hãng hàng không quốc tế thành các thành phần có xu hướng phụ gia, theo mùa và bất thường. Hãy sử dụng mô hình hàm mũ để dự đoán du lịch trong tương lai. Lại, bạn sẽ sử dụng các giá trị nhật ký để mô hình phụ gia phù hợp với dữ liệu. Đoạn mã sau danh sách áp dụng phương pháp làm trơn theo hàm mũ Holt-Winters để dự đoán năm giá trị tiếp theo của chuỗi thời gian AirPassengers .

#### Liệt kê 15.5 Làm mịn theo cấp số nhân với các thành phần cấp độ, độ dốc và theo mùa

```
> thư viện (dự báo)
> phù hợp <- ets(log(AirPassengers), model="AAA") > phù hợp
```

ET(A,A,A)

Gọi:

```
ets(y = log(AirPassengers), model = "AAA")
```

Các thông số làm mịn: alpha =  
0,8528 beta = 4e-04

b Thông số làm mịn

gamma = 0,0121

Trạng thái ban đầu:

```
tối = 4,8362
b = 0,0097
s=-0,1137 -0,2251 -0,0756 0,0623 0,2079 0,2222
          0,1235 -0,0099 0 0,0203 -0,1203 -0,0925
```

sigma: 0,0367

AIC	AICc	BIC
-204.1	-199.8	-156.5

> độ chính xác (phù hợp)

TỐI	RMSE	MAE	MPE	MAPE	MASE	
Tập huấn luyện	-0,0003695	0,03672	0,02835	-0,007882	0,5206	0,07532

> trước <- dự báo (phù hợp, 5) > trước

c Dự đoán tương lai

Dự Báo Điểm Lo 80 Hi 80 Lo 95 Hi 95					
Tháng 1 năm 1961	6.101	6.054	6.148	6.029	6.173
Tháng 2 năm 1961	6.084	6.022	6.146	5.989	6.179
Tháng 3 năm 1961	6.233	6.159	6.307	6.120	6.346
Tháng 4 năm 1961	6.222	6.138	6.306	6.093	6.350
Tháng 5 năm 1961	6.225	6.131	6.318	6.082	6.367

```
> plot(pred, main="Dự báo cho Du lịch Hàng không", ylab="Nhật
ký(Hành khách Hàng không)", xlab="Thời gian")
```

```
> pred$mean <- exp(pred$mean) > pred$lower <-
exp(pred$lower)
> pred$upper <- exp(pred$upper)
> p <- cbind(pred$mean, pred$lower, pred$upper)
> dimnames(p)[2] <- c("mean", "Lo 80", "Lo 95", "Hi 80", "Hi 95")
> p
```

 Đưa ra dự báo trong quy mô ban đầu

có nghĩa là Lo 80 Lo 95 Hi 80 Hi 95

Tháng 1 năm 1961 446,3 425,8 415,3 467,8 479,6

Tháng 2 năm 1961 438,8 412,5 399,2 466,8 482,3

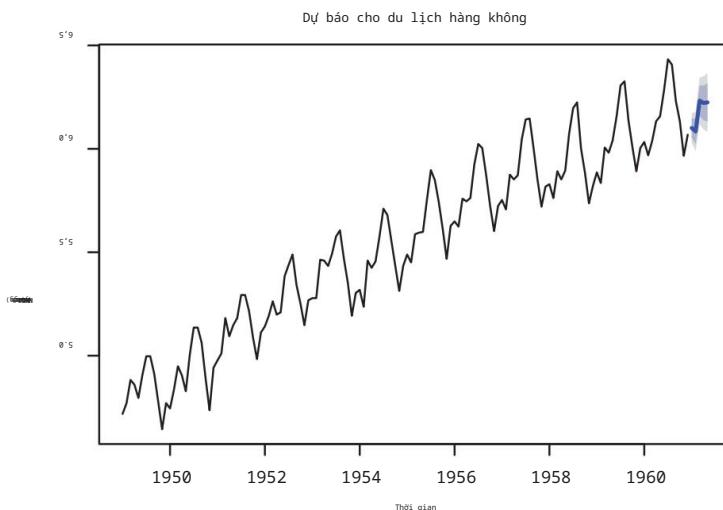
Tháng 3 năm 1961 509,2 473,0 454,9 548,2 570,0

Tháng 4 năm 1961 503,6 463,0 442,9 547,7 572,6

Tháng 5 năm 1961 505,0 460,1 437,9 554,3 582,3

Các tham số làm mịn cho mức (.82), xu hướng (.0004) và các thành phần theo mùa (.012) được đưa ra trong b. Giá trị thấp cho xu hướng (0,0004) không có nghĩa là có không dốc; nó chỉ ra rằng độ dốc ước tính từ những quan sát ban đầu không cần đã được cập nhật.

Hàm dự báo () tạo dự báo cho năm tháng tới c và là vẽ trên hình 15.9. Bởi vì các dự đoán ở thang log, nên lũy thừa là được sử dụng để có được những dự đoán trong số liệu ban đầu: số lượng (tính bằng nghìn) hành khách d. Ma trận pred\$mean chứa các điểm dự đoán và các ma trận pred\$lower và pred\$upper chứa độ tin cậy trên và dưới 80% và 95% giới hạn, tương ứng. Hàm exp() được sử dụng để trả về các dự đoán về thang đo ban đầu và cbind() tạo một bảng duy nhất. Do đó, mô hình dự đoán sẽ có 509.200 hành khách vào tháng 3, với độ tin cậy 95% nằm trong khoảng từ 454.900 đến 570.000.



Hình 15.9 Nhật ký dự báo 5 năm (số lượng hành khách của hãng hàng không quốc tế tính bằng nghìn) dựa trên mô hình làm mịn hàm mũ Holt-Winters. Dữ liệu từ chuỗi thời gian của AirPassengers .

### 15.3.3 Hàm ets() và dự báo tự động

Hàm ets() có các khả năng bổ sung. Bạn có thể sử dụng nó để phù hợp với các mô hình hàm mũ có thành phần nhân, thêm thành phần giảm dần và thực hiện dự báo tự động. Hãy xem xét từng cái một lần lượt.

Trong phần trước, bạn khớp một mô hình mũ phụ với nhật ký của Chuỗi thời gian AirPassengers . Ngoài ra, bạn có thể khớp một mô hình nhân với dữ liệu gốc. Lệnh gọi hàm sẽ là ets(AirPassengers, model="MAM") hoặc tương đương hw(AirPassengers, Season="multiplicative"). Xu hướng vẫn là phụ gia, nhưng các thành phần theo mùa và không đều được giả định là đa bộ. Bằng cách sử dụng mô hình nhân trong trường hợp này, số liệu thống kê về độ chính xác và giá trị dự đoán được báo cáo theo số liệu ban đầu (hàng nghìn hành khách)–a lợi thế quyết định.

Hàm ets() cũng có thể phù hợp với một thành phần giảm xóc. Dự đoán chuỗi thời gian thường cho rằng một xu hướng sẽ tiếp tục đi lên mãi mãi (thị trường nhà đất, có ai không?). MỘT thành phần giảm chấn buộc xu hướng thành tiệm cận ngang trong một khoảng thời gian. Trong nhiều trường hợp, một mô hình giảm chấn đưa ra những dự đoán thực tế hơn.

Cuối cùng, bạn có thể gọi hàm ets() để tự động chọn một giá trị phù hợp nhất mô hình cho dữ liệu. Hãy điều chỉnh mô hình hàm mũ tự động cho dữ liệu của Johnson & Johnson được mô tả trong phần giới thiệu của chương này. Đoạn mã sau cho phép phần mềm để chọn một mô hình phù hợp nhất.

#### Lịch kê 15.6 Dự báo hàm mũ tự động với ets()

```
> thư viện (dự báo)
> phù hợp <- ets(JohnsonJohnson)
> phù hợp

ETS(M,M,M)

Gọi:
et(y = JohnsonJohnson)

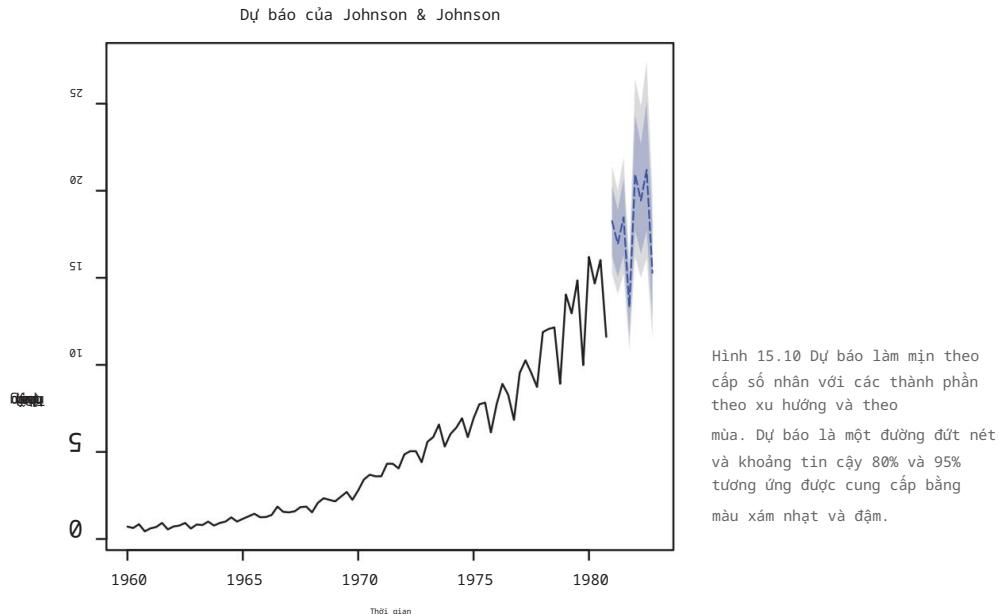
Thông số làm mìn:
alpha = 0,2328 beta
= 0,0367
gamma = 0,5261

Trạng thái ban đầu:
t0i = 0,625
b = 1,0286
s=0,6916 1,2639 0,9724 1,0721

sigma: 0,0863

AIC          AICc          BIC
162.4737 164.3937 181.9203

> cốt truyện(dự báo(phù hợp), main="Dự báo của Johnson & Johnson",
  ylab="Thu nhập hàng quý (Đô la)", xlab="Thời gian", flty=2)
```



Bởi vì không có mô hình nào được chỉ định, phần mềm thực hiện tìm kiếm trên một loạt các các mô hình để tìm một mô hình giảm thiểu tiêu chí phù hợp (khả năng đăng nhập theo mặc định). Các mô hình được chọn là mô hình có các thành phần xu hướng, mùa vụ và lối nhận lên. Cốt truyện, cùng với dự báo cho tám quý tiếp theo (mặc định trong trường hợp này), là cho trên hình 15.10. Tham số  $\alpha$  đặt loại đường cho đường dự báo (nét đứt trong trường hợp này).

Như đã nêu trước đó, mô hình chuỗi thời gian hàm mũ phô biến vì nó có thể cho dự báo ngắn hạn tốt trong nhiều tình huống. Một cách tiếp cận hai cũng phô biến là phương pháp Box-Jenkins, thường được gọi là **mô hình ARIMA**. Đó là được mô tả trong phần tiếp theo.

## 15.4 Mô hình dự báo ARIMA

Trong phương pháp dự báo trung bình di chuyển tích hợp tự hồi quy (ARIMA), các giá trị được dự đoán là một hàm tuyến tính của các giá trị thực tế gần đây và các lối dự đoán gần đây (phản dứ). ARIMA là một cách tiếp cận phức tạp để dự báo. Trong phần này, chúng ta sẽ giới hạn thảo luận về các mô hình ARIMA cho chuỗi thời gian không theo mùa.

Tuy nhiên, khi mô tả các mô hình ARIMA, một số thuật ngữ cần được xác định, bao gồm độ trễ, tự tương quan, tự tương quan một phần, khác biệt và tính dừng. Mỗi là xem xét trong phần tiếp theo.

### 15.4.1 Các khái niệm tiên quyết

Khi bạn làm trễ một chuỗi thời gian, bạn dịch chuyển nó trở lại một số lần quan sát nhất định. Xem xét một vài quan sát đầu tiên từ chuỗi thời gian sông Nile, được trình bày trong bảng 15.5. Trễ 0

là chuỗi thời gian không đổi. Trễ 1 là chuỗi thời gian dịch chuyển sang trái một vị trí. Lỗi 2 dịch chuyển chuỗi thời gian sang trái hai vị trí, v.v. Chuỗi thời gian có thể bị trễ sử dụng hàm lag(ts,k), trong đó ts là chuỗi thời gian và k là số lần trễ.

Bảng 15.5 Chuỗi thời gian sông Nile ở các độ trễ khác nhau

Lỗi	1869	1870	1871	1872	1873	1874	1875 .	
0			1120	1160	963	1210	1160	.
1		1120	1160	963	1210	1160	1160	.
2	1120	1160	963	1210	1160	1160	813	.

Tự tương quan đo lường cách các quan sát trong một chuỗi thời gian liên quan với nhau. ACk là mối tương quan giữa một tập hợp các quan sát ( $Y_t$ ) và các quan sát k giai đoạn trước đó ( $Y_{t-k}$ ). Vậy AC1 là mối tương quan giữa chuỗi thời gian Lag 1 và Lag 0, AC2 là tương quan giữa chuỗi thời gian Lag 2 và Lag 0, v.v. Việc vẽ đồ thị các mối tương quan này (AC1, AC2, .., ACk) sẽ tạo ra một đồ thị hàm tự tương quan (ACF). Âm mưu ACF là được sử dụng để chọn các tham số thích hợp cho mô hình ARIMA và để đánh giá sự phù hợp của mô hình cuối cùng.

Biểu đồ ACF có thể được tạo bằng hàm acf() trong gói thống kê hoặc gói Acf() trong gói dự báo. Ở đây, hàm Acf() được sử dụng vì nó tạo ra một cốt truyện dễ đọc hơn một chút. Định dạng là Acf(ts), trong đó ts là chuỗi thời gian gốc. Biểu đồ ACF cho chuỗi thời gian sông Nile, với  $k=1$  đến 18, được cung cấp một ít lâu sau, ở nửa trên của hình 15.12.

Tự tương quan một phần là mối tương quan giữa  $Y_t$  và  $Y_{t-k}$  với tác động của tất cả  $Y$  các giá trị giữa hai ( $Y_{t-1}, Y_{t-2}, \dots, Y_{t-k+1}$ ) đã bị xóa. Tự tương quan một phần cũng có thể được vẽ cho nhiều giá trị của k. Biểu đồ PACF có thể được tạo bằng pacf() trong gói thống kê hoặc hàm Pacf() trong gói dự báo tuồi. Một lần nữa, hàm Pacf() được ưu tiên hơn do định dạng của nó. Lời gọi hàm là Pacf(ts), trong đó ts là chuỗi thời gian được đánh giá. Biểu đồ PACF cũng được sử dụng để xác định các tham số phù hợp nhất cho mô hình ARIMA. Kết quả cho Chuỗi thời gian sông Nile được đưa ra ở nửa dưới của hình 15.12.

Các mô hình ARIMA được thiết kế để phù hợp với chuỗi thời gian tĩnh (hoặc chuỗi thời gian có thể làm cố định). Trong một chuỗi thời gian cố định, các thuộc tính thống kê của chuỗi không thay đổi theo thời gian. Ví dụ, giá trị trung bình và phương sai của  $Y_t$  là không đổi. Ngoài ra, hệ số tự tương quan đổi với bất kỳ độ trễ k nào không thay đổi theo thời gian.

Có thể cần phải chuyển đổi các giá trị của chuỗi thời gian để đạt được phương sai không đổi trước khi tiếp tục điều chỉnh mô hình ARIMA. Việc chuyển đổi nhật ký là thường hữu ích ở đây, như bạn đã thấy trong phần 15.1.3. Các phép biến đổi khác, chẳng hạn như phép biến đổi Box Cox được mô tả trong phần 8.5.2, cũng có thể hữu ích.

Bởi vì chuỗi thời gian dừng được giả định là có nghĩa là không đổi, nên chúng không thể có một thành phần xu hướng. Nhiều chuỗi thời gian không cố định có thể được tạo thành tĩnh thông qua

sự khác biệt. Trong sự khác biệt, mỗi giá trị của chuỗi thời gian  $Y_t$  được thay thế bằng  $Y_{t-1} - Y_t$ . Phân biệt một chuỗi thời gian một khi loại bỏ một xu hướng tuyến tính. Khác biệt nó lần thứ hai loại bỏ một xu hướng bậc hai. Lần thứ ba loại bỏ một xu hướng khói. Nó hiếm khi cần thiết để chênh lệch hơn hai lần.

Bạn có thể phân biệt chuỗi thời gian bằng hàm `diff()`. Định dạng là khác (`ts, d=1`), trong đó `d` biểu thị số lần chuỗi thời gian `ts` khác nhau. Mặc định là `d=1`. Hàm `ndiffs()` trong gói dự báo có thể là được sử dụng để giúp xác định giá trị tốt nhất của `d`. Định dạng là `ndiffs(ts)`.

Tính dừng thường được đánh giá bằng cách kiểm tra trực quan biểu đồ chuỗi thời gian. Nếu phương sai không phải là hằng số, dữ liệu được chuyển đổi. Nếu có xu hướng, dữ liệu được phân biệt. Bạn cũng có thể sử dụng một quy trình thống kê được gọi là Augmented Dickey-Fuller (ADF) để đánh giá giả định về tính dừng. Trong R, hàm `adf.test()` trong gói `tseries` thực hiện kiểm tra. Định dạng là `adf.test(ts)`, trong đó `ts` là chuỗi thời gian cần đánh giá. Một kết quả quan trọng cho thấy sự ổn định.

Tóm lại, đồ thị ACF và PCF được sử dụng để xác định các tham số của ARIMA người mẫu. Tính dừng là một giả định quan trọng, và các phép biến đổi cũng như sự khác biệt được sử dụng để giúp đạt được tính dừng. Với những khái niệm này trong tay, bây giờ chúng ta có thể chuyển sang các mô hình phù hợp với thành phần tự hồi quy (AR), đường trung bình động (MA) hoặc cả hai thành phần (ARMA). Cuối cùng, chúng ta sẽ kiểm tra các mô hình ARIMA bao gồm các thành phần ARMA và sự khác biệt để đạt được tính ổn định (Tích hợp).

#### 15.4.2 Mô hình ARMA và ARIMA

Trong mô hình tự hồi quy của thứ tự  $p$ , mỗi giá trị trong chuỗi thời gian được dự đoán từ sự kết hợp tuyến tính của các giá trị  $p$  trước đó

$$AR(p): Y_t = \mu + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \dots + \beta_p Y_{t-p} + \varepsilon_t$$

trong đó  $Y_t$  là một giá trị nhất định của chuỗi,  $\mu$  là giá trị trung bình của chuỗi,  $\beta_s$  là trọng số và là thành phần không đều. Trong một mô hình trung bình động của đơn hàng  $q$ , mỗi giá trị trong chuỗi thời gian được dự đoán từ sự kết hợp tuyến tính của  $q$  lỗi trước đó. TRONG trường hợp này

$$MA(q): Y_t = \mu - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \dots - \theta_q \varepsilon_{t-q} + \varepsilon_t$$

ở đây  $\varepsilon_s$  là sai số dự đoán và  $s$  là trọng số. Điều quan trọng là lưu ý rằng các đường trung bình động được mô tả ở đây không phải là các đường trung bình động đơn giản được mô tả trong phần 15.1.2.)

Kết hợp hai cách tiếp cận mang lại mô hình ARMA( $p, q$ ) có dạng

$$\text{có} = \mu + \beta_1 Y_{t-1} + \beta_2 Y_{t-2} + \dots + \beta_p Y_{t-p} - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \dots - \theta_q \varepsilon_{t-q} + \varepsilon_t$$

dự đoán từng giá trị của chuỗi thời gian từ các giá trị  $p$  trong quá khứ và  $q$  phần dư.

Mô hình ARIMA( $p, d, q$ ) là mô hình trong đó chuỗi thời gian đã được phân biệt  $d$  lần và các giá trị kết quả được dự đoán từ  $p$  giá trị thực trước đó và  $q$

các lỗi trước đó. Các dự đoán là "không khác biệt" hoặc được tích hợp để đạt được kết quả cuối cùng sự dự đoán.

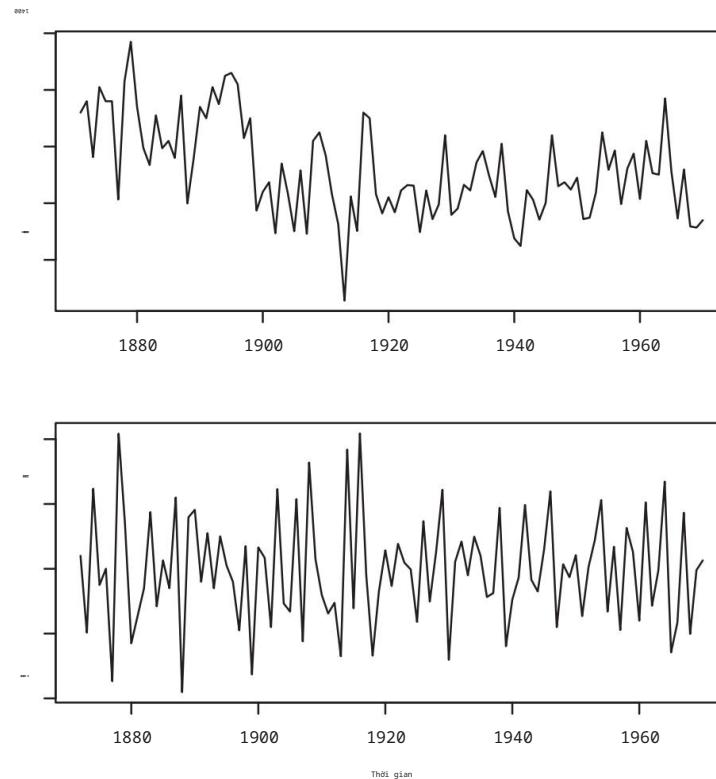
Các bước trong mô hình ARIMA như sau:

- 1 Đảm bảo rằng chuỗi thời gian là cố định.
- 2 Xác định một hoặc nhiều mô hình hợp lý (các giá trị có thể có của p và q).
- 3 Lắp mô hình.
- 4 Đánh giá mức độ phù hợp của mô hình, bao gồm các giả định thống kê và dự báo không phù hợp.
- 5 Đưa ra dự báo.

Hãy lần lượt áp dụng từng bước để khớp mô hình ARIMA với chuỗi thời gian sông Nile .

#### **ĐẢM BẢO RẰNG ĐỒNG THỜI GIAN LÀ VÔ CÙNG**

Trước tiên, bạn vẽ đồ thị chuỗi thời gian và đánh giá tính dừng của nó (xem liệt kê 15.7 và nửa trên của hình 15.11). Phương sai dường như ổn định qua các năm được quan sát, vì vậy không cần phải chuyển đổi. Có thể có một xu hướng, được hỗ trợ bởi kết quả của hàm ndiffs() .



Hình 15.11 Chuỗi thời gian hiển thị dòng chảy hàng năm của sông Nile tại Ashwan từ 1871 đến 1970 (trên cùng) cùng với chuỗi thời gian chênh lệch một lần (dưới cùng).

Sự khác biệt loại bỏ xu hướng giảm rõ ràng trong cốt truyện gốc.

### Liệt kê 15.7 Biến đổi chuỗi thời gian và đánh giá tính dừng

```
> thư viện (dự báo)
> thư viện (tseries)
> cốt truyện (Nile)
> khác biệt (Nile)
```

```
[1] 1
```

```
> dNile <- diff(Nile) > cốt
truyện(dNile)
> adf.test(dNile)
```

Thử nghiệm Dickey-Fuller tăng cường

dữ liệu: dNile  
 Dickey-Fuller = -6,5924, Trễ thứ tự = 4, giá trị p = 0,01  
 giả thuyết thay thế: văn phòng phẩm

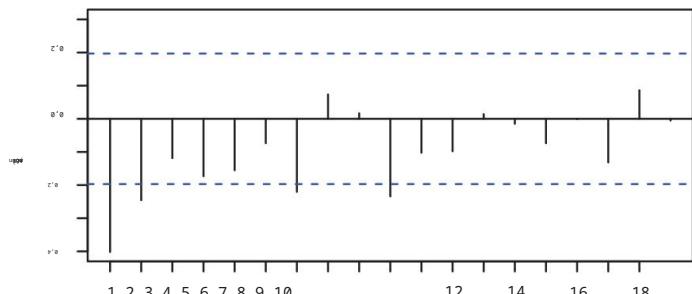
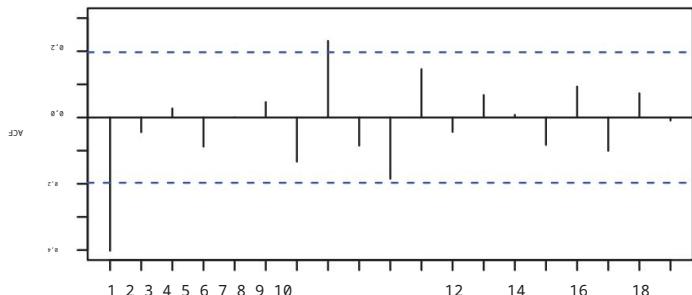
Sê-ri được phân biệt một lần (độ trễ = 1 là mặc định) và được lưu dưới dạng dNile. Chuỗi thời gian khác nhau được vẽ ở nửa dưới của hình 15.11 và chắc chắn là tĩnh hơn. Áp dụng thử nghiệm ADF cho các sê-ri khác nhau cho thấy rằng bây giờ đứng yên, vì vậy bạn có thể tiến hành bước tiếp theo.

#### XÁC ĐỊNH MỘT HOẶC NHIỀU MÔ HÌNH HỢP LÝ

Các mô hình có thể được chọn dựa trên các biểu đồ ACF và PACF :

```
Acf(dNile)
Gói (dNile)
```

Các ô kết quả là  
 cho trên hình 15.12.



Hình 15.12 Tự tương quan  
 và tự tương quan một phần  
 nhằm mục đích sự khác biệt  
 chuỗi thời gian sông Nile

Lỗi

Mục tiêu là xác định các tham số p, d, và q. Bạn đã biết rằng d=1 từ phân trước đó. Bạn nhận được p và q bằng cách so sánh **các đồ thị ACF và PACF** với hướng dẫn trong bảng 15.6.

Bảng 15.6 Hướng dẫn lựa chọn mô hình ARIMA

Người mẫu	ACF	PACF
ARIMA(p, d, 0)	Đường mòn về không	Zero sau lag p
ARIMA(0, d, q)	Không sau độ trễ q	Đường mòn về không
ARIMA(p, d, q)	Đường mòn về không	Đường mòn về không

Kết quả trong bảng 15.6 là lý thuyết, ACF và PACF **thực tế** có thể không khớp chính xác điều này. Nhưng chúng có thể được sử dụng để đưa ra hướng dẫn sơ bộ về các mô hình hợp lý để thử. Đối với chuỗi thời gian sông Nile trong hình 15.12, đường như có một tự tương quan lớn ở độ trễ 1 và tự tương quan một phần giảm dần về 0 khi độ trễ lớn hơn. Cái này gợi ý thử mô hình ARIMA(0, 1, 1).

#### LẮP (CÁC) MÔ HÌNH

Mô hình ARIMA phù hợp với hàm `arima()`. Định dạng là `arima(ts, thứ tự=c(q, d, q))`. Kết quả của việc khớp mô hình ARIMA(0, 1, 1) với thời gian sông Nile loạt được đưa ra trong danh sách sau đây.

#### Liệt kê 15.8 Lắp mô hình ARIMA

```
> thư viện (dự báo)
> phù hợp <- arima(Nile, order=c(0,1,1)) > phù hợp
```

Dòng: sông Nile  
ARIMA(0,1,1)

hệ số:

```
ma1
-0,7329
```

đây là 0,1143

```
sigma^2 ước tính là 20600: khả năng đăng nhập=-632,55
AIC=1269,09 AICc=1269,22 BIC=1274,28
```

```
> độ chính xác (phù hợp)
```

TỔI RMSE MAE	MPE MAPE MASE
Tập huấn luyện -11.94 142.8 112.2 -3.575 12.94 0.8089	

Lưu ý rằng bạn áp dụng mô hình cho chuỗi thời gian ban đầu. Bằng cách chỉ định d=1, nó sẽ tính toán sự khác biệt đầu tiên cho bạn. Hệ số cho các đường trung bình động (-0,73) được cung cấp cùng với AIC. Nếu bạn phù hợp với các mô hình khác, AIC có thể giúp bạn chọn mô hình nào là hợp lý nhất. Giá trị AIC nhỏ hơn gợi ý các mô hình tốt hơn. Sự chính xác

các biện pháp có thể giúp bạn xác định xem mô hình có phù hợp với độ chính xác đầy đủ hay không. Ở đây sai số phần trăm tuyệt đối trung bình là 13% của mực nước sông.

#### ĐÁNH GIÁ SỰ PHÙ HỢP CỦA MÔ HÌNH

Nếu mô hình phù hợp, phần dư phải có phân phối chuẩn với giá trị trung bình bằng 0 và hệ số tự tương quan phải bằng 0 cho mọi độ trễ có thể xảy ra. Nói cách khác, phần dư phải được phân phối bình thường và độc lập (không có mối quan hệ giữa họ). Các giả định có thể được đánh giá với đoạn mã sau.

#### Lịch kê 15.9 Đánh giá sự phù hợp của mô hình

```
> qqnorm(fit$residuals) >
qqline(fit$residuals)
> Box.test(fit$residuals, type="Ljung-Box")
```

Thử nghiệm Box-Ljung

```
dữ liệu: fit$residuals X-squared
= 1,3711, df = 1, giá trị p = 0,2416
```

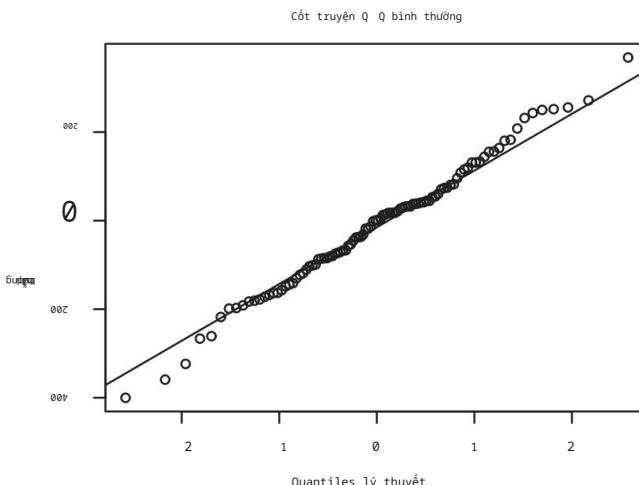
Các hàm `qqnorm()` và `qqline()` tạo ra đồ thị trong hình 15.13. Thông thường, dữ liệu được phân bổ sẽ nằm dọc theo dòng. Trong trường hợp này, kết quả có vẻ tốt.

Hàm `Box.test()` cung cấp một bài kiểm tra rằng các giá trị tự tương quan đều bằng không. Các kết quả không đáng kể, cho thấy rằng tự tương quan không khác 0.

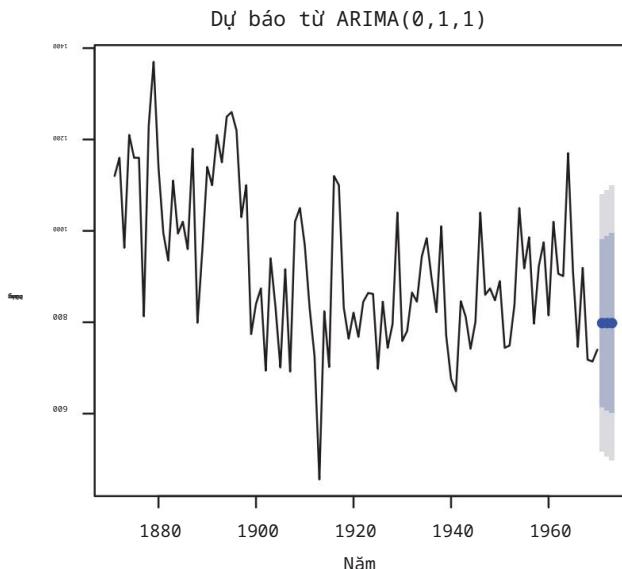
Mô hình ARIMA này có vẻ phù hợp với dữ liệu.

#### DỰ BÁO

Nếu mô hình không đáp ứng các giả định về phần dư thông thường và tự tương quan bằng không, thì cần phải thay đổi mô hình, thêm tham số hoặc thử một cách tiếp cận khác. Khi một mô hình cuối cùng đã được chọn, nó có thể được sử dụng để thực hiện dự đoán giá trị tương lai. Trong danh sách tiếp theo, chức năng `dự báo()` từ gói `dự báo` được sử dụng để dự đoán ba năm tới.



Hình 15.13 Biểu đồ QQ chuẩn để xác định tính chuẩn của phần dư chuỗi thời gian



Hình 15.14 Dự báo ba năm cho chuỗi thời gian sông Nile từ mô hình ARIMA( $0,1,1$ ) phù hợp. Các chấm màu xanh biểu thị các ước tính điểm và các dải màu xám nhạt và đậm tương ứng biểu thị các giới hạn của dải tin cậy 80% và 95%.

#### Liệt kê 15.10 Dự báo với mô hình ARIMA

```
> dự báo (phù hợp, 3)
```

	Dự báo điểm	Lộ 80	Chào 80	Lộ 95	Chào 95
1971	798.3673	614.4307	982.3040	517.0605	1079.674
1972	798.3673	607.9845	988.7502	507.2019	1089.533
1973	798.3673	601.7495	994.9851	497.6663	1099.068

```
> cốt truyện(dự báo(phù hợp, 3), xlab="Năm", ylab="Đòng chảy hàng năm")
```

Hàm `plot()` được sử dụng để vẽ dự báo trong hình 15.14. Ước tính điểm là được cho bởi các chấm màu xanh lam và các dải tin cậy 80% và 95% được thể hiện bằng màu tối và dải ánh sáng, tương ứng.

#### 15.4.3 Dự báo ARIMA tự động

Trong phần 15.2.3, bạn đã sử dụng hàm `ets()` trong gói dự báo để tự động hóa việc lựa chọn một mô hình hàm mũ tốt nhất. Gói này cũng cung cấp một `auto.arima()` để chọn mô hình ARIMA tốt nhất. Danh sách tiếp theo áp dụng điều này cách tiếp cận chuỗi thời gian vết đèn mặt trời được mô tả trong phần giới thiệu của chương.

#### Liệt kê 15.11 Dự báo ARIMA tự động

```
> thư viện (dự báo)
> phù hợp <- auto.arima(vết đèn mặt trời)
> phù hợp
Đóng: vết đèn mặt trời
ARIMA(2,1,2)
```

hệ số:

ar1	ar2	ma1	ma2
1,35	-0,396	-1,77	0,810
se	0,03	0,029	0,02
		0,019	

$\sigma^2$  ước tính là 243: khả năng đăng nhập=-11746  
AIC=23501 AICc=23501 BIC=23531

> dự báo (phù hợp, 3)

	Dự báo điểm	Lộ 80	Chào 80	Lộ 95	Chào 95
Tháng 1 năm 1984	40.437722	20.4412613	60.43418	9.855774	71.01967
Tháng 2 năm 1984	41.352897	18.2795867	64.42621	6.065314	76.64048
Tháng 3 năm 1984	39.796425	15.2537785	64.33907	2.261686	77.33116

> độ chính xác (phù hợp)

TỔI RMSE MAE MPE MAPE MASE

Tập huấn luyện -0,02673 15,6 11,03 NaN Inf 0,32

Hãm chọn một mô hình ARIMA với p=2, d=1 và q=2. Đây là những giá trị mà giảm thiểu tiêu chí AIC trên một số lượng lớn các mô hình có thể. MPE và Độ chính xác của MAPE tăng lên vì không có giá trị nào trong chuỗi (một nhược điểm của hai thống kê này). Vẽ biểu đồ kết quả và đánh giá sự phù hợp được đề lại cho bạn như một bài tập.

## 15.5 Đi xa hơn

Có rất nhiều cuốn sách hay về phân tích và dự báo chuỗi thời gian. Nếu bạn là người mới chủ đề này, tôi đề nghị bắt đầu với cuốn sách Chuỗi thời gian (Đại học Mở, 2006).

Mặc dù nó không bao gồm mã R, nhưng nó cung cấp rất dễ hiểu và trực quan giới thiệu. A Little Book of R for Time Series của Avril Coghlan (<http://mng.bz/8fz0>, 2010) kết hợp tốt với văn bản của Đại học Mở và bao gồm mã R và các ví dụ.

Dự báo: Nguyên tắc và Thực hành (<http://otexts.com/fpp>, 2013) là một cuốn sách giáo khoa trực tuyến rõ ràng và sâu sắc được viết bởi Rob Hyndman và George Athanasopoulos; Nó bao gồm mã R trong suốt. Tôi khuyên bạn nên nó. Ngoài ra, Cowpertwait & Met cale (2009) đã viết một bài viết xuất sắc về phân tích chuỗi thời gian với R. A more điều trị nâng cao cũng bao gồm mã R có thể được tìm thấy trong Shumway & Stoffer (2010).

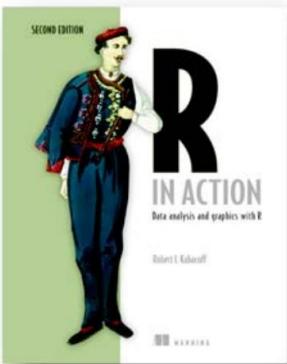
Cuối cùng, bạn có thể tham khảo Ché độ xem tác vụ CRAN trên Phân tích chuỗi thời gian (<http://cran.r-project.org/web/views/TimeSeries.html>). Nó chứa một bản tóm tắt toàn diện về tất cả các khả năng chuỗi thời gian của R.

## 15.6 Tóm tắt

Dự báo có một lịch sử lâu dài và đa dạng, từ những pháp sư đầu tiên dự đoán thời tiết cho các nhà khoa học dữ liệu hiện đại dự đoán kết quả của các cuộc bầu cử gần đây. Dự đoán là niềm vui cơ bản cho cả khoa học và bản chất con người. Trong chương này, chúng ta đã xem xét làm thế nào để tạo chuỗi thời gian trong R, đánh giá xu hướng và kiểm tra các hiệu ứng theo mùa. Sau đó chúng ta

được coi là hai trong số các phương pháp phổ biến nhất để dự báo: mô hình hàm mũ và mô hình ARIMA .

Mặc dù những phương pháp này có thể rất quan trọng trong việc hiểu và dự đoán một nhiều hiện tượng khác nhau, điều quan trọng cần nhớ là mỗi hiện tượng đều đòi hỏi phép ngoại suy-vượt ra ngoài dữ liệu. Họ cho rằng các điều kiện trong tương lai phản ánh hiện tại điều kiện. Dự đoán tài chính được thực hiện trong năm 2007 giả định tăng trưởng kinh tế tiếp tục trong năm 2008 và hơn thế nữa. Như tất cả chúng ta đều biết, đó không phải là cách mọi thứ diễn ra chính xác. Các sự kiện quan trọng có thể thay đổi xu hướng và mô hình trong một chuỗi thời gian và càng xa bạn cố gắng dự đoán, sự không chắc chắn càng lớn.



Các chuyên gia kinh doanh và nhà nghiên cứu phát triển mạnh mẽ dữ liệu và R nói ngôn ngữ của phân tích dữ liệu. R là một ngôn ngữ lập trình mạnh mẽ dành cho tính toán thống kê, không giống công cụ đa năng, R cung cấp hàng nghìn mô-đun để giải quyết mọi thách thức về xử lý dữ liệu hoặc trình bày mà bạn có thể gặp phải. R chạy trên tất cả các nền tảng quan trọng và được sử dụng bởi hàng ngàn các tập đoàn và tổ chức trên toàn thế giới.

**R trong hành động, Phiên bản thứ hai** dạy bạn cách sử dụng ngôn ngữ R bằng cách trình bày các ví dụ liên quan đến các nhà phát triển khoa học, kỹ thuật và kinh doanh. Tập trung vào giải pháp thực tế, cuốn sách cung cấp một khóa học cấp tốc về thống kê, bao gồm các phương pháp tao nhã để xử lý sự lộn xộn

và dữ liệu không đầy đủ. Bạn cũng sẽ thành thạo các khả năng đồ họa mở rộng của R để khám phá và trình bày dữ liệu một cách trực quan. Và phiên bản thứ hai mở rộng này bao gồm mới các chương về dự báo, khai thác dữ liệu và viết báo cáo động.

### Có gì bên trong

Hoàn thành hướng dẫn ngôn ngữ R

Sử dụng R để quản lý, phân tích và trực quan hóa dữ liệu

Kỹ thuật gỡ lỗi chương trình và tạo gói

OOP trong R

Hơn 160 biểu đồ

Cuốn sách này được thiết kế cho những độc giả cần giải quyết các vấn đề phân tích dữ liệu thực tế sử dụng ngôn ngữ R và các công cụ. Một số nền tảng trong toán học và thống kê là hữu ích, nhưng không cần có kinh nghiệm trước về R hoặc lập trình máy tính.

# Học sâu và mạng lưới thần kinh

Mạng lưới thần kinh từ lâu đã là một công cụ học máy được giám sát của lựa chọn khi dữ liệu là số và đại diện cho một tình huống vật lý. Điều này bao gồm làm việc với hình ảnh, âm thanh được ghi lại và các phép đo khoa học. Chương sau giới thiệu các khái niệm cơ bản đằng sau các ứng dụng mạng nơ-ron cổ điển. Các ví dụ được triển khai bằng Python và scikit-learning, cùng với gói gầu trúc, cung cấp một nền tảng lập trình mạnh mẽ cho máy học và khoa học dữ liệu. Chương này kết thúc với phần mô tả về cái gọi là máy Boltzmann bị hạn chế và việc chuyển sang quy trình đào tạo không giám sát dự đoán các phương pháp tiên tiến hiện tại như học sâu và word2vec.

Chương 6 từ Algorithms of the Intelligent Web, Second  
Edition của Douglas G. McIlwraith, Haralambos  
Marmanis, và Dmitry Babenko

# Học sâu và mạng lưới thần kinh

## Chương này bao gồm

Kiến thức cơ bản về mạng nơ-ron

Giới thiệu về học sâu      Nhận dạng  
chữ số bằng máy Boltzmann bị hạn chế

Hiện tại có nhiều cuộc thảo luận về deep learning và nó được coi là bước tiến lớn tiếp theo trong machine learning và trí tuệ nhân tạo. Trong chương này, chúng tôi muốn cắt bỏ phần hùng biện để cung cấp cho bạn sự thật. Ở cuối chương này, bạn nên hiểu khái niệm cơ bản của bất kỳ mạng học sâu nào, perceptron và hiểu cách chúng khớp với nhau trong một mạng sâu. Các mạng lưới thần kinh báo trước sự ra đời của perceptron, vì vậy chúng ta sẽ thảo luận về những vấn đề này trước khi khám phá các mạng sâu hơn, biểu cảm hơn. Những công việc mạng sâu này đi kèm với những thách thức đáng kể trong việc trình bày và đào tạo, vì vậy chúng tôi cần đảm bảo kiến thức nền tảng tốt trước khi bắt đầu.

Trước khi thực hiện tất cả những điều này, chúng ta sẽ thảo luận về bản chất của deep learning và các loại vấn đề mà deep learning đã được áp dụng và điều gì làm nên thành công của chúng. Điều này sẽ cung cấp cho bạn một động lực cơ bản để học sâu và một khung để treo một số khái niệm lý thuyết phức tạp hơn sau này trong

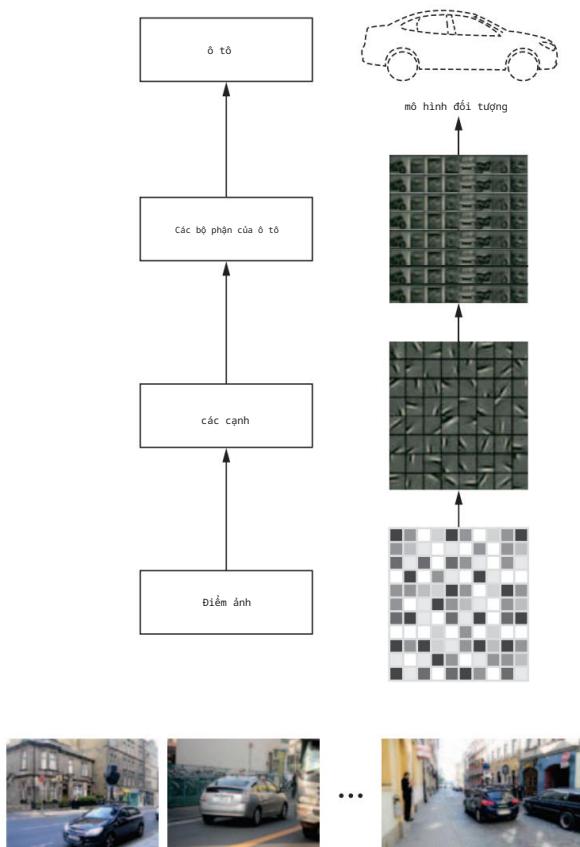
chương. Hãy nhớ rằng đây vẫn là một lĩnh vực nghiên cứu sôi nổi và tích cực trong cộng đồng, vì vậy tôi khuyên bạn nên cập nhật những tiến bộ mới nhất bằng cách làm theo văn học. Các tài nguyên sau đây có thể cung cấp một bản tóm tắt cập nhật về những gì xảy ra trong cộng đồng: Startup<sup>1</sup> và KD Nuggets.<sup>2</sup> Nhưng tôi khuyên bạn nên làm nghiên cứu của riêng bạn và đưa ra kết luận của riêng bạn!

## 6.1 Một cách tiếp cận trực quan để học sâu

Để hiểu sâu  
học tập, hãy chọn  
ứng dụng nhận dạng hình ảnh;  
cụ thể là, đưa ra một hình ảnh  
hoặc một video, làm thế nào để chúng tôi xây dựng  
phân loại sẽ nhận ra  
các đối tượng? Một ứng dụng như vậy  
có tiềm năng lan rộng  
các ứng dụng. Với sự ra đời  
của bản thân định lượng<sup>3,4</sup> và  
Google Glass, chúng ta có thể tưởng tượng  
ứng dụng cho thiết bị này  
nhận dạng các đối tượng và cung  
cấp thông tin cho người dùng.

Hãy lấy ví dụ về  
nhận biết xe hơi. Học sâu xây  
dựng các lớp hiểu biết, với mỗi  
lớp  
sử dụng cái trước. Hình 6.1 cho  
thấy một số lớp hiểu biết có  
thể  
cho một mạng sâu được đào tạo để  
nhận biết ô tô. Cá ví dụ này  
và một số hình ảnh  
phản tiếp theo được sao chép  
lại từ bài giảng của Andrew Ng  
về chủ đề này<sup>5</sup>

Ở dưới cùng của hình 6.1  
bạn có thể thấy một số cổ phiếu  
hình ảnh của xe ô tô. chúng tôi sẽ xem xét



Hình 6.1 Trực quan hóa mạng sâu để nhận dạng ô tô.

Một số nội dung đồ họa được sao chép từ bài nói chuyện của Andrew Ng về chủ đề này, được trích dẫn trước đây. Một bộ ảnh huấn luyện cơ sở được sử dụng để tạo cơ sở cho các cạnh. Các cạnh này có thể được kết hợp để phát hiện các bộ phận của ô tô và các bộ phận này của ô tô có thể được kết hợp để phát hiện một loại đối tượng, trong trường hợp này là ô tô.

<sup>1</sup> Startup.ML, "Deep Learning News," ngày 30 tháng 6 năm 2015, <http://news.startup.ml/> (truy cập ngày 21 tháng 12 năm 2015).

<sup>2</sup> KD Nuggets, "Học sâu," <http://www.kdnuggets.com/tag/deep-learning> (truy cập ngày 21 tháng 12 năm 2015).

<sup>3</sup> Gina Neff và Dawn Nafus, Bản ngã định lượng (Boston: MIT Press, 2016).

<sup>4</sup> Deborah Lupton, Bản ngã định lượng (Cambridge: Polity Press, 2016).

<sup>5</sup> Andrew Ng, "Bay Area Vision Meeting: Unsupervised Feature Learning và Deep Learning," YouTube, Ngày 7 tháng 3 năm 2011, <https://www.youtube.com/watch?v=ZmNOAtZIGk> (truy cập ngày 21 tháng 12 năm 2015).

những bộ đào tạo của chúng tôi. Câu hỏi bây giờ là làm thế nào để chúng ta sử dụng deep learning để nhận ra điểm tương đồng giữa những hình ảnh này, đó là tất cả chúng đều chứa một chiếc ô tô, có thể không có bất kỳ sự thật mặt đất dán nhãn? Thuật toán không nói rằng cảnh có một chiếc ô tô.

Như bạn sẽ thấy, học sâu dựa trên sự trừu tượng hóa khái niệm cao hơn dần dần được xây dựng trực tiếp từ trừu tượng cấp thấp hơn. Trong trường hợp bài toán nhận dạng hình ảnh của chúng ta, chúng ta bắt đầu với thành phần thông tin nhỏ nhất trong ảnh, pixel.

Toàn bộ tập hợp hình ảnh được sử dụng để xây dựng cơ sở của các tính năng-nghĩ lại chương 3 nơi chúng ta đã thảo luận về việc trích xuất cấu trúc từ dữ liệu-có thể được sử dụng ở dạng tổng hợp để phát hiện mức độ trừu tượng cao hơn một chút như đường thẳng và đường cong. Ở cấp độ cao nhất tiếp theo, những đường này là những đường cong được kết hợp để tạo ra các bộ phận của ô tô có đã được nhìn thấy trong tập huấn luyện và các phần này được kết hợp thêm để tạo đối tượng máy dò cho toàn bộ chiếc xe.

Có hai khái niệm quan trọng cần lưu ý ở đây. Đầu tiên, không có kỹ thuật tính năng rõ ràng nào được thực hiện. Nếu bạn còn nhớ, ở cuối chương trước chúng ta đã nói chuyện về tầm quan trọng của việc tạo ra một đại diện tốt cho dữ liệu của bạn. Chúng tôi đã thảo luận điều này trong bối cảnh dự đoán nhấp chuột cho quảng cáo và lưu ý rằng các chuyên gia trong space thường thực hiện việc này theo cách thủ công. Nhưng trong ví dụ này, tính năng học tập không giám sát đã được thực hiện; nghĩa là, các biểu diễn của dữ liệu đã được học mà không có bất kỳ tương tác rõ ràng nào từ người dùng. Điều này có thể song song với cách chúng ta là con người có thể thực hiện nhận dạng-và chúng tôi thực sự rất giỏi trong việc nhận dạng mẫu!

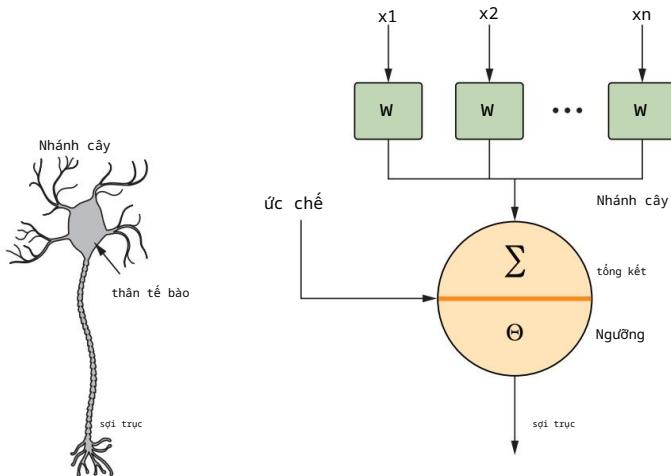
Thực tế quan trọng thứ hai cần lưu ý là khái niệm về một chiếc xe hơi đã không được thực hiện rõ ràng. Cung cấp đủ phương sai trong bộ ảnh đầu vào, chiếc xe độ cao nhất máy dò nên hoạt động đủ tốt trên bất kỳ chiếc xe nào được trình bày. Tuy nhiên, trước khi chúng ta vượt qua chính mình, hãy làm rõ một số điều cơ bản về mạng lưới thần kinh.

## 6.2 Mạng thần kinh

Mạng lưới thần kinh không phải là một công nghệ mới và đã tồn tại từ những năm 1940. Chúng là một khái niệm lấy cảm hứng từ sinh học, theo đó một nơ-ron đầu ra được kích hoạt dựa trên đầu vào từ một số nơ-ron đầu vào được kết nối. Mạng lưới thần kinh là đôi khi được gọi là mạng thần kinh nhân tạo, bởi vì chúng đạt được một cách giả tạo chức năng tương tự như tế bào thần kinh của con người. Jeubin Huang<sup>1</sup> giới thiệu về sinh học của bộ não con người. Mặc dù nhiều khía cạnh về chức năng của bộ não con người vẫn còn là một bí ẩn, chúng ta có thể hiểu được các khía cạnh xây dựng cơ bản của hoạt động - nhưng điều này làm phát sinh ý thức như thế nào lại là một vấn đề khác.

Các tế bào thần kinh trong não sử dụng một số đuôi gai để thu thập cả thông tin đầu ra tích cực (kích thích) và tiêu cực (ức chế) từ các tế bào thần kinh khác và mã hóa. Cái này bằng điện, gửi cái này xuống một sợi trực. Sợi trực này tách ra và đạt tới hàng trăm hoặc hàng ngàn đuôi gai gắn vào các tế bào thần kinh khác. Một khoảng cách nhỏ tồn tại giữa sợi trực và sợi nhánh đầu vào của tế bào thần kinh tiếp theo, và khoảng trống này được gọi là khớp thần kinh.

<sup>1</sup> Jeubin Huang, "Overview of Cerebral Function," Merck Manual, ngày 1 tháng 9 năm 2015, <http://www.merckmanuals.com/professional/neurologic-disorders/function-and-dysfunction-of-the-cerebral-lobes/overview-of-cerebral-function> (truy cập ngày 21 tháng 12 năm 2015).



Hình 6.2 Ở bên trái, chúng tôi cung cấp một sơ đồ về tế bào thần kinh sinh học của con người.

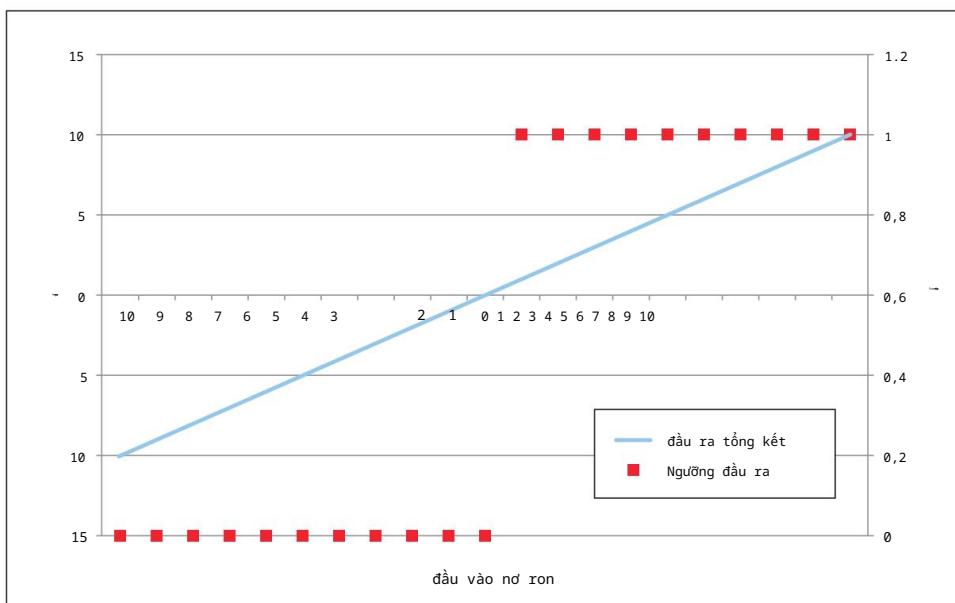
Ở bên phải, chúng tôi hiển thị một mạng lưới thần kinh lấy cảm hứng từ con người được triển khai bằng cách sử dụng phép tính tổng có trọng số, đầu vào ức chế và ngưỡng.

Thông tin điện được chuyển đổi thành đầu ra hóa học, sau đó kích thích den-drite của tế bào thần kinh tiếp theo. Trong kịch bản này, việc học được mã hóa bởi chính tế bào thần kinh. Các tế bào thần kinh chỉ gửi thông điệp xuống sợi trực của chúng nếu kích thích tổng thể của chúng lớn đủ.

Hình 6.2 cho thấy sơ đồ của một nơ-ron sinh học của con người và một nơ-ron nhân tạo do McCulloch và Pitts phát triển, cái gọi là mô hình MCP.<sup>1</sup> Nơ-ron nhân tạo của chúng tôi được xây dựng bằng cách sử dụng một giá trị ngưỡng và tổng kết đơn giản và hoạt động như sau. Hợp lý đầu vào, cả tích cực và tiêu cực, được nhận từ dendrite tương đương và một tổng trọng số được thực hiện. Nếu đầu ra này vượt quá một ngưỡng nhất định và không đầu vào ức chế được quan sát, một giá trị tích cực được phát ra. Đầu ra này sau đó có thể được cung cấp chuyển sang đầu vào của các tế bào thần kinh khác như vậy thông qua các đầu vào tương đương của sợi nhánh của chúng. Một chút suy nghĩ sẽ tiết lộ rằng đây là-bỏ qua đầu vào ức chế-một mô hình tuyến tính trong không gian n chiều, với các hệ số được liên kết, trong đó n là số lượng đầu vào để tế bào thần kinh. Hình 6.3 minh họa hành vi của mô hình này với n = 1.

Trong hình minh họa này, chúng tôi sử dụng một nơ-ron đơn giản được tạo bằng tay với trọng số đơn vị ( $w = 1$ ). Đầu vào của nơ-ron được phép thay đổi từ -10 đến 10 và tổng của giá trị đầu vào có trọng số được cung cấp trên trục Y. Chọn ngưỡng là 0, neuron sẽ kích hoạt nếu đầu vào lớn hơn 0 nhưng không phải ngược lại.

<sup>1</sup> Warren S. McCulloch và Walter H. Pitts, "A Logical Calculus of the Ideas Immanent in Nervous Activity," Bulletin of Mathematical Biophysics 5 (1943): 115-33.



Hình 6.3 MCP dưới dạng mô hình tuyến tính 2 chiều không có đầu vào ức chế. Trọng số của mô hình tương ứng với các hệ số của mô hình tuyến tính. Trong trường hợp này, nơ-ron của chúng tôi chỉ hỗ trợ một đầu vào duy nhất và trọng số đã được đặt thành 1 để minh họa. Với ngưỡng 0, tất cả các đầu vào có giá trị nhỏ hơn hoặc bằng 0 sẽ ngăn nơ-ron kích hoạt, trong khi tất cả các đầu vào có giá trị lớn hơn 0 sẽ khiến nơ-ron kích hoạt.

### 6.3 Perceptron

Trong phần trước chúng tôi đã giới thiệu về nơ-ron MCP. Với cách tiếp cận cơ bản này, nó hóa ra là có thể học và khai quát hóa dữ liệu đào tạo nhưng trong một phạm vi rất hạn chế thời trang. Nhưng chúng ta có thể làm tốt hơn, và do đó, perceptron đã ra đời. tri giác xây dựng trên mô hình MCP theo ba cách quan trọng:<sup>1,2</sup>

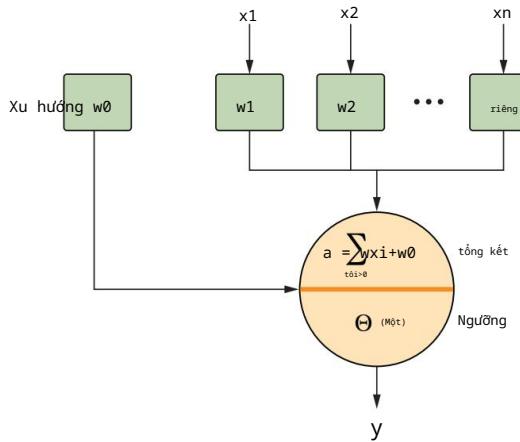
Một độ lệch ngưỡng đã được thêm vào làm đầu vào cho tổng kết. Điều này phục vụ một số mục đích tương đương. Đầu tiên, nó cho phép ghi lại sai lệch từ các nơ-ron đầu vào. Thứ hai, điều đó có nghĩa là các ngưỡng đầu ra có thể được chuẩn hóa xung quanh một giá trị bằng 0 mà không mất tính tổng quát.

Perceptron cho phép các trọng số đầu vào độc lập, vì vậy nơ-ron không cần được kết nối với một đầu vào nhiều lần để có tác động lớn hơn.

Sự phát triển của perceptron báo trước sự phát triển của thuật toán để tìm hiểu các trọng số tốt nhất được cung cấp một bộ dữ liệu đầu vào và đầu ra.

<sup>1</sup> Frank Rosenblatt, *The Perceptron-a automaton nhận thức và công nhận* (New York: Cornell Aeronautical Laboratory, 1957).

<sup>2</sup> Rosenblatt, "The perceptron: Một mô hình xác suất để lưu trữ và tổ chức thông tin trong não," Đánh giá tâm lý 65, không. 6 (tháng 11 năm 1958): 386-408.



Hình 6.4 Perceptron. Đầu vào  $x_n$  được nhận và nhân với trọng số liên quan của chúng, với độ lệch perceptron,  $w_0$ , được thêm vào sau đó. Đầu ra này, được cung cấp bởi  $a$ , sau đó được chuyển qua một hàm ngưỡng để thu được đầu ra.

Hình 6.4 cung cấp một hình ảnh tổng quan về mô hình mờ rộng mới này. Như trước đây, một giá trị trung gian được tạo bằng cách sử dụng tổng trọng số của các đầu vào, nhưng bây giờ chúng ta lưu ý việc bao gồm một giá trị sai lệch,  $w_0$ . Điều này được học cùng với các trọng số đầu vào trong bước đào tạo; thêm về điều này trong các phần sau. trung gian giá trị, được biểu thị bằng  $a$  ở đây, sau đó được chuyển qua hàm ngưỡng để lấy kết quả cuối cùng,  $y$ .

### 6.3.1 Đào tạo

Bây giờ bạn đã biết rằng một mạng thần kinh bao gồm nhiều thành phần cơ bản hơn được gọi là perceptron, hãy xem cách chúng ta huấn luyện một perceptron độc lập. Vì vậy, nó có nghĩa gì để đào tạo một perceptron? Hãy lấy một ví dụ cụ thể hơn bằng cách sử dụng hàm AND logic. Chúng ta sẽ xem xét một perceptron của hai đầu vào nhị phân với hàm kích hoạt ngưỡng nhị phân khoảng 0. Làm thế nào để chúng ta tìm hiểu các trọng số, sao cho đầu ra của perceptron là 1, khi và chỉ khi cả hai đầu vào đều bằng 1? Nói cách khác, có thể chúng tôi chọn các trọng số có giá trị liên tục sao cho tổng trọng số của các đầu vào là lớn hơn 0 khi cả hai đầu vào đều bằng 1 với đầu ra nhỏ hơn 0 khác? Hãy chính thức hóa vấn đề này. Chúng tôi đặt  $X$  làm vectơ giá trị đầu vào nhị phân và  $W$  là vectơ trọng số đầu vào liên tục của chúng tôi.

$$X \stackrel{=} {x_1 \ x_2 \ (\ ,\ )'} W \stackrel{=} {w_0 \ , \ w_1 \ , \ w_2 \ =}$$

Vì vậy, chúng ta cần tìm hiểu các trọng số sao cho các hạn chế sau đây đúng với sự kết hợp của các đầu vào nhị phân  $x_1, x_2$ :

$$\begin{aligned} X \cdot W &> 0 \quad x_1 \cdot x_2 = 1 \\ &\leq 0 \text{ ngược lại} \end{aligned}$$

Thật không may cho chúng tôi, không có giải pháp nếu chúng tôi đặt vấn đề như thế này! Ở đó là hai lựa chọn để giải quyết vấn đề này. Chúng tôi hoặc cho phép ngưỡng để

đi chuyển, xác định nó là một giá trị khác 0 hoặc chúng tôi giới thiệu một phần bù; cả hai đều được cho vay tương đương. Chúng tôi sẽ chọn cái sau trong văn bản này, cung cấp cho chúng tôi các vectơ mới

$$X \stackrel{=}{=} x_1 x_2 ( , ) , 1 W w_1 w_2 = , \theta$$

và những biến đẳng hiện có của chúng ta vẫn giữ nguyên. Bây giờ bạn có thể thấy rằng với trọng lượng cần thận lựa chọn chúng ta có thể tạo hàm AND . Hãy xem xét trường hợp mà

$$w_1 = 1 w_2 = -1 \text{ và } \theta = -1,5$$

Bảng 6.1 cung cấp đầu ra từ perceptron của chúng tôi và đầu ra từ AND chức năng.

Bảng 6.1 So sánh đầu ra của perceptron của chúng tôi với đầu ra của logic AND trả về 1 nếu cả hai đầu vào đều bằng 1. Kết quả được cung cấp là dành cho trường hợp ở đâu  $w_1 = 1, w_2 = -1$  và  $\theta = -1,5$  .

x1	x2	theta	tổng trọng số đầu	của tổng trọng số x1 VÀ x2	
1	0	-1,5	-0,5	tiêu cực	0
0	1	-1,5	-0,5	tiêu cực	0
0	0	-1,5	-1,5	tiêu cực	0
1	1	-1,5	0,5	tích cực	1

Bây giờ chúng ta đã hiểu rằng thực sự có thể biểu diễn một logic AND bằng cách sử dụng một perceptron, chúng ta phải phát triển một cách có hệ thống để tìm hiểu các trọng số của mình theo cách được giám sát thái độ. Nói cách khác, cho một tập dữ liệu bao gồm đầu vào và đầu ra, liên quan đến lin sớm trong trường hợp này, làm thế nào để chúng ta tìm hiểu các trọng số của perceptron của chúng ta? Chúng ta có thể đạt được điều này sử dụng thuật toán perceptron do Rosenblatt phát triển.<sup>1,2</sup> Trong danh sách sau đây, chúng tôi trình bày mã giả sử dụng cho học tập.

### Liệt kê 6.1 Thuật toán học perceptron

Khởi tạo W để chứa các số nhỏ ngẫu nhiên

Đối với mỗi mục trong tập huấn luyện:

Tính toán đầu ra hiện tại của perceptron cho mặt hàng đó.

Đối với mỗi trọng số, hãy cập nhật, tùy thuộc vào độ chính xác của đầu ra.

Càng xa càng tốt. Có vẻ dễ dàng, phải không? Chúng tôi bắt đầu với một số giá trị ngẫu nhiên nhỏ cho trọng số và sau đó chúng tôi lặp lại các điểm dữ liệu của mình và cập nhật trọng số tùy thuộc vào về tính chính xác của perceptron của chúng tôi. Trên thực tế, chúng tôi chỉ cập nhật trọng số nếu chúng tôi nhận sai đầu ra; nếu không, chúng tôi để trọng số một mình. Hơn nữa, chúng tôi cập nhật

<sup>1</sup> Rosenblatt, "The perceptron: Một mô hình xác suất để lưu trữ và tổ chức thông tin trong não."

<sup>2</sup> Rosenblatt, Nguyên tắc của thần kinh học; tri giác và lý thuyết về cơ chế não bộ (Washington: Spartan Sách, 1962).

các trọng số sao cho chúng trở nên giống với các vectơ đầu vào hơn về độ lớn nhưng với đầu tương ứng của giá trị đầu ra. Hãy viết điều này ra một cách chính thức hơn, như hiển thị trong danh sách tiếp theo.

### Liệt kê 6.2 Thuật toán học perceptron (2)

Khởi tạo để chứa các  $w_k$  nhỏ ngẫu nhiên

Đối với mỗi ví dụ  $j$ :

$$y_j(t) = \text{tanh}(w_k^T x_j),$$

Tính toán đầu ra của perceptron với trọng số hiện tại (thời gian  $t$ ).

Đối với mỗi trọng số tính năng  $k$ :

$$\text{tuần}(t+1)_k = (w_k^T x_j) y_j(t) x_{jk},$$

Cập nhật các tính năng. Lưu ý rằng điều này thường được thực hiện dưới dạng thao tác véc-tơ hơn là vòng lặp for.

Các tính năng chỉ được cập nhật nếu đầu ra dự kiến và đầu ra thực tế khác nhau. Nếu đúng như vậy, chúng ta di chuyển trọng số theo dấu của câu trả lời đúng nhưng độ lớn được đưa ra bởi đầu vào tương ứng.

Với điều kiện là dữ liệu đầu vào có thể phân tách tuyến tính, một thuật toán như vậy được đảm bảo để hội tụ một giải pháp.<sup>1</sup>

### 6.3.2 Đào tạo một perceptron trong scikit-learning

Trước đây chúng tôi đã trình bày dạng đơn giản nhất của mạng thần kinh, perceptron; chúng tôi cũng đã thảo luận về cách thuật toán này được đào tạo. Bây giờ chúng ta hãy chuyển sang scikit-learning và khám phá cách chúng ta có thể đào tạo một trí giác bằng cách sử dụng một số dữ liệu. Danh sách tiếp theo cung cấp để thực hiện các thao tác nhập cần thiết và tạo một mảng điểm dữ liệu NumPy.

### Liệt kê 6.3 Tạo dữ liệu cho việc học perceptron

```
nhập numpy dưới dạng np
nhập matplotlib.pyplot dưới dạng plt
nhập ngẫu nhiên
từ sklearn.linear_model nhập perceptron

# Hãy thiết lập dữ liệu và mục tiêu của chúng ta
dữ liệu = np.array([[0,1],[0,0],[1,0],[1,1]])
mục tiêu = np.array([0,0,0,1])
```

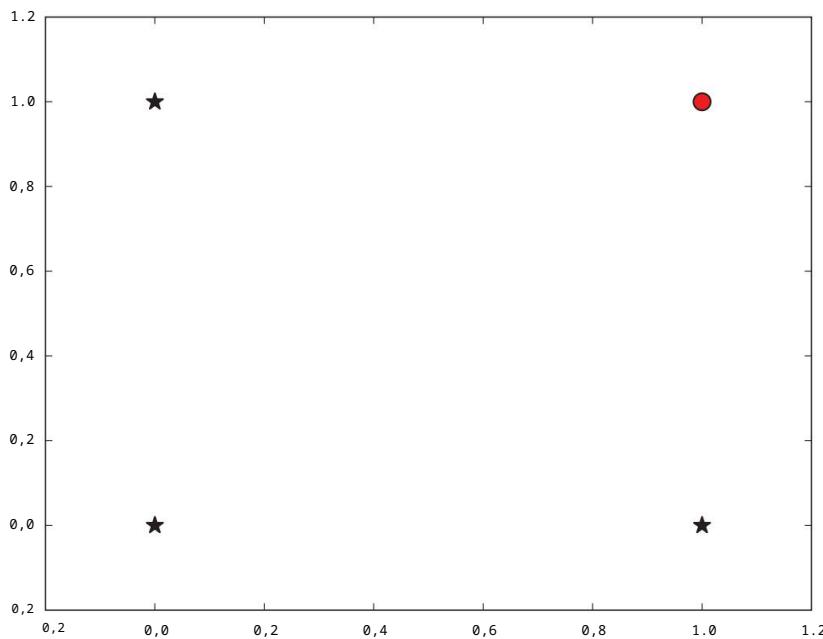
Tạo bốn điểm dữ liệu trong một mảng NumPy.

Chỉ định các lớp của các điểm dữ liệu này. Trong ví dụ này, chỉ điểm dữ liệu  $x=1, y=1$  có lớp mục tiêu là 1; các điểm dữ liệu khác đã được gán một lớp 0.

Trong danh sách 6.3, chúng tôi thực hiện các thao tác nhập cần thiết cho ví dụ perceptron duy nhất của chúng tôi và tạo một tập dữ liệu rất nhỏ chỉ có bốn điểm dữ liệu. Tập dữ liệu này được chứa trong một mảng NumPy được gọi là data. Mỗi điểm dữ liệu được gán cho lớp 0 hoặc lớp 1, với các lớp này được lưu trữ trong mảng đích. Hình 6.5 cung cấp một tổng quan đồ họa của dữ liệu này.

Trong hình 6.5, điểm dữ liệu duy nhất có lớp tích cực (có nhãn là 1) được tìm thấy tại tọa độ  $(1,1)$  và được thể hiện bằng màu đỏ. Tất cả các điểm dữ liệu khác được liên kết với

<sup>1</sup> Brian Ripley, Nhận dạng mẫu và Mạng thần kinh (Cambridge: Nhà xuất bản Đại học Cambridge, 1996).



Hình 6.5 Tổng quan đồ họa về dữ liệu của chúng tôi cho một perceptron. Dữ liệu có nhãn lớp 1 được biểu thị bằng dấu chấm tròn, trong khi dữ liệu có nhãn lớp 0 được biểu thị bằng dấu sao. Mục đích của perceptron là tách các điểm này.

lớp tiêu cực. Danh sách sau đây cung cấp mã mẫu để huấn luyện perceptron đơn giản của chúng ta và trả về các hệ số ( $w_1$ ,  $w_2$  tương ứng với  $x_1$  và  $x_2$ ) cùng với với độ lệch  $w_0$ .

#### Lệnh 6.4 Huấn luyện một perceptron đơn lẻ

```
p = perceptron.Perceptron(n_iter=100)
p_out = p.fit(dữ liệu,mục tiêu)
in p_out
msg = ("Hệ số: %s, Chặn: %s")
in thông điệp % (str(p.coef_),str(p.intercept_))
```

Tạo một perceptron duy nhất. `n_iter` chỉ định số lần dữ liệu đào tạo sẽ được lặp lại khi đào tạo.

Huấn luyện perceptron bằng cách sử dụng dữ liệu và các lớp mục tiêu liên quan.

In ra các hệ số và độ chêch của perceptron.

Lệnh 6.4 cung cấp đầu ra tương tự như sau:

```
Perceptron(alpha=0.0001, class_weight=None, eta0=1.0, fit_intercept=True,
n_iter=100, n_jobs=1, penalty=None, random_state=0, shuffle=False,
dài dòng=0, warm_start=Sai)
Hệ số: [[ 3.  2.]] ,Đánh chặn: [-4.]
```

Dòng đầu tiên cung cấp các tham số theo đó perceptron được đào tạo; các thứ hai cung cấp trọng số đầu ra và độ lệch của perceptron. Đừng lo lắng nếu các hệ số và hệ số chặn hơi khác khi bạn chạy cái này. Có nhiều giải pháp cho vấn đề này và thuật toán học có thể trả về bất kỳ giải pháp nào trong số đó. Cho một hiểu rõ hơn về các tham số, tôi khuyến khích bạn đọc các tài liệu liên quan scikit-learn tài liệu.<sup>1</sup>

### 6.3.3 Diễn giải hình học của perceptron cho hai đầu vào

Trong ví dụ này, chúng tôi đã đào tạo thành công một perceptron duy nhất và trả về trọng số và độ lệch của perceptron cuối cùng. Tuyệt vời! Nhưng làm thế nào để chúng ta diễn giải những trọng số này một cách trực quan? May mắn thay, điều này có thể dễ dàng thực hiện được trong không gian 2 chiều và chúng ta có thể mở rộng điều này trực giác vào không gian chiều cao hơn nữa.

Hãy chỉ xem xét perceptron cho hai giá trị đầu vào. Từ hình 6.3 ta có

$$y = \bar{w}^T w_1 x_1 + w_2 x_2 + b$$

Điều này sẽ quen thuộc với bạn khi phương trình của một mặt phẳng (ba chiều) trong  $x_1$ ,  $x_2$  và  $y$ . Nếu chúng không tương đương, mặt phẳng này giao với mặt phẳng quan sát của hình 6.5 và bạn còn lại một đường thẳng. Khi nhìn từ điểm quy chiếu của hình 6.5, các điểm về một phía của đường tương ứng với các giá trị của các ~~điểm~~ <sup>điểm</sup> ~~đã~~ <sup>đã</sup> được lai trên phía bên kia của dòng tương ứng với các giá trị của  $w_1 x_1 + w_2 x_2 + b$ . Điểm trên đường core  $x_1 w_1 + x_2 w_2 + b < 0$  tài trợ chở  $w$ . Hãy lấy ví dụ cụ thể từ trước đó và hình dung điều này.

Sử dụng các hệ số vừa cung cấp, chúng ta có phương trình của một mặt phẳng được cho bởi

$$y = 4x_1 - 2x_2$$

Giá trị của  $y$  nằm ở góc 90 độ so với mặt phẳng ( $x_1, x_2$ ) và do đó có thể được coi là dọc theo một đường thẳng theo mắt người xem, đi thẳng qua mặt phẳng xem. Giá trị của  $y$  trên mặt phẳng xem được cho bởi 0, và vì vậy chúng ta có thể tìm giao tuyến của thay thế giá trị của  $y=0$  trong phương trình trước:

$$0 = 4x_1 - 2x_2$$

$$4x_1 - 2x_2 = 0$$

$$x_2 = 2x_1$$

Dòng cuối cùng này theo dạng tiêu chuẩn của một đường thẳng; bây giờ tất cả những gì chúng ta cần làm là vẽ đồ thị điều này để xem perceptron đã phân tách dữ liệu huấn luyện của chúng ta như thế nào. Danh sách tiếp theo cung cấp mã liên quan và hình 6.6 cung cấp đầu ra.

---

<sup>1</sup> Scikit-learn, "Perceptron," ngày 01 tháng 1 năm 2014, [http://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.Perceptron.html](http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Perceptron.html) (truy cập ngày 25 tháng 2 năm 2016).

## Liệt kê 6.5 Vẽ sơ đồ đầu ra của perceptron

```

colors = np.array(['k','r'])
diểm đánh dấu = np.array(['*','o'])

đối với dữ liệu, mục tiêu trong zip(dữ liệu, mục tiêu):
plt.scatter(dữ liệu[0],dữ liệu[1],s=100,
            c=màu sắc[mục tiêu],diễn giải đánh dấu=diễn giải đánh dấu[mục tiêu])

cáp độ = -p.coef_[0]/p.coef_[0][1]
chặn = -p.intercept_/p.coef_[0][1]

x_vals = np.linspace(0,1)
y_vals = grad*x_vals + chặn
plt.plot(x_vals,y_vals)

plt.show()

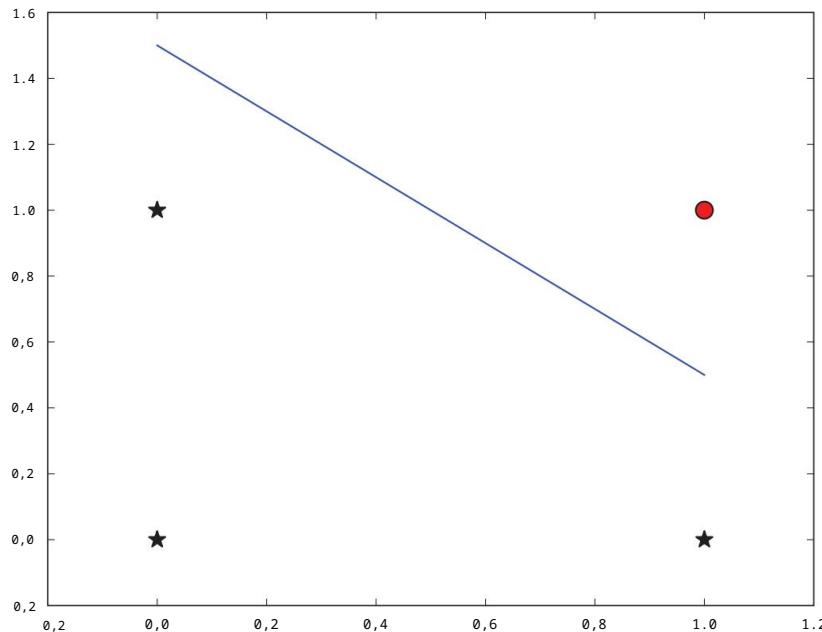
```

Thiết lập một mảng màu và vẽ các điểm dữ liệu: ngôi sao đen cho lớp 0, chấm đỏ cho lớp 1.

Tính các tham số của đường thẳng (giao tuyến của hai mặt phẳng).

Tạo các điểm dữ liệu và vẽ đường giao tuyến giữa hai mặt phẳng.

Trong danh sách 6.5, chúng tôi vẽ bốn điểm dữ liệu cùng với hình chiếu của đường phân cách mặt phẳng được học bởi perceptron lên mặt phẳng xem. Điều này cung cấp cho chúng tôi một tỷ lệ riêng biệt như trong hình 6.6. Nói chung, đối với số lượng biến đầu vào lớn hơn, bạn có thể nghĩ trong số này tồn tại trong không gian n chiều, với perceptron phân tách chúng bằng cách sử dụng một siêu phẳng trong  $n+1$  chiều. Bây giờ bạn có thể thấy rằng tuyến tính cơ bản dạng của perceptron, tức là với ngưỡng kích hoạt, tương đương với sự phân tách sử dụng một siêu phẳng. Do đó, các mô hình như vậy chỉ được sử dụng khi dữ liệu là tuyến tính có thể tách rời và sẽ không thể tách biệt các lớp tích cực và tiêu cực nếu không.

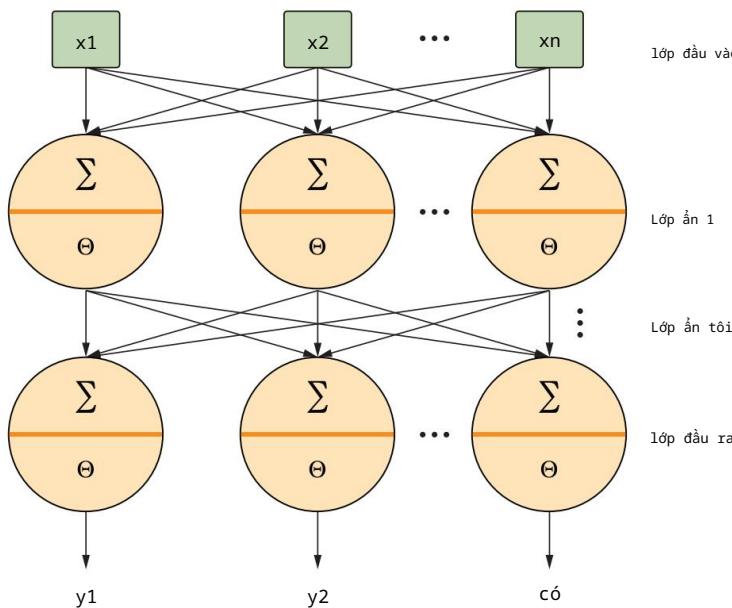


Hình 6.6 Hình chiếu của mặt phân cách của chúng ta lên mặt phẳng quan sát ( $y=0$ ). Tất cả các điểm phía trên bên phải của hình thỏa mãn ràng buộc  $W \cdot x > 0$ , trong khi các điểm phía dưới bên trái của đường thẳng thỏa mãn ràng buộc  $W \cdot x < 0$ .

## 6.4 Perceptron nhiều lớp

Trong các phần trước, chúng ta đã xem xét deep learning từ cấp độ rất cao, và bạn bắt đầu hiểu những điều cơ bản xung quanh mạng thần kinh, cụ thể là một đơn vị duy nhất của một mạng lưới thần kinh được gọi là một perceptron. Chúng tôi cũng chỉ ra rằng dạng cơ bản của perceptron tương đương với một mô hình tuyến tính.

Để thực hiện phân tách phi tuyến tính, chúng ta có thể giữ chức năng kích hoạt nguồn đơn giản và tăng độ phức tạp của kiến trúc mạng để tạo ra cái gọi là mạng chuyển tiếp nguồn cấp dữ liệu nhiều lớp. Đây là những mạng có perceptron được tổ chức theo lớp, với đầu vào của lớp được cung cấp bởi lớp trước đó và đầu ra của lớp này đóng vai trò là đầu vào cho lớp tiếp theo. Chuyển tiếp nguồn cấp dữ liệu đến từ thực tế là dữ liệu chỉ chảy từ đầu vào đến đầu ra của mạng chứ không phải trong hướng ngược lại, nghĩa là, không có chu kỳ. Hình 6.7 cung cấp một bản tóm tắt đồ họa về điều này khái niệm, mở rộng ký hiệu mà chúng ta đã sử dụng trong hình 6.3.



Hình 6.7 Một perceptron đa lớp. Đọc từ trên xuống dưới, nó bao gồm một lớp đầu vào (vector giá trị X), một số lớp ẩn và lớp đầu ra trả về vectơ Y.

Với tính thần minh minh tính phi tuyến tính, chúng ta hãy xem xét một ví dụ rất nhỏ mà sẽ thất bại nếu chúng ta trình bày nó với một perceptron, cụ thể là **hàm XOR**. ví dụ này được lấy từ cuốn sách năm 1969 của Minsky và Papert, *Perceptrons: An Introduction to Computational Geometry*.<sup>1</sup> Sau đó, chúng ta sẽ xem xét cách một perceptron hai lớp có thể được sử dụng để

<sup>1</sup> Marvin Minsky và Seymour Papert, *Perceptrons: Giới thiệu về Hình học Tính toán* (Boston, MA: MIT Press, 1969).

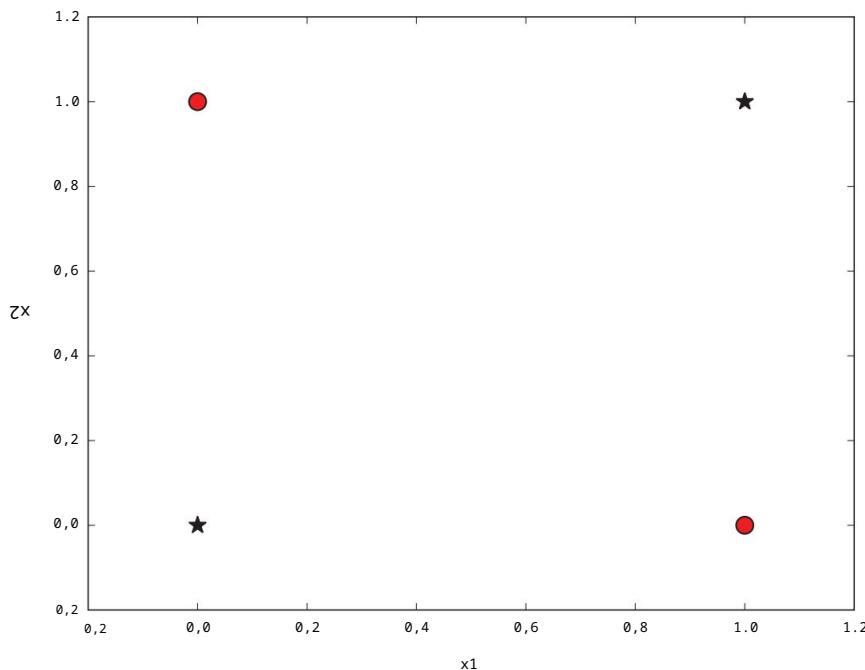
xấp xỉ hàm này và thảo luận về thuật toán lan truyền ngược được sử dụng để huấn luyện một mạng lưới như vậy. Nhớ lại rằng hàm XOR hoạt động như được cung cấp bởi bảng 6.2.

Bảng 6.2 Giá trị đầu vào và đầu ra cho hàm XOR. Xuất ra 1 nếu  $x_1$  hoặc  $x_2$  được đặt thành 1 nhưng cả hai số 0 đều được đặt thành 1 (hoặc cả hai được đặt thành 0).

$x_1$	$x_2$	đầu ra
00 0		
01 1		
10 1		
11 0		

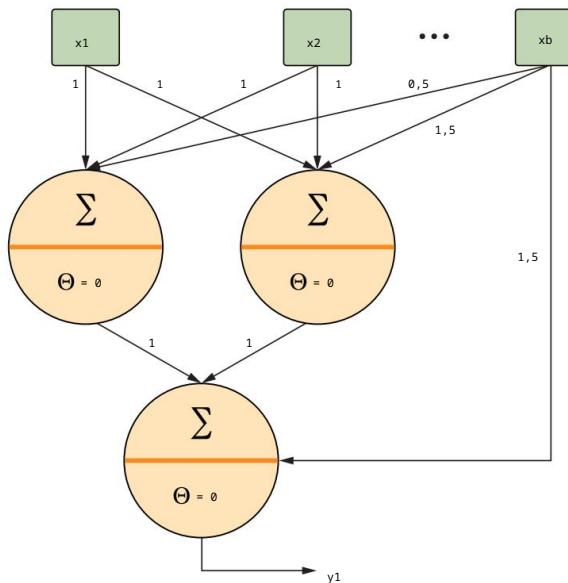
Nếu chúng ta xem xét hàm XOR bằng đồ thị, sử dụng các quy ước tương tự như trong hình 6.5, ta được hình 6.8.

Như bạn có thể thấy trong hình 6.8, đầu ra từ hàm XOR không tuyến tính có thể tách rời trong không gian hai chiều; không tồn tại một siêu phẳng duy nhất nào có thể tách biệt các lớp tích cực và tiêu cực một cách hoàn hảo. Cố gắng vẽ một đường ở bất cứ đâu trên biểu đồ này



Hình 6.8 Biểu diễn đồ họa của hàm XOR. Các lớp tích cực được chỉ định bằng các vòng tròn màu đỏ, trong khi các lớp tiêu cực được chỉ định bằng các ngôi sao màu đen. Không một siêu phẳng đơn lẻ nào có thể tách các điểm dữ liệu này thành hai tập hợp và vì vậy chúng tôi nói rằng tập dữ liệu không thể phân tách tuyến tính.

với tất cả các lớp tích cực ở một bên của đường và tất cả các lớp tiêu cực ở phía đối diện và bạn sẽ thất bại! Tuy nhiên, chúng ta có thể tách các điểm dữ liệu này nếu chúng ta có nhiều hơn một siêu phẳng và một cách để kết hợp chúng. Vì vậy, chúng ta hãy làm điều đó! Đây là tương đương với việc tạo một mạng với một lớp ẩn duy nhất và một lớp đặt kết hợp cuối cùng. Hãy xem hình 6.9, minh họa một mạng như vậy bằng đồ họa.



Hình 6.9 Một perceptron hai lớp có thể tách hàm phi tuyến tính có thể tách rời (XOR). Các giá trị trên các kết nối thể hiện trọng số của các kết nối đó. Việc đưa ra thuật ngữ sai lệch ngũ ý rằng ngưỡng kích hoạt cho các tế bào thần kinh của chúng ta là 0.

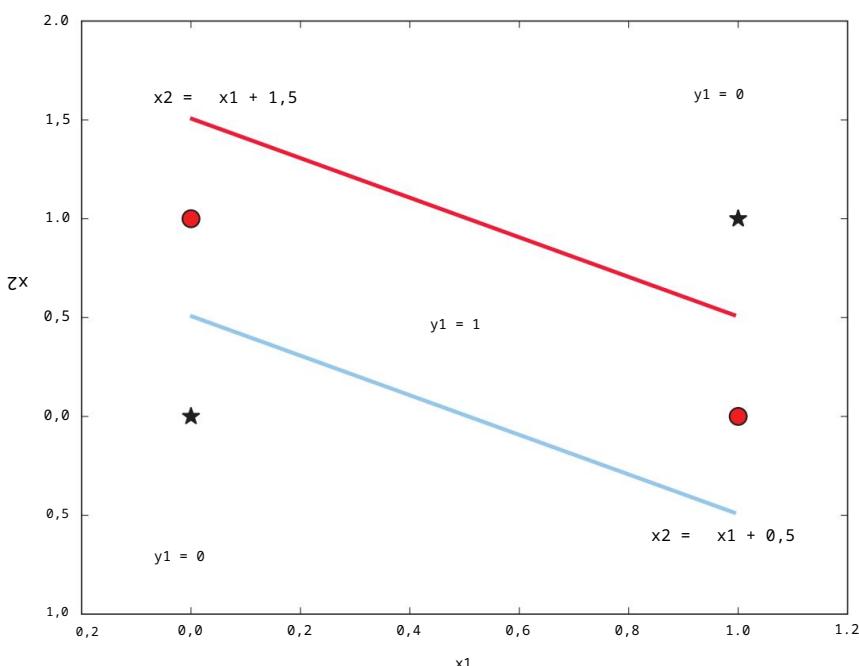
Về mặt khái niệm, hai nơ-ron ẩn tương ứng với hai siêu phẳng. Perceptron kết hợp cuối cùng tương đương với AND logic trên đầu ra của hai nơ-ron ẩn. Điều này có thể được coi là chọn ra các khu vực của mặt phẳng  $(x_1, x_2)$  mà hai nơ-ron ẩn kích hoạt cùng nhau.

Những gì bạn thấy ở đây là một mạng ẩn hai lớp, với hai đầu vào và một đầu ra.

Được kết nối với mỗi nút ẩn và nút đầu ra là một thuật ngữ sai lệch. nhớ từ trước đó bản thân độ lệch sẽ luôn bằng 1; chỉ trọng lượng sẽ thay đổi. Điều này cho phép cả cấu hình kích hoạt và độ lệch của các nút sẽ được học trong quá trình huấn luyện. Hãy dành một chút thời gian để thuyết phục bản thân rằng điều này thực sự hiệu quả.

Mỗi nút ẩn tạo ra một siêu phẳng duy nhất, như với trường hợp perceptron duy nhất của chúng ta từ hình 6.6, và chúng được kết hợp với một perceptron cuối cùng. Perceptron cuối cùng này chỉ đơn giản là một **cỗng AND** khi tuân theo hai điều kiện. Đầu tiên là đầu vào trong không gian  $x_1, x_2$  nằm trên đường màu xanh trong hình 6.10. thứ hai là đầu vào nằm dưới đường màu đỏ, cũng được đưa ra trong hình 6.10. Như vậy, perceptron hai lớp này vạch ra đường chéo trên không gian bao gồm cả hai ví dụ tích cực từ dữ liệu đào tạo nhưng không có ví dụ tiêu cực. Nó đã tách thành công một tập dữ liệu không thể tách tuyến tính.

Trong phần này, chúng ta đã nghiên cứu ứng dụng của mạng nơ-ron vào phi tuyến tính bộ dữ liệu có thể tách rời. Sử dụng hàm XOR làm ví dụ, chúng tôi đã chỉ ra rằng có thể để tạo một mạng thần kinh bằng tay phân tách một tập hợp có thể phân tách phi tuyến tính và cung cấp trực giác về cách thức hoạt động của mạng này về mặt hình học. Chúng ta đang thiếu một bước quan trọng, Tuy nhiên! Điều quan trọng là có thể tự động tìm hiểu trọng số cho mạng



Hình 6.10 Tách một bộ dữ liệu có thể tách phi tuyến tính bằng cách sử dụng mạng thần kinh được đưa ra trong hình 6.9. Bạn nên chú ý rằng nơ-ron bên trái của hình 6.9 khi giao nhau với mặt phẳng quan sát sẽ tạo ra đường dưới cùng, trong khi nơ-ron ngoài cùng bên phải tạo ra đường trên cùng. Tê bào thần kinh ngoài cùng bên trái chỉ kích hoạt đầu ra 1 khi đầu vào nằm trên dòng dưới cùng, trong khi tế bào thần kinh ngoài cùng bên phải chỉ kích hoạt đầu ra 1 khi đầu vào nằm dưới dòng trên cùng. Tê bào thần kinh kết hợp cuối cùng chỉ kích hoạt  $y=1$  khi cả hai ràng buộc này đều được thỏa mãn. Do đó, mạng chỉ xuất ra 1 khi các điểm dữ liệu nằm trong hành lang hẹp giữa dòng dưới cùng và dòng trên cùng.

được cung cấp một tập dữ liệu huấn luyện. Những trọng số này sau đó có thể được sử dụng để phân loại và dự đoán dữ liệu ngoài các đầu vào ban đầu. Đây là chủ đề của phần tiếp theo.

#### 6.4.1 Huấn luyện sử dụng lan truyền ngược

Trong các ví dụ trước, chúng ta đã sử dụng hàm bước làm hàm kích hoạt nơ-ron.

Thật không may, rất khó để tìm ra một phương pháp tự động để đào tạo một mạng như vậy. Điều này là do chức năng bước không cho phép chúng tôi mã hóa không chắc chắn trong bất kỳ

hình thức – các ngưỡng là khó.

Sẽ phù hợp hơn nếu chúng ta có thể sử dụng một hàm xấp xỉ bước

hoạt động nhưng dần dần. Theo cách này, một thay đổi nhỏ đối với một trọng số trong

mạng sẽ thực hiện một thay đổi nhỏ đối với hoạt động của toàn bộ mạng. Đây là

thực sự những gì chúng ta sẽ làm. Nhớ lại chức năng logistic từ chương 4. Thay vì sử dụng

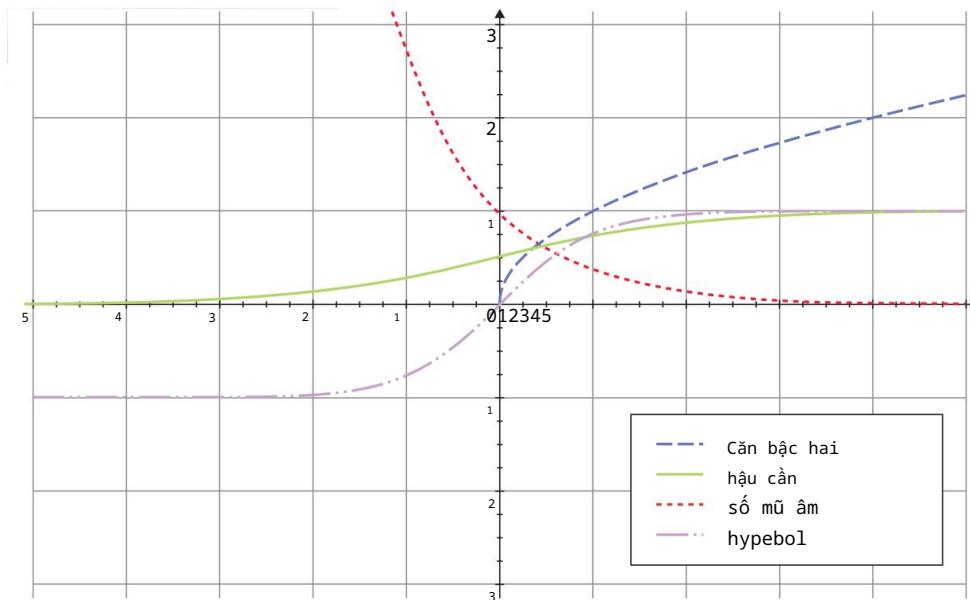
bước, bây giờ chúng ta sẽ thay thế chức năng này bằng một chức năng kích hoạt tổng quát hơn. bên trong

phần tiếp theo chúng tôi sẽ giới thiệu ngắn gọn về các chức năng kích hoạt phổ biến trước khi giải thích

việc lựa chọn hàm kích hoạt sẽ giúp chúng ta rút ra thuật toán huấn luyện như thế nào.

#### 6.4.2 Các chức năng kích hoạt

Hãy dành một chút thời gian để xem xét một số chức năng kích hoạt có thể có cho per ceptron của chúng ta. Chúng ta đã thấy trường hợp đơn giản nhất-một người đứng cứng quanh 0. Với một bù bằng 0, điều này mang lại cấu hình đầu ra như được cho trong hình 6.3. Nhưng những gì khác có thể chúng ta làm gì? Hình 6.11 cung cấp cấu hình kích hoạt của một số chức năng khác; của họ định nghĩa sau.



Hình 6.11 Cấu hình đầu ra cho một số chức năng kích hoạt trong cùng phạm vi như được cung cấp trong hình 6.3. Cấu hình kích hoạt được thể hiện là căn bậc hai, logistic, hàm mũ âm và hyperbolic.

Căn bậc hai-Được định nghĩa  $y = \sqrt{x}$ , miền  $[0, \infty]$ , khoảng  $[0, \infty]$ .  
 là y Logistic-Được  $y = \frac{1}{1 + e^{-x}}$ , miền được xác định cho  $(-\infty, \infty)$ , phạm vi  $[0, 1]$ .  
 định nghĩa là Hàm mũ âm-Cho bởi, miền  $(-\infty, \infty)$  phạm vi  $[0, \infty]$ .  
 Hyperbolic (tanh)-Định nghĩa là  $y = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ . Lưu ý rằng điều này tương đương với logistic với đầu ra của nó được chuyển đổi sang một phạm vi khác, miền  $(-\infty, \infty)$  phạm vi  $[-1, 1]$ .

Nói chung, việc sử dụng các hàm kích hoạt như vậy cho phép chúng ta tạo và huấn luyện các mạng thần kinh nhiều lớp xấp xỉ một lớp hàm lớn hơn nhiều. Nhiều nhất tính chất quan trọng của hàm kích hoạt là nó khả vi. Bạn sẽ thấy tại sao trong phần sau.

### Hồi quy logistic, perceptron và Mô hình tuyến tính tổng quát

Hãy nhớ lại chương 4 nơi chúng tôi đã giới thiệu khái niệm về hồi quy logistic. Ở đó chúng tôi đã xác định rằng một mô hình phản hồi tuyến tính sẽ không phù hợp để ước tính xác suất và thay vào đó, chúng tôi đã sửa đổi phản hồi logistic thành đường cong để tạo ra một mô hình phù hợp hơn. phản ứng. Từ điểm bắt đầu này, chúng tôi rút ra rằng tỷ lệ cược log là tuyến tính trong sự kết hợp của các trọng số và biến đầu vào và áp dụng điều này cho một bài toán phân loại.

Trong chương này, chúng tôi bắt đầu từ một khái niệm sinh học cơ bản và xây dựng một tính toán chủ nghĩa hình thức nắm bắt được điều này. Chúng tôi đã không thảo luận về xác suất mà bắt đầu từ một quan điểm kích hoạt trong một tế bào thần kinh. Theo trực giác, chúng tôi đã mở rộng điều này thành một khái niệm chung và đạt được cùng một phương trình, cụ thể là,

$$y = \frac{1}{1 + e^{-(w_0 + w_1 x_1 + \dots + w_n x_n)}}$$

Trên thực tế, những gì chúng ta gặp phải ở đây là một loại vấn đề tổng quát hơn được gọi là mô hình tuyến tính tổng quát (GLM).a Trong lớp mô hình này, một mô hình tuyến tính ( $w_0 + w_1 x_1 + \dots + w_n x_n$ ) có liên quan đến biến đầu ra,  $y$ , bởi một hàm liên kết, trong đó trường hợp này là chức năng logistic.

$$y = \frac{1}{1 + e^{-z}}$$

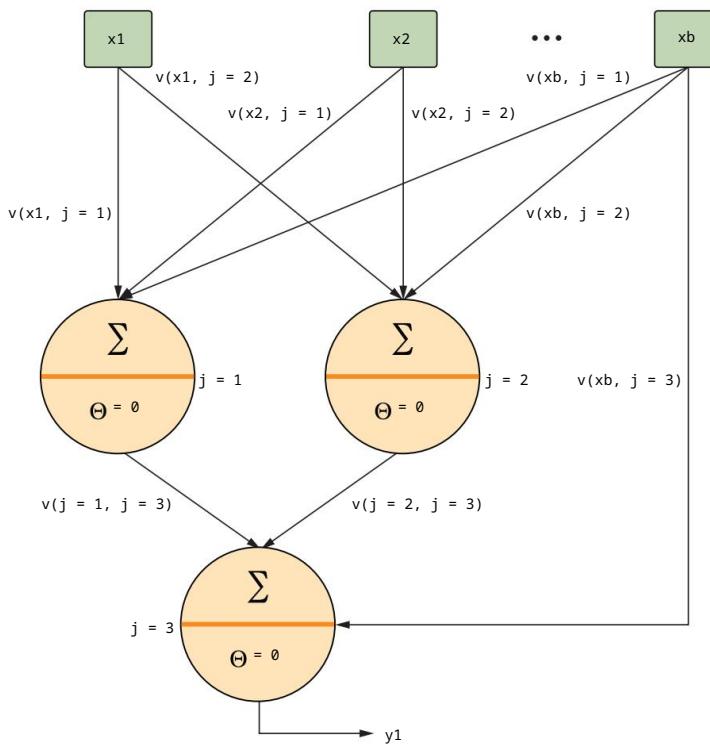
Sự tương đương giữa các thuật toán và khái niệm này là phổ biến trong học máy và bạn đã nhìn thấy nó trong các trang của cuốn sách này. Chỉ cần nghĩ lại phần 2.5 ở đó chúng ta đã thảo luận về sự tương đương của tối đa hóa kỳ vọng (EM) đối với mô hình hỗn hợp Gaussian với hiệp phương sai ràng buộc và kết hợp, và thuật toán vanilla k-means. Lý do thông thường cho điều này là nhiều nhà nghiên cứu đã bắt đầu từ các điểm khác nhau trong quá trình nghiên cứu và phát hiện ra sự tương đương thông qua việc mở rộng các khối xây dựng cơ bản.

Một. Peter McCullagh và John A. Nelder, Generalized Linear Models (London: Chapman và Hội trường/CRC) 1989.

#### 6.4.3 Trực giác đằng sau lan truyền ngược

Để cung cấp trực giác đằng sau lan truyền ngược, chúng ta sẽ làm việc với cùng một ví dụ trước đây, hàm XOR, nhưng bây giờ chúng ta sẽ cố gắng tìm hiểu trọng lượng hơn là chỉ định chúng bằng tay. Cũng lưu ý rằng kể từ bây giờ, việc kích hoạt chức năng sẽ được coi là sigmoid (hậu cần). Hình 6.12 cung cấp đồ họa tổng quan về mạng mới của chúng tôi.

Công việc của chúng ta là họđộđẹp( $x_1, x_2$ ), bằng cách sử dụng một tập dữ liệu đào tạo được chỉ định. Cụ thể hơn, liệu chúng ta có thể đưa ra một thuật toán giảm thiểu sai số (bình phương sự khác biệt giữa giá trị dự kiến và thực tế của  $y$ ) trên tập dữ liệu đó, khi tập dữ liệu đó đầu vào được áp dụng cho mạng?



Hình 6.12 Tổng quan về ví dụ lan truyền ngược của chúng tôi. Đưa ra một tập hợp các đầu vào  $(x_1, x_2)$  và các biến mục tiêu ( $y_1$ ) tuân theo hàm XOR trên  $x_1$  và  $x_2$ , chúng ta có thể tìm hiểu các giá trị của  $W$  để giảm thiểu sự khác biệt bình phương giữa các giá trị huấn luyện và đầu ra của mạng không? Trong e) này, chúng tôi sử dụng chức năng kích hoạt logistic, cụ thể là  $\sigma(\theta) = 1/(1 + e^{-\theta})$ .

Một cách để làm điều này là thông qua một thuật toán được gọi là backpropagation. Rithm thuật toán này hoạt động rộng rãi như sau. Đầu tiên, chúng tôi khởi tạo tất cả các trọng số thành các giá trị ngẫu nhiên, và sau đó chúng tôi chuyển một mục dữ liệu đào tạo duy nhất trên toàn mạng. Chúng tôi tính toán lỗi ở đầu ra và lan truyền ngược lỗi này qua mạng, do đó có tên!

Mỗi trọng số trong mạng được thay đổi theo hướng tương ứng với hướng mà giảm thiểu lỗi của mạng. Điều này tiếp tục cho đến khi một điều kiện kết thúc được đáp ứng, ví dụ: một số lần lặp lại bị trùng hoặc mạng đã hội tụ.

#### 6.4.4 Lý thuyết lan truyền ngược

Để dễ hiểu, quy tắc cập nhật cho lan truyền ngược có thể được xem xét trong hai phần: cập nhật trọng số dẫn đến nơ-ron đầu ra và cập nhật trọng số dẫn đến các tế bào thần kinh ẩn. Cả hai đều giống hệt nhau về mặt logic, nhưng toán học cho sau này phức tạp hơn một chút. Vì lý do này, chúng tôi sẽ chỉ thảo luận vấn đề đầu tiên ở đây để cung cấp cho bạn một hướng dẫn về backpropagation. Xem tạp chí Nature chuyên đề1 nếu bạn muốn hiểu hình thức đầy đủ.

<sup>1</sup> David E. Rumelhart, Geoffrey E. Hinton, và Ronald J Williams, "Học các biểu diễn bằng lỗi lan truyền ngược," Nature 323 (Tháng 10 năm 1986): 533-36.

Điều đầu tiên cần lưu ý về đào tạo là chúng tôi quan tâm đến lỗi tại đầu ra thay đổi đối với sự thay đổi trong một giá trị trọng lượng duy nhất. Chỉ thông qua điều này giá trị, chúng ta có thể di chuyển trọng lượng theo hướng giảm thiểu lỗi đầu ra không. Hãy bắt đầu bằng cách tìm ra **đạo hàm riêng** của lỗi đối với một đầu vào cụ thể trọng lượng ở lớp bên dưới. Đó là, chúng tôi giả định rằng tất cả các trọng số khác không đổi.

Để làm điều này, chúng ta sẽ sử dụng quy tắc dây chuyền:

$$\begin{aligned} \frac{\delta E}{\delta w_{ij}} &= \\ b & \frac{\partial (\dots)}{\partial w_{ij}} \times \frac{\delta E}{\delta o_j(\dots)} \\ C & \frac{\partial o_j(\dots)}{\partial w_{ij}} \times \\ D. & \frac{\delta o_j(\dots)}{\delta w_{ij}} \end{aligned}$$

Nói một cách đơn giản, tốc độ thay đổi của lỗi đầu ra được liên kết với trọng số, thông qua tốc độ thay đổi sau:

- B Lỗi do đầu ra của chức năng kích hoạt
- C Đầu ra của hàm kích hoạt với tổng trọng số của các đầu vào của nó
- D Tổng trọng số của đầu vào và trọng số riêng

Nếu bạn còn nhớ trước đó, chức năng kích hoạt hậu cần được sử dụng vì nó tạo ra đào tạo dễ dàng theo dõi. Lý do chính cho điều này là hàm khả vi. Bạn hãy giờ nêu hiểu tại sao đây là một yêu cầu. Thuật ngữ C của phương trình là đẳng thức cho mượn đạo hàm của hàm kích hoạt. Điều này có thể được viết như

$$\frac{\delta o_j(\dots)}{\delta w_{ij}} = \frac{\delta}{\delta w_{ij}} o_j(\dots) = \frac{\delta}{\delta w_{ij}} \sum_k w_{kj} f(x_k)$$

nghĩa là, tốc độ thay đổi đầu ra của hàm kích hoạt của chúng ta có thể được viết dưới dạng của chính chức năng kích hoạt! Nếu chúng ta có thể tính toán B và D, chúng ta sẽ biết hướng trong đó di chuyển bất kỳ trọng lượng cụ thể nào để giảm thiểu lỗi ở đầu ra. Nó hóa ra điều này thực sự có thể. Thuật ngữ D có thể được phân biệt trực tiếp như

$$\frac{\delta o_j(\dots)}{\delta w_{ij}} \times i =$$

Nghĩa là, tốc độ thay đổi của đầu vào đối với hàm kích hoạt đối với một trọng số cụ thể liên kết I và J chỉ được cho bởi giá trị của chính  $x_I$ . Bởi vì chúng tôi đã nhìn chỉ ở lớp đầu ra, xác định chênh lệch của lỗi cho đầu ra B thật dễ dàng nếu chúng ta có thể trực tiếp rút ra khái niệm lỗi:

$$\frac{\delta E}{\delta o_j(\dots)} = \frac{\delta}{\delta o_j(\dots)} \sum_k w_{kj} f(x_k) = \sum_k w_{kj} f'(x_k) \Delta x_k$$

Bây giờ chúng ta có thể thể hiện quy tắc cập nhật trọng số hoàn chỉnh cho trọng số dẫn đầu ra nút:

( -axi 2 oj ( ) kỳ vọng )θ ( (( )) lnθ - ( ) ) nj

Do đó, chúng tôi cập nhật trọng số tùy thuộc vào giá trị nhập tương ứng với đó trọng số, sự khác biệt trong đầu ra và giá trị mong đợi, và đầu ra của đạo hàm của hàm kích hoạt với tất cả đầu vào và trọng số. Lưu ý rằng chúng tôi thêm một tiêu cực đầu và một thuật ngữ alpha. Cái trước đảm bảo rằng chúng ta di chuyển theo hướng tiêu cực lõi và lõi sau xác định tốc độ chúng ta di chuyển theo hướng đó.

Điều này sẽ cho bạn cảm giác về cách các trọng số được cập nhật vào lớp đầu ra và các chức năng cập nhật của lớp bên trong tuân theo cùng một logic. Nhưng chúng tôi phải sử dụng quy luật dây chuyền để tìm ra sự đóng góp của giá trị đầu ra ở bên trong đó nút đến lõi chung của mạng, nghĩa là chúng ta phải biết tốc độ thay đổi của đầu vào/đầu ra trên đường dẫn từ nút được đề cập đến nút đầu ra.

Chỉ sau đó, tốc độ thay đổi của lõi đầu ra mới có thể được đánh giá cho sự thay đổi delta trong trọng lượng của một nút bên trong. Điều này dẫn đến hình thức lan truyền ngược đầy đủ.<sup>1</sup>

#### 6.4.5 MLNN trong scikit-learning

Cho rằng bây giờ bạn đã hiểu perceptron đa lớp và lý thuyết dang sau đào tạo bằng cách sử dụng backpropagation, hãy quay lại Python để viết mã ví dụ. Vì không có triển khai MLP nào trong scikit-learning nên chúng tôi sẽ sử dụng PyBrain<sup>2</sup>. PyBrain tập trung đặc biệt vào việc xây dựng và đào tạo mạng lưới thần kinh. Các danh sách sau đây cung cấp cho bạn đoạn mã đầu tiên cần thiết để xây dựng một hệ thống thần kinh mạng tương đương với mạng được trình bày trong hình 6.12. Vui lòng tham khảo mã đầy đủ được phân phối cùng với cuốn sách này cho các lần nhập liên quan cần thiết để chạy mã này.

##### Lịch kê 6.6 Xây dựng một perceptron đa lớp bằng PyBrain

```
#Tạo mô-đun mạng
mạng = FeedForwardNetwork()
inl = LinearLayer(2)
hidl = SigmoidLayer(2)
outl = LinearLayer(1)
b = BiasUnit()
```

Trong danh sách 6.6, trước tiên chúng ta tạo một đối tượng `FeedForwardNetwork`. Chúng tôi cũng tạo một lớp đầu vào (`inl`) và lớp đầu ra (`outl`) và một lớp ẩn (`hidl`) của các nơron. lưu ý rằng của chúng tôi các lớp đầu vào và đầu ra sử dụng chức năng kích hoạt vanilla (ngưỡng ở 0), trong khi lớp ẩn của chúng tôi sử dụng chức năng kích hoạt sigmoid vì lý do đào tạo, như chúng tôi đã thảo luận trước đó. Cuối cùng, chúng tôi tạo ra một đơn vị thiên vị. Chúng ta chưa hoàn toàn có một mạng lưới thần kinh, bởi vì chúng tôi đã không kết nối các lớp. Đó là những gì chúng tôi làm trong danh sách tiếp theo.

<sup>1</sup> Rumelhart và cộng sự, "Học biểu diễn bằng các lõi lan truyền ngược," 533-36.

<sup>2</sup> Tom Schaul, và cộng sự, "PyBrain," Tạp chí Nghiên cứu Máy học 11 (2010): 743-46.

**Lиет kê 6.7 Xây dựng một perceptron đa lớp bằng PyBrain (2)**

```
#Tạo kết nối
in_to_h = FullConnection(inl, hid1)
h_to_out = FullConnection(hid1, outl)
bias_to_h = FullConnection(b,hid1)
bias_to_out = FullConnection(b,outl)

#Thêm mô-đun vào mạng
net.addInputModule (inl)
net.addModule(hid1);
net.addModule(b)
net.addOutputModule(outl)

#Thêm kết nối vào mạng và sắp xếp
net.addConnection(in_to_h)
net.addConnection(h_to_out)
net.addConnection(bias_to_h)
net.addConnection(bias_to_out)
net.sortModules()
```

Trong danh sách 6.7, bây giờ chúng ta tạo các đối tượng kết nối và thêm các neuron (mô-đun) đã tạo trước đó của chúng ta và các kết nối của chúng vào **đối tượng FeedForwardNetwork**. gọi `sortModules()` hoàn thành việc khởi tạo mạng.

Trước khi tiếp tục, chúng ta hãy dành một chút thời gian để đi sâu vào **đối tượng FullConnection**. Ở đây chúng tôi tạo bốn phiên bản của đối tượng để chuyển đến đối tượng mạng. Chữ ký của các hàm tạo này có hai lớp và bên trong đối tượng tạo ra một kết nối giữa mọi nơ-ron trong lớp của các tham số đầu tiên và mọi nơ-ron trong lớp tham số đầu tiên.

lớp thứ hai. Phương thức cuối cùng sắp xếp các mô-đun trong **đối tượng** công việc FeedForwardNet và thực hiện một số khởi tạo nội bộ.

Bây giờ chúng ta có một mạng thần kinh tương đương với hình 6.11, chúng ta cần tìm hiểu nó trọng lượng! Để làm điều này, chúng tôi cần một số dữ liệu. Danh sách tiếp theo cung cấp mã để thực hiện việc này, và phần lớn được sao chép từ tài liệu PyBrain.1

**Lиет kê 6.8 Huấn luyện mạng thần kinh của chúng ta**

```
d = [(0,0),
      (0,1),
      (1,0),
      (1,1)]
#lớp mục tiêu
c = [0,1,1,0]
data_set = SupervisedDataSet(2, 1) # 2 đầu vào, 1 đầu ra
hạt giống ngẫu nhiên()
cho tôi trong xrange(1000):
    r = ngẫu nhiên.randint(0,3)
    data_set.addSample(d[r],c[r])
```

Tạo tập dữ liệu và các mục tiêu được liên kết phản ánh hàm XOR.

Tạo đối tượng PyBrain SupervisedDataSet trống.

Lấy mẫu ngẫu nhiên bốn điểm dữ liệu 1000 lần và thêm vào tập huấn luyện.

<sup>1</sup> Khởi động nhanh PyBrain, ngày 12 tháng 11 năm 2009, <http://pybrain.org/docs/index.html#quickstart> (truy cập ngày 22 tháng 12 năm 2015).

```

backprop_trainer = \
BackpropTrainer(net, data_set, learningrate=0.1)

cho i trong xrange(50):
    err = backprop_trainer.train() print
    "Iter. %d, err.: %.5f" % (i, err)

```

Tạo một đối tượng huấn luyện  
lên truyền ngược mới với mạng,  
tập dữ liệu và tốc độ học tập.

Thực hiện lan truyền ngược 50 lần bằng  
cách sử dụng cùng 1000 điểm dữ liệu.  
In lỗi sau mỗi lần lặp.

Như bạn đã biết từ phần 6.4.4, lan truyền ngược đi qua không gian trọng số để đạt được cực  
tiều về sai số giữa các số hạng đầu ra và công suất đầu ra dự kiến. Mỗi lệnh gọi đến train()  
đều làm cho các trọng số được cập nhật để mạng nơ-ron thể hiện tốt hơn chức năng tạo dữ liệu.  
Điều này có nghĩa là chúng tôi có thể sẽ cần một lượng dữ liệu hợp lý (đối với ví dụ XOR của  
chúng tôi, bốn điểm dữ liệu sẽ không bị cắt giảm!) cho mỗi lệnh gọi huấn luyện(). Để giải  
quyết vấn đề này, chúng tôi sẽ tạo ra nhiều điểm dữ liệu được rút ra từ bản phân phối XOR và  
sử dụng những điểm này để đào tạo mạng của chúng tôi bằng cách truyền ngược. Như bạn sẽ thấy,  
các lệnh gọi tiếp theo tới train() đã giảm thành công lỗi giữa đầu ra mạng và mục tiêu đã chỉ  
định. Số lần lặp chính xác cần thiết để tìm cực tiểu toàn cục sẽ phụ thuộc vào nhiều yếu tố,  
một trong số đó là tốc độ học. Điều này kiểm soát tốc độ cập nhật trọng số ở mỗi khoảng thời  
gian đào tạo. Các tỷ lệ nhỏ hơn sẽ mất nhiều thời gian hơn để hội tụ, nghĩa là tìm ra cực  
tiểu toàn cầu, nhưng chúng ít có khả năng dẫn đến các mô hình dưới mức tối ưu. Chúng ta hãy  
xem nhanh kết quả được tạo bởi danh sách 6.8 và sử dụng kết quả này để minh họa khái niệm này:

```

Lặp lại 0, lỗi: 0,1824 Lặp lại 1, lỗi:
0,1399 Lặp lại 2, lỗi: 0,1384 Lặp lại 3,
lỗi: 0,1406 Lặp lại 4, lỗi: 0,1264 Lặp
lại 5, lỗi: 0,1333 Lặp lại 6, lỗi: 0,1398
Lặp lại 7, lỗi: 0,1374 Lặp lại 8 , lỗi:
0,1317 Lặp lại 9, lỗi: 0,1332

```

Như bạn sẽ thấy, các cuộc gọi liên tiếp làm giảm lỗi của mạng. Chúng tôi biết rằng có ít nhất  
một giải pháp tồn tại, nhưng lan truyền ngược không được đảm bảo để tìm ra giải pháp này.  
Trong một số trường hợp nhất định, lỗi sẽ giảm và sẽ không thể cải thiện thêm nữa hoặc lỗi  
có thể bị trả lại giữa các giải pháp dưới mức tối ưu (hoặc cục bộ). Thuật toán có thể gặp khó  
khăn và không thể tìm thấy cực tiểu toàn cầu, nghĩa là lỗi thấp nhất có thể.

Vì kết quả này phụ thuộc vào giá trị ban đầu của các trọng số nên chúng tôi không thể nói  
liệu ví dụ của bạn có hội tụ nhanh hay không, vì vậy hãy thử chạy kết quả này một vài lần.  
Ngoài ra, hãy thử trải nghiệm với tỷ lệ học tập từ danh sách 6.8. Bạn có thể tạo ra tỷ lệ lớn  
đến mức nào trước khi thuật toán bị mắc kẹt trong các giải pháp cục bộ trong hầu hết thời  
gian? Trong thực tế, việc lựa chọn tốc độ đào tạo luôn là sự đánh đổi giữa việc tìm ra các  
giải pháp dưới mức tối ưu và tốc độ, vì vậy bạn muốn chọn tốc độ lớn nhất mang lại cho bạn  
câu trả lời đúng. Thử nghiệm điều này cho đến khi bạn còn lại một mạng đã hội tụ với sai số bằng không.

#### 6.4.6 MLP đã học

Trong ví dụ trước, chúng tôi đã tạo MLP bằng gói PyBrain và đào tạo một perceptron nhiều lớp để bắt chước chức năng XOR . Với điều kiện lỗi của bạn trong đầu ra trước đó bằng 0, bạn sẽ có thể tự mình theo dõi phần này người mẫu! Đầu tiên, hãy tham vấn mô hình của chúng ta để có được các trọng số của mạng, tương ứng với hình 6.11. Đoạn mã sau chỉ cho bạn cách thực hiện.

##### Liệt kê 6.9 Lấy trọng số của mạng thần kinh được đào tạo của bạn

```
#print net.params
print "[w(x_1,j=1),w(x_2,j=1),w(x_1,j=2),w(x_2,j=2)]: " print "[w(j=1,
j=3),w(j=2,j=3)]: "+str(h_to_out.params)
print "[w(x_b,j=1),w(x_b,j=2)]: "+str(bias_to_h.params)
in "[w(x_b,j=3)": "+str(bias_to_out.params)

> [w(x_1,j=1),w(x_2,j=1),w(x_1,j=2),w(x_2,j=2)]: [-2.32590226 2.25416963 -
2.74926055 2.64570441]
> [w(j=1,j=3),w(j=2,j=3)]: [-2.57370943 2.66864851]
> [w(x_b,j=1),w(x_b,j=2)]: [ 1.29021983 -1.82249033]
> [w(x_b,j=3)]:[ 1.6469595]
```

Đầu ra từ việc thực hiện liệt kê 6.9 cung cấp đầu ra được đào tạo của tôi cho một neuron. Kết quả của bạn có thể thay đổi, tuy nhiên. Điều quan trọng là hành vi của mạng là chính xác. Bạn có thể kiểm tra điều này bằng cách kích hoạt mạng với đầu vào và kiểm tra xem đầu ra có như mong đợi không. Quan sát danh sách tiếp theo.

##### Liệt kê 6.10 Kích hoạt mạng thần kinh của bạn

```
print "Đang kích hoạt 0,0. Đầu ra: " print "Đang      + str(net.activate([0,0)))
kích hoạt 0,1. Đầu ra: " print "Đang kích hoạt      + str(net.activate([0,1]))
1,0. Đầu ra: " print "Đang kích hoạt 1,1. Đầu      + str(net.activate([1,0]))
ra: "      + str(net.activate([1,1]))

> Kích hoạt 0,0. Đầu ra: [ -1.33226763e-15]
> Kích hoạt 0,1. Đầu ra: [ 1.]
> Kích hoạt 1,0. Đầu ra: [ 1.]
> Kích hoạt 1,1. Đầu ra: [ 1.55431223e-15]
```

Trong danh sách 6.10, bạn có thể thấy rằng đầu ra của mạng được đào tạo của chúng tôi rất gần với 1 cho những mẫu đó sẽ dẫn đến một giá trị tích cực. Ngược lại, đầu ra rất gần bằng 0 đối với những mẫu sẽ dẫn đến đầu ra âm. Nói chung, các mẫu thử nghiệm dương tính phải có đầu ra lớn hơn 0,5 và các mẫu thử nghiệm âm tính nên cung cấp đầu ra nhỏ hơn 0,5. Để đảm bảo rằng bạn hiểu đầy đủ điều này mạng, hãy thử sửa đổi các giá trị đầu vào và theo dõi chúng thông qua mạng trong hỗ trợ bảng tính nội dung phân phối với cuốn sách này.

## 6.5 Đi sâu hơn: từ mạng lưới thần kinh đa lớp đến học sâu

Trong nhiều lĩnh vực nghiên cứu, tiến bộ được thực hiện phù hợp và bắt đầu. Các khu vực có thể trở nên cũ kỹ trong một thời gian thời gian và sau đó trải qua một cơn sốt nhanh chóng, thường được kích hoạt bởi một tiến bộ cụ thể hoặc khám phá. Mô hình này không khác gì trong lĩnh vực mạng lưới thần kinh và chúng tôi thật may mắn ở ngay giữa một số tiến bộ thực sự thú vị, hầu hết trong số đó đã được nhóm dưới sự bảo trợ của học tập sâu. Tôi muốn chia sẻ một vài trong số này với bạn bây giờ trước khi đi sâu vào ví dụ đơn giản nhất về mạng sâu mà chúng ta có thể. Vậy tại sao mạng lưới thần kinh có nóng trở lại không? Vâng, đó là một chút của một cơn bão hoàn hảo.

Đầu tiên, có nhiều dữ liệu hơn bao giờ hết. Những ga khổng lồ internet lớn đã truy cập vào kho lưu trữ dữ liệu hình ảnh khổng lồ có thể được tận dụng để làm những điều thú vị. Một ví dụ mà bạn có thể đã nghe nói đến là bài báo năm 2012 của Google, nơi họ đào tạo một mạng chín lớp với 10 triệu hình ảnh được tải xuống từ internet để nhận dạng đại diện trung gian mà không cần dán nhãn, công khai nhất là một khuôn mặt mèo! Điều này làm tăng thêm sức nặng cho giả thuyết rằng nhiều dữ liệu hơn sẽ đánh bại một thuật toán thông minh hơn.<sup>2</sup> Một thành tích như vậy sẽ không thể đạt được chỉ một vài năm trước đó.

Bước tiến thứ hai là bước nhảy vọt về kiến thức lý thuyết. Mặc cho đến gần đây những tiến bộ của Geoffrey Hinton và cộng tác viên mà cộng đồng hiểu rằng các mạng học sâu có thể được đào tạo một cách hiệu quả bằng cách coi mỗi lớp là Máy Boltzmann bị hạn chế (RBM).<sup>3,4</sup> Thật vậy, nhiều mạng học sâu hiện đã được xây dựng bằng cách xếp chồng các RBM-thêm về những điều này trong giây lát. Yann Le Cun, Yoshua Bengio, và những người khác đã đạt được nhiều tiến bộ lý thuyết hơn nữa trong lĩnh vực này, và tôi giới thiệu bạn đến một xem xét công việc của họ để đạt được cái nhìn sâu sắc hơn.<sup>5</sup>

### 6.5.1 Máy Boltzmann bị hạn chế

Trong phần này, chúng ta sẽ xem xét các Máy Boltzmann bị Hạn chế (RBM). Hơn cụ thể, chúng ta sẽ xem xét một hướng vị cụ thể của RBM được gọi là Bernoulli RBM (BRBM). Tốt hiểu tại sao đây là một trường hợp đặc biệt của RBM trong giây lát. Nói chung, RBM được nhắc đến trong bối cảnh học sâu, bởi vì chúng là những máy học tính năng tốt. Do đó, chúng có thể được sử dụng trong một mạng sâu hơn để tìm hiểu các biểu diễn tính năng, với đầu ra của chúng được sử dụng làm đầu vào cho các RBM khác hoặc một tri giác đa lớp (MLP). Hãy nhớ lại phần 6.1 và ví dụ về đề xuất ô tô của chúng tôi. Cho đến nay, chúng tôi đã đã dành khá nhiều thời gian để đưa tin về MLP nói chung; bây giờ chúng ta phải khám phá ra cơ chế tự động khía cạnh trích xuất tính năng của học sâu!

<sup>1</sup> Quoc V. Le, et al., "Xây dựng các tính năng cấp cao sử dụng học tập không giám sát quy mô lớn," ICML 2012: 29 Hội thảo quốc tế về học máy (Edinburgh: ICML, 2012):1.

<sup>2</sup> Pedro Domingos, "Một vài điều hữu ích cần biết về học máy," Truyền thông của ACM, ngày 1 tháng 10 năm 2012: 78-87.

<sup>3</sup> Miguel A. Carreira-Perpiñán và Geoffrey Hinton, "On Contrastive Divergence Learning," (NJ: Society for Trí tuệ nhân tạo và Thống kê, 2005), 33-40.

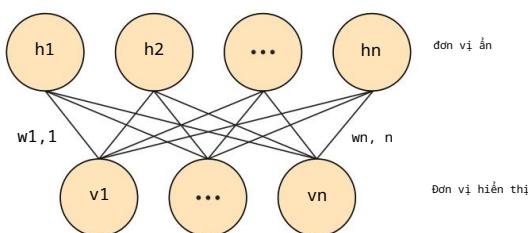
<sup>4</sup> GE Hinton và RR Salakhutdinov, "Reducing the Dimensionality of Data with Neural Networks," Science, 28 tháng 7 năm 2006: 504-507.

<sup>5</sup> Yann LeCun, Yoshua Bengio và Geoffrey Hinton, "Học sâu," Nature 521 (tháng 5 năm 2015): 436-44.

Để làm điều này, chúng tôi cũng sẽ sử dụng một ví dụ từ tài liệu scikit-learning.<sup>1</sup> Ví dụ này sử dụng BRBM để trích xuất các tính năng từ bộ dữ liệu chữ số scikit-learning và sau đó sử dụng hồi quy logistic để phân loại các mục dữ liệu với các tính năng đã học. Sau khi xem qua ví dụ này, chúng ta sẽ đề cập đến cách bạn có thể thực hiện mạng sâu hơn và đi sâu hơn vào lĩnh vực học tập sâu đang phát triển. Trước khi bắt đầu, hãy đảm bảo rằng bạn hiểu những điều cơ bản—vậy thì sao? BRBM là gì?

#### 6.5.2 Máy Boltzmann hạn chế Bernoulli

Nói chung, một RBM là một biểu đồ hai bên trong đó các nút trong mỗi phân vùng hoàn toàn được kết nối với các nút trong phân vùng khác. Khía cạnh hạn chế đến từ thực tế là các nút hiển thị chỉ có thể được kết nối với các nút ẩn và ngược lại. Các Bernoulli RBM hạn chế mỗi nút hơn nữa là nhị phân. Hình 6.13 cung cấp một cái nhìn tổng quan về đồ họa của một RBM.



Hình 6.13 Tổng quan về đồ họa của Máy Boltzmann bị hạn chế. RBM là một biểu đồ hai bên giữa các đơn vị ẩn và hiển thị, với mỗi phần tử của mỗi phân vùng được kết nối đầy đủ với các đơn vị khác trong phân vùng đối diện. Chúng ta sẽ sử dụng  $H$  để chỉ vectơ bao gồm các đơn vị ẩn và  $V$  để chỉ vectơ bao gồm các đơn vị hiển thị.

Lưu ý rằng mỗi nút có thể có trọng số sai lệch liên quan, trọng số này không được trình bày ở đây để đơn giản.

Nói chung, số lượng đơn vị hiển thị được xác định bởi vấn đề; đối với một vấn đề phân loại nhị phân, bạn có thể có hai. Bằng cách tăng số lượng giá trị ẩn, bạn tăng khả năng của RBM để mô hình hóa các chức năng phức tạp, nhưng điều này xảy ra ở giá của overfitting. Hinton<sup>2</sup> cung cấp công thức chọn số ẩn đơn vị phụ thuộc vào độ phức tạp của dữ liệu của bạn và số lượng mẫu đào tạo.

Tương tự như các MLP có chức năng kích hoạt logistic, xác suất của một kích hoạt đơn vị có thể nhìn thấy với giá trị của các biến ẩn là

$$P(v_i | \theta) = \frac{1}{1 + \bar{H}(\sum_j w_{ij} h_j + b_i)}$$

<sup>1</sup> scikit-learning, Các tính năng của Máy Boltzmann bị hạn chế để phân loại chữ số, ngày 01 tháng 1 năm 2014, [http://scikit-learn.org/stable/auto\\_examples/neural\\_networks/plot\\_rbm\\_logistic\\_classification.html#example-neural-networks-plot-rbm-logistic-classification-py](http://scikit-learn.org/stable/auto_examples/neural_networks/plot_rbm_logistic_classification.html#example-neural-networks-plot-rbm-logistic-classification-py) (truy cập ngày 22 tháng 12 năm 2015).

<sup>2</sup> Geoffrey Hinton, "Hướng dẫn thực hành để đào tạo các máy Boltzmann bị hạn chế" trong Mạng thần kinh: Thủ thuật the Trade, biên tập bởi Grégoire Montavon, GB Orr, và KR Müller (NY: Đại học Toronto, 2012) 599–619.

trong đó  $\theta$  là hàm hậu cần. Xác suất của các giá trị ẩn được đưa ra tương tự. RBM được đào tạo thành các mô hình dựa trên năng lượng để đo lường sự thỏa thuận giữa trạng thái ẩn và trạng thái hiển thị.

$$EVH(\cdot) = - \sum_{j \in j_{\text{vis}}} (\log p_j + \sum_{i \in i_{\text{hid}}} \theta_{ij} h_i)$$

Giá trị này giảm khi có nhiều thỏa thuận hơn giữa các nút ẩn và những cái nhìn thấy được; do đó, giảm kết quả năng lượng trong các cấu hình dễ chấp nhận hơn đã cho dữ liệu huấn luyện. Chúng ta có thể liên hệ hàm năng lượng này với xác suất như sau

$$PVH(\cdot) = e^{-E(\cdot)} / Z$$

trong đó  $Z$  là hệ số chuẩn hóa bằng  $(\sum_{\text{all states}} e^{-E(\cdot)})$ . Tức là chúng tôi đánh giá lựa chọn trọng số hiện tại bằng cách chuẩn hóa trên tất cả các hoán vị ẩn và hiển thị nêu các khả năng. Để huấn luyện RBM, chúng ta cần tối đa hóa khả năng  $P(V)$  hoặc tương tự như khả năng đăng nhập. Lấy log của  $P(V, H)$  ta viết được

$$\begin{aligned} \text{Nhật ký}(P)VH(\cdot) &= \text{Nhật ký}(\text{tử}) - E(\cdot) \\ \text{nhật ký}(P)V(\cdot) &= H \log e^{-(E(\cdot))} - \text{LogZ}(W(\cdot)) \end{aligned}$$

Do đó, đối với một dữ liệu hiển thị nhất định, chúng ta có thể tối đa hóa khả năng của nó bằng cách chọn trọng số tối đa hóa thỏa thuận đối với các biến ẩn có thể (thuật ngữ đầu tiên) và giảm thiểu hệ số chuẩn hóa đánh giá lựa chọn tყობები trên  $E(\cdot)$ , tất cả các giá trị hiện và ẩn có thể có. Theo một nghĩa nào đó, bạn có thể coi cái trước là cố gắng để kích hoạt mô hình khi cần và mô hình sau cung cấp sự tối ưu hóa này với ngữ cảnh của dữ liệu huấn luyện. Tối đa hóa các trọng số lựa chọn đầu tiên mà trên tất cả các giá trị ẩn có thể có sẽ có khả năng tạo ra giá trị hiển thị tối đa, trong khi giảm thiểu thứ hai đảm bảo rằng mô hình vẫn đại diện cho dữ liệu.

Nếu bạn còn nhớ từ trước đó, chúng ta cần phân biệt chức năng trước đó với đối với trọng lượng để tìm hướng mà chúng ta nên di chuyển trọng lượng để tối đa hóa khả năng. Nếu bạn làm theo lý thuyết,<sup>1</sup> bạn sẽ thấy rằng điều đó là có thể để tính đạo hàm của số hạng đầu tiên một cách chính xác, nhưng đối với số hạng thứ hai, chúng ta cần dựa vào trên một thuật toán lặp để ước tính nó. Nói tóm lại, sẽ phức tạp hơn để thực hiện đào tạo, nhưng trong thực tế nó có thể. Bạn sẽ thấy làm thế nào trong phần tiếp theo!

### 6.5.3 RBMS đang hoạt động

Trong phần này, chúng ta sẽ sử dụng một phiên bản sửa đổi của bài toán phân loại hậu cần được trình bày trong tài liệu scikit-learning.<sup>2</sup> Phải ghi công đầy đủ cho Dauphin,

<sup>1</sup> Kevin Swersky, Bo Chen, Ben Marlin, và N de Freitas, "Hướng dẫn về thuật toán xấp xỉ ngẫu nhiên để đào tạo Máy Boltzmann bị hạn chế và Mạng lưới niềm tin sâu sắc," Lý thuyết thông tin và ứng dụng Hội thảo (ITA), 2010 (San Diego: ITA, 2010) 1-10.

<sup>2</sup> scikit-learn, "Các tính năng của Máy Boltzmann bị hạn chế để phân loại chữ số," [http://scikit-learn.org/stable/auto\\_examples/neural\\_networks/plot\\_rbm\\_logistic\\_classification.html#example-neural-networks-plot\\_rbm-logistic-classification-py](http://scikit-learn.org/stable/auto_examples/neural_networks/plot_rbm_logistic_classification.html#example-neural-networks-plot_rbm-logistic-classification-py) (truy cập ngày 22 tháng 12 năm 2015).

Niculae, và Synnaeve cho ví dụ minh họa này, và chúng ta sẽ không đi quá xa tài liệu của họ ở đây. Như trong các ví dụ trước, chúng tôi sẽ bỏ qua khái niệm và tập trung vào mã. Danh sách đầy đủ có thể được tìm thấy trong nội dung hỗ trợ.

#### Liệt kê 6.11 Tạo tập dữ liệu của bạn

```
chữ số = datasets.load_digits()
X = np.asarray(digits.data, 'float32')
X, Y = nudge_dataset(X, chữ số.mục tiêu)
X = (X - np.min(X, 0)) / (np.max(X, 0) + 0,0001) # 0-1 chia tỷ lệ X_train, X_test, Y_train, Y_test =
train_test_split(X,
Y,test_size=0.2,random_state=0)
```

Điều đầu tiên chúng tôi làm là tải tập dữ liệu vào, nhưng chúng tôi thực sự làm nhiều hơn thế. Từ tập dữ liệu ban đầu, chúng tôi tạo thêm các mẫu nhân tạo bằng cách dịch chuyển tuyến tính một pixel vào tập dữ liệu, chuẩn hóa từng pixel sao cho mỗi giá trị pixel nằm trong khoảng từ 0 đến 1. Vì vậy, đối với mỗi hình ảnh được gắn nhãn, bốn hình ảnh khác được tạo ra-lần lượt dịch chuyển lên, xuống, phải và trái-mỗi hình ảnh có cùng một nhãn, tức là số mà hình ảnh đại diện. Điều này cho phép đào tạo để tìm hiểu các biểu diễn dữ liệu tốt hơn bằng cách sử dụng một tập dữ liệu nhỏ như vậy, các biểu diễn cụ thể ít phụ thuộc vào tập lệnh được tập trung trong hình ảnh. Điều này đạt được bằng cách sử dụng **chức năng nudge\_dataset**, được xác định trong danh sách sau.

#### Liệt kê 6.12 Tạo dữ liệu nhân tạo

```
def nudge_dataset(X, Y):
    """
    Điều này tạo ra một tập dữ liệu lớn hơn 5 lần so với tập dữ liệu ban đầu, bằng cách di chuyển các
    hình ảnh 8x8 trong X xung quanh 1px sang trái, phải, xuống, lên
    """

    direction_vectors = [[[0, 1, 0],[0, 0, 0],[0, 0, 0]],
                         [[0, 0, 0],[1, 0, 0],[0, 0, 0]], [[0, 0, 0],[0, 0, 0],
                         1],[0, 0, 0]], [[0, 0, 0],[0, 0, 0],[0, 1, 0]]]

    thay đổi = \
        lambda x, w: convolve(x.reshape((8, 8)), mode='constant', weights=w).ravel()

    X = np.concatenate([X] + \
        [np.apply_along_axis(shift, 1, X, vec-to) cho vec-to trong direction_vectors])

    Y = np.concatenate([Y trả về X, Y _ trong phạm vi (5)], trực = 0)
```

Với dữ liệu này, giờ đây thật đơn giản để tạo một quy trình quyết định bao gồm RBM, theo sau là hồi quy logistic. Danh sách tiếp theo trình bày mã để thiết lập quy trình này và huấn luyện mô hình.

### Liệt kê 6.13 Thiết lập và huấn luyện đường ống RBM/LR

```
# Mô hình chúng tôi sẽ sử dụng
logistic = linear_model.LogisticRegression() rbm =
BernoulliRBM(random_state=0, verbose=True)

classifier = Pipeline(steps=[('rbm', rbm), ('hậu cần', logistic)])

#####
# Đào tạo

# Siêu tham số. Chúng được đặt bằng cách xác thực chéo, # sử dụng GridSearchCV. Ở đây
chúng tôi không thực hiện xác thực chéo để # tiết kiệm thời gian.

rbm.learning_rate = 0.06 rbm.n_iter =
20 # Nhiều thành phần
hơn có xu hướng cho hiệu suất dự đoán tốt hơn, nhưng lớn hơn # thời gian điều chỉnh rbm.n_components = 100 logistic.C
= 6000,0

# Đào tạo RBM-Logistic Pipeline classifier.fit(X_train,
Y_train)

# Đào tạo Hồi quy logistic logistic_classifier
= linear_model.LogisticRegression(C=100.0) logistic_classifier.fit(X_train, Y_train)
```

Mã này được lấy trực tiếp từ Scikit-learning<sup>1</sup> và có một số điều quan trọng cần lưu ý. Các siêu tham số, nghĩa là các tham số của RBM, đã được chọn đặc biệt cho tập dữ liệu để cung cấp một ví dụ minh họa mà chúng ta có thể thảo luận.

Thông tin chi tiết có thể được tìm thấy trong tài liệu gốc.

Bạn sẽ thấy rằng ngoài điều này, mã làm được rất ít. Một quy trình phân loại được thiết lập bao gồm RBM, theo sau là một công cụ phân loại hồi quy logistic, cũng như một công cụ phân loại hồi quy logistic độc lập để so sánh. Trong một phút, bạn sẽ thấy hai cách tiếp cận này hoạt động như thế nào. Danh sách sau đây cung cấp mã để thực hiện việc này.

### Liệt kê 6.14 Đánh giá đường ống RBM/LR

```
print("Hồi quy logistic sử dụng các tính năng RBM:\n%s\n" % (
    số liệu.classification_report( Y_test,
        classifier.predict(X_test)))))

print("Hồi quy logistic sử dụng các tính năng pixel thô:\n%s\n" % (
    số liệu.classification_report( Y_test,
        logistic_classifier.predict(X_test))))
```

Đầu ra của điều này cung cấp một bản tóm tắt chi tiết về hai cách tiếp cận và bạn sẽ thấy rằng đường dẫn RBM/LR vượt xa cách tiếp cận LR cơ bản về độ chính xác,

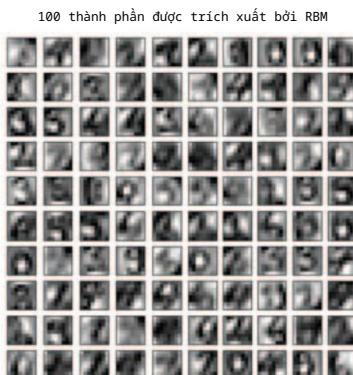
<sup>1</sup> Scikit-learning, "Bernoulli RBM."

nhớ lại, và điểm số f1. Nhưng tại sao lại thế này? Nếu chúng ta vẽ các thành phần ẩn của RBM, chúng ta nên bắt đầu hiểu tại sao. Danh sách tiếp theo cung cấp mã để thực hiện việc này và hình 6.14 cung cấp tổng quan bằng đồ họa về các thành phần ẩn trong RBM của chúng ta.

#### Liet kê 6.15 Biểu diễn đồ thị các đơn vị ẩn

```
plt.figure(figsize=(4.2, 4))
đối với tôi, tính theo kiểu liệt kê (rbm.components_):
    #in(i)
    #print(comp)
    plt.subplot(10, 10, i + 1)
    plt.imshow(comp.reshape((8, 8)),
               cmap=plt.cm.gray_r, interpolation='gần nhất')
    plt.xticks(())
    plt.yticks(())

plt.suptitle('100 thành phần được trích xuất bởi RBM', fontsize=16)
plt.subplots_adjust(0,08, 0,02, 0,92, 0,85, 0,08, 0,23)
plt.show()
```



Hình 6.14 Biểu diễn đồ họa về trọng số giữa các đơn vị ẩn và hiển thị trong RBM của chúng tôi. Mỗi ô vuông đại diện cho một đơn vị ẩn duy nhất và 64 giá trị thang độ xám bên trong biểu thị trọng số từ giá trị ẩn đó cho tất cả các đơn vị hiển thị. Theo một nghĩa nào đó, điều này chỉ ra mức độ ẩn giấu đó biến có thể nhận dạng hình ảnh giống như hình ảnh được trình bày.

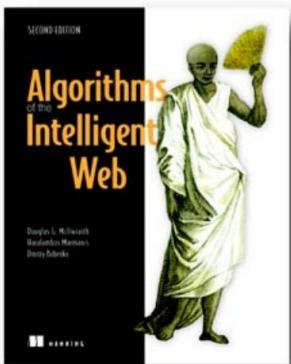
Trong RBM của chúng tôi, chúng tôi có 100 nút ẩn và 64 đơn vị hiển thị, vì đây là kích thước của những hình ảnh đang được sử dụng. Mỗi ô vuông trong hình 6.14 là một diễn giải thang độ xám của trọng số giữa thành phần ẩn đó và từng đơn vị hiển thị. Theo một nghĩa nào đó, mỗi thành phần ẩn đơn có thể được coi là nhận dạng hình ảnh như đã cho trước đó. bên trong đường ống, mô hình hồi quy logistic sau đó sử dụng 100 xác suất kích hoạt ( $P_{hi} = \sigma(v_i)$  cho mỗi  $i$ ) làm đầu vào của nó; do đó, thay vì thực hiện hồi quy logistic trên 64 pixel thô, nó thực hiện trên 100 đầu vào, mỗi đầu vào có giá trị cao khi đầu vào trông gần giống với đầu vào được cung cấp trong hình 6.14. Quay trở lại phần đầu tiên của chương này, bây giờ bạn có thể thấy rằng chúng tôi đã tạo một mạng đã tự động học một số biểu diễn trung gian của các số, sử dụng một RBM. Về bản chất, chúng tôi đã tạo một lớp duy nhất của mạng sâu! Hãy tưởng tượng bây giờ những gì có thể đạt được với các mạng sâu hơn và nhiều lớp RBM để tạo ra nhiều đại diện trung gian!

## 6.6 Kết luận

Trong chương này

Chúng tôi đã cung cấp cho bạn một chuyến tham quan ngắn gọn về mạng lưới thần kinh và mối quan hệ của chúng để học sâu. Bắt đầu với mô hình mạng thần kinh đơn giản nhất, MCP mô hình, chúng tôi chuyển sang perceptron và thảo luận về mối quan hệ của nó với hồi quy logistic.

Chúng tôi thấy rằng không thể biểu diễn các hàm phi tuyến tính bằng một perceptron, nhưng điều đó là có thể nếu chúng ta tạo ra các perceptron đa lớp (MLP). Chúng ta đã thảo luận về cách các MLP được huấn luyện thông qua lan truyền ngược –và việc áp dụng các hàm kích hoạt khả vi–và cung cấp cho bạn một ví dụ theo đó một hàm phi tuyến tính được học bằng cách sử dụng lan truyền ngược trong PyBrain. Chúng ta đã thảo luận về những tiến bộ gần đây trong lĩnh vực học sâu, cụ thể là xây dựng nhiều lớp mạng có thể học các biểu diễn trung gian của dữ liệu. Chúng tôi tập trung vào một mạng như vậy được gọi là Boltzmann hạn chế Máy và chúng tôi đã chỉ ra cách bạn có thể xây dựng mạng sâu đơn giản nhất qua tập dữ liệu chữ số, sử dụng một RBM duy nhất và bộ phân loại hồi quy logistic.



Có thông tin chi tiết vô giá bị mắc kẹt trong vô số dữ liệu mà người dùng web để lại khi họ tương tác với các trang và ứng dụng của bạn. Bạn có thể mở khóa những hiểu biết sâu sắc đó bằng cách sử dụng các thuật toán thông minh như những thuật toán đã giúp Facebook, Google, Twitter và Microsoft trở thành một trong những công ty khai thác mẫu dữ liệu web khổng lồ. Cải thiện tìm kiếm, phân loại dữ liệu và các kỹ thuật đối sánh mẫu thông minh khác có thể mang lại cho bạn lợi thế to lớn khi bạn cần hiểu và tương tác với người dùng của mình.

[Các thuật toán của Web thông minh, Phiên bản thứ hai](#) dạy các phương pháp quan trọng nhất để phân tích dữ liệu web theo thuật toán, cho phép bạn tạo các ứng dụng máy học của

riêng mình để xử lý, trộn và sắp xếp dữ liệu được thu thập từ người dùng, ứng dụng web, cảm biến và nhật ký trang web. Trong ấn bản được sửa đổi hoàn toàn này, bạn sẽ xem xét các thuật toán thông minh qua lăng kính máy học, với các ví dụ về mã cho bạn biết cách trích xuất giá trị từ dữ liệu của chúng. Các khái niệm học máy chính được giải thích và giới thiệu với các ví dụ về mã trong scikit-learn của Python. Cuốn sách này hướng dẫn bạn về bộ máy cơ bản và các thuật toán thông minh để nắm bắt, lưu trữ và cấu trúc các luồng dữ liệu đến từ web. Bạn sẽ khám phá các công cụ để xuất từ ví dụ về đề xuất phim Netflix và đi sâu vào phân loại thông qua các thuật toán thống kê, mạng thần kinh và học sâu. Bạn cũng sẽ xem xét các chi tiết bên trong và bên ngoài của xếp hạng cũng như cách kiểm tra các ứng dụng dựa trên các thuật toán thông minh.

### Có gì bên trong

Học máy cho người mới      Các  
thuật toán cho IoT, sức khỏe cộng đồng và các lĩnh vực  
khác      Giới thiệu về học sâu và mạng lưới thần kinh      Làm rõ  
cách các công cụ đề xuất thực sự hoạt động

## Khai thác văn bản và phân tích văn bản

Chữ khai thác là khoa học trích xuất thông tin có thể sử dụng máy từ chữ. Học máy cổ điển tập trung vào phân tích dữ liệu có cấu trúc hoặc có trật tự, nhưng dữ liệu đó chỉ đại diện cho một tỷ lệ nhỏ thông tin xung quanh chúng ta. Chương sau sử dụng Python và Natural Language Toolkit (NLTK) để minh họa các phép biến đổi cổ điển được sử dụng để chuyển đổi văn bản tự do thành dữ liệu có cấu trúc. Dự án ví dụ xây dựng một hệ thống để phân loại các bài đăng Reddit thành danh mục khác nhau. Sự phân loại này sau đó có thể được sử dụng làm thông tin có cấu trúc trong các dự án bổ sung.

Chương 8 từ Giới thiệu Khoa học Dữ liệu  
của Davy Cielen, Arno DB Meysman, và  
Mohamed Ali

## Khai thác văn bản và phân tích văn bản

### Chương này bao gồm

- Hiểu tầm quan trọng của khai thác văn bản ■ Giới thiệu các khái niệm quan trọng nhất trong khai thác văn bản
- Làm việc thông qua một dự án khai thác văn bản

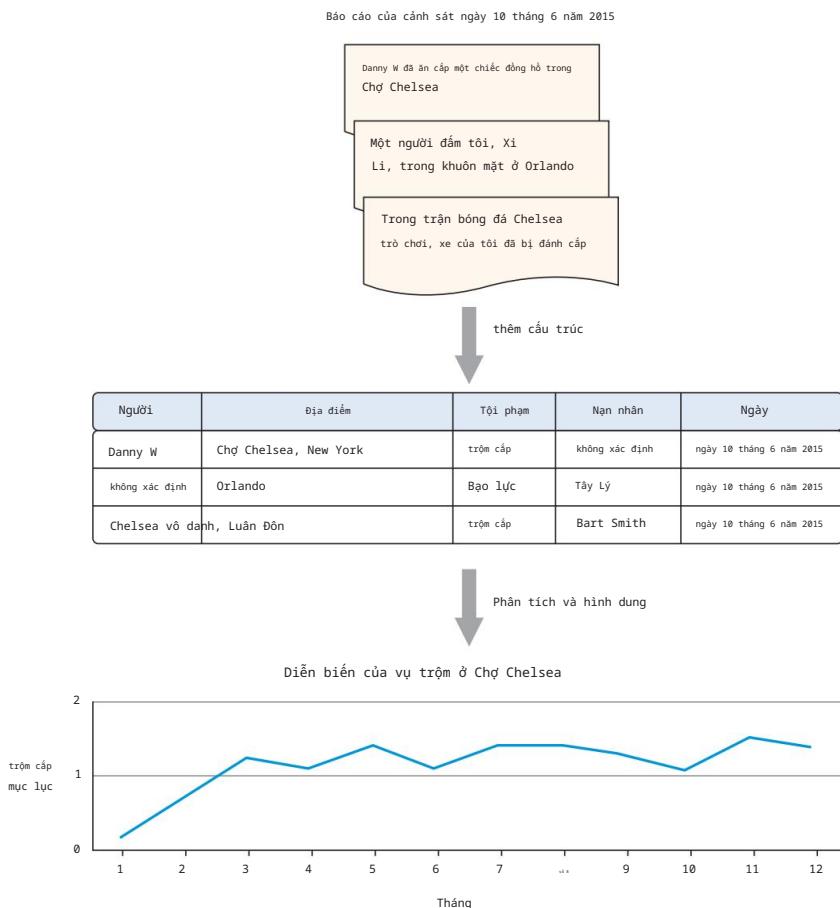
Hầu hết các thông tin được con người ghi lại trên thế giới đều ở dạng văn bản.

Tất cả chúng ta đều học đọc và viết từ khi còn nhỏ để chúng ta có thể thể hiện bản thân thông qua việc viết và tìm hiểu những gì người khác biết, suy nghĩ và cảm nhận. Chúng tôi sử dụng kỹ năng này mọi lúc khi đọc hoặc viết email, blog, tin nhắn văn bản hoặc cuốn sách này, vì vậy không có gì ngạc nhiên khi ngôn ngữ viết đến với hầu hết chúng ta một cách tự nhiên. Các doanh nghiệp tin rằng nhiều giá trị có thể được tìm thấy trong các văn bản mà mọi người tạo ra, và đúng như vậy bởi vì chúng chứa thông tin về những gì mọi người thích, không thích, những gì họ biết hoặc muốn biết, khao khát và mong muốn, sức khỏe hoặc tâm trạng hiện tại của họ, và nhiều hơn nữa. Nhiều thứ trong số này có thể phù hợp với các công ty hoặc nhà nghiên cứu, nhưng không một người nào có thể tự mình đọc và giải thích cơn sóng thần tài liệu bằng văn bản này. Một lần nữa, chúng ta cần chuyển sang máy tính để thực hiện công việc cho chúng ta.

Tuy nhiên, đáng buồn là ngôn ngữ tự nhiên không “tự nhiên” đối với máy tính như đối với con người. Rút ra ý nghĩa và lọc ra những thứ không quan trọng

quan trọng vẫn là thứ con người giỏi hơn máy móc. May mắn thay, dữ liệu các nhà khoa học có thể áp dụng các kỹ thuật phân tích văn bản và khai thác văn bản cụ thể để tìm thông tin liên quan trong hàng đồng văn bản mà nếu không họ sẽ mất hàng thế kỷ để đọc chúng tôi.

Khai thác văn bản hoặc phân tích văn bản là một môn học kết hợp khoa học ngôn ngữ và khoa học máy tính với các kỹ thuật thống kê và học máy. Khai thác văn bản được sử dụng để phân tích văn bản và biến chúng thành một dạng có cấu trúc hơn. Sau đó, nó mất cái này dạng có cấu trúc và cố gắng rút ra những hiểu biết sâu sắc từ nó. Khi phân tích tội phạm từ cảnh sát báo cáo, ví dụ: khai thác văn bản giúp bạn nhận ra người, địa điểm và loại tội phạm từ các báo cáo. Sau đó, cấu trúc mới này được sử dụng để hiểu rõ hơn về sự phát triển của tội phạm. Xem hình 8.1.



Hình 8.1 Trong phân tích văn bản, (thường) thách thức đầu tiên là cấu trúc văn bản đầu vào; sau đó nó có thể được phân tích kỹ lưỡng

Mặc dù ngôn ngữ không giới hạn ở ngôn ngữ tự nhiên, nhưng trọng tâm của chương này sẽ là Xử lý ngôn ngữ tự nhiên (NLP). Ví dụ về các ngôn ngữ phi tự nhiên sẽ là nhật ký máy, toán học và mã Morse. Về mặt kỹ thuật, ngay cả Esperanto, Klingon và Dragon language cũng không thuộc lĩnh vực ngôn ngữ tự nhiên vì chúng được phát minh một cách có chủ ý thay vì phát triển theo thời gian; chúng không đến với chúng tôi một cách "tự nhiên".

Tuy nhiên, những ngôn ngữ cuối cùng này phù hợp với giao tiếp tự nhiên (nói, viết); chúng có ngữ pháp và từ vựng giống như tất cả các ngôn ngữ tự nhiên, và các kỹ thuật khai thác văn bản tương tự có thể áp dụng cho chúng.

### 8.1 Khai thác văn bản trong thế giới thực

Trong cuộc sống hàng ngày, bạn đã bắt gặp các ứng dụng khai thác văn bản và ngôn ngữ tự nhiên. Tự động điền và sửa lỗi chính tả liên tục phân tích văn bản bạn nhập trước khi gửi email hoặc tin nhắn văn bản. Khi Facebook tự động hoàn thành trạng thái của bạn với tên của một người bạn, nó thực hiện điều này với sự trợ giúp của một kỹ thuật gọi là nhận dạng thực thể có tên, mặc dù đây chỉ là một thành phần trong toàn bộ hoạt động của họ. Mục tiêu không chỉ để phát hiện bạn đang gõ một danh từ mà còn để đoán bạn đang đề cập đến một người và nhận ra đó có thể là ai. Một ví dụ khác về nhận dạng thực thể được đặt tên được hiển thị trong hình 8.2. Google biết Chelsea là một câu lạc bộ bóng đá nhưng phản hồi khác khi được hỏi về một người.

Google sử dụng nhiều loại khai thác văn bản khi trình bày cho bạn kết quả của một truy vấn. Điều gì hiện lên trong đầu bạn khi ai đó nói "Chelsea"? Chelsea có thể là nhiều thứ: một con người; một câu lạc bộ bóng đá; một khu phố ở Manhattan, New York hoặc London; chợ thực phẩm; một buổi trình diễn hoa; và như thế. Google biết điều này và trả về các câu trả lời khác nhau cho câu hỏi "Chelsea là ai?" so với "Chelsea là gì?"

Để cung cấp câu trả lời phù hợp nhất, Google phải thực hiện (trong số những việc khác) tất cả những điều sau:

- Xử lý trước tất cả các tài liệu mà nó thu thập cho các thực thể được đặt tên ■ Thực hiện nhận dạng ngôn ngữ ■ Phát hiện loại thực thể bạn đang đề cập đến ■ Khớp truy vấn với kết quả ■ Phát hiện loại nội dung sẽ trả về (PDF, nhạy cảm với người lớn)

Ví dụ này cho thấy rằng khai thác văn bản không chỉ về ý nghĩa trực tiếp của văn bản mà còn liên quan đến các thuộc tính meta như ngôn ngữ và loại tài liệu.

Google sử dụng khai thác văn bản cho nhiều mục đích hơn là trả lời các truy vấn. Bên cạnh việc bảo vệ người dùng Gmail khỏi thư rác, nó cũng chia email thành các danh mục khác nhau như xã hội, cập nhật và diễn đàn, như thể hiện trong hình 8.3.

Có thể tiến xa hơn nhiều so với việc trả lời các câu hỏi đơn giản khi bạn kết hợp văn bản với logic và toán học khác.

what is chelsea

Web Images Maps News Videos More Search tools

About 202.000.000 results (0,48 seconds)

**Chelsea F.C. - Wikipedia, the free encyclopedia**  
[en.wikipedia.org/wiki/Chelsea\\_F.C.](https://en.wikipedia.org/wiki/Chelsea_F.C.) Chelsea Football Club / / are a professional football club based in Fulham, London, who play in the Premier League, the highest level of English football. Founded in 1905, the club have spent most of their history in the top tier of English football.  
 2014–15 Chelsea FC season - Roman Abramovich - Stamford Bridge - Eden Hazard

**Chelsea, London - Wikipedia, the free encyclopedia**  
[en.wikipedia.org/wiki/Chelsea,\\_London](https://en.wikipedia.org/wiki/Chelsea,_London) Chelsea is an affluent area in central London, bounded to the south by the River Thames. Its frontage runs from Chelsea Bridge along the Chelsea Embankment, Cheyne Walk, Lots Road and Chelsea Harbour.  
 History - The borough of artists - Swinging Chelsea and today - Sports

**Urban Dictionary: Chelsea**  
[www.urbandictionary.com/define.php?term=Chelsea](https://www.urbandictionary.com/define.php?term=Chelsea) Chelsea is a beautiful creature of a peculiar nature. She is often starving or not hungry in the least, but she is dangerous in her hungry state. Possibly the sexiest ...

## Chelsea F.C.

Football club



Chelsea Football Club are a professional football club based in Fulham, London, who play in the Premier League, the highest level of English football. Founded in 1905, the club have spent most of their history in the top tier of English football. [Wikipedia](#)

**Manager:** José Mourinho

**League:** Premier League

**Arena/Stadium:** Stamford Bridge

**Training ground:** Cobham Training Centre

**Founded:** March 10, 1905

**Founders:** Gus Mears, Joseph Mears

who is chelsea

Web Images Videos News Maps More Search tools

About 198.000.000 results (0,37 seconds)

**Chelsea Handler - Wikipedia, the free encyclopedia**

[en.wikipedia.org/wiki/Chelsea\\_Handler](https://en.wikipedia.org/wiki/Chelsea_Handler) Chelsea Joy Handler (born February 25, 1975) is an American comedian, actress, author, television host, producer, and activist for gay rights. She hosted a late-night talk show called *Chelsea Lately* on the E! network from 2007 to 2014, and is currently preparing to host a show on Netflix in 2016.

Ted Harbert - *Chelsea Lately* - Uganda Be Kidding Me: Live - Ford Pinto

**See results about**

Chelsea F.C. (Football club)

Manager: José Mourinho

League: Premier League



Feedback

Hình 8.2 Các câu trả lời khác nhau cho câu hỏi "Chelsea là ai?" và "Chelsea là gì?" ngũ ý rằng Google sử dụng các kỹ thuật khai thác văn bản để trả lời các truy vấn này.

Primary	Social	Promotions
<input type="checkbox"/> <input type="checkbox"/> Stack Exchange	2 new items in your Stack Exchange inbox - The follow	
<input type="checkbox"/> <input type="checkbox"/> Stack Exchange	1 new item in your Stack Exchange inbox - The follow	

Hình 8.3 Email có thể được tự động chia theo danh mục dựa trên nội dung và nguồn gốc.

Điều này cho phép tạo ra các công cụ suy luận tự động được điều khiển bởi các truy vấn ngôn ngữ tự nhiên. Hình 8.4 cho thấy cách “Wolfram Alpha”, một công cụ kiến thức tính toán, sử dụng khai thác văn bản và suy luận tự động để trả lời câu hỏi “Dân số Hoa Kỳ có lớn hơn Trung Quốc không?”

The screenshot shows the WolframAlpha interface. At the top, the logo and tagline "computational knowledge engine" are visible. Below the logo, the search query "Is the USA population bigger than China" is entered. Underneath the query, there are two explanatory notes: "Assuming 'China' is a country | Use as an administrative division instead" and "Assuming 'bigger than' is referring to math | Use 'bigger' as referring to an adjective instead". The main result section is titled "Input interpretation:" and shows the query "is United States population > China population" with the "United States" and "population" terms highlighted. The "Results:" section displays the comparison: "is United States population > 1.36 billion people (2014 estimate)" and "China population 1.36 billion people (2014 estimate)". Below the results, there are links for "Sources" and "Download page", and a note "POWERED BY THE WOLFRAM LANGUAGE".

Hình 8.4 Công cụ Wolfram Alpha sử dụng khai thác văn bản và lập luận logic để trả lời một câu hỏi.

Nếu điều này vẫn chưa đủ ấn tượng, thì IBM Watson đã khiến nhiều người kinh ngạc vào năm 2011 khi cỗ máy này được thiết lập để chống lại hai người chơi là con người trong trò chơi Jeopardy. Jeopardy là một chương trình đố vui của Mỹ, trong đó mọi người nhận được câu trả lời cho một câu hỏi và điểm được tính khi đoán đúng câu hỏi cho câu trả lời đó. Xem hình 8.5.

Có thể nói vòng này dành cho trí tuệ nhân tạo. IBM Watson là một công cụ nhận thức có thể diễn giải ngôn ngữ tự nhiên và trả lời các câu hỏi dựa trên nền tảng kiến thức sâu rộng.



Hình 8.5 IBM Watson thắng Jeopardy trước người chơi là con người.

Khai thác văn bản có nhiều ứng dụng, bao gồm, nhưng không giới hạn ở các ứng dụng sau:

■ Nhận dạng thực thể ■

Phát hiện đạo văn ■ Nhận  
dạng chủ đề ■ Phân cụm  
văn bản ■ Dịch thuật

■ Tự động tóm tắt văn bản

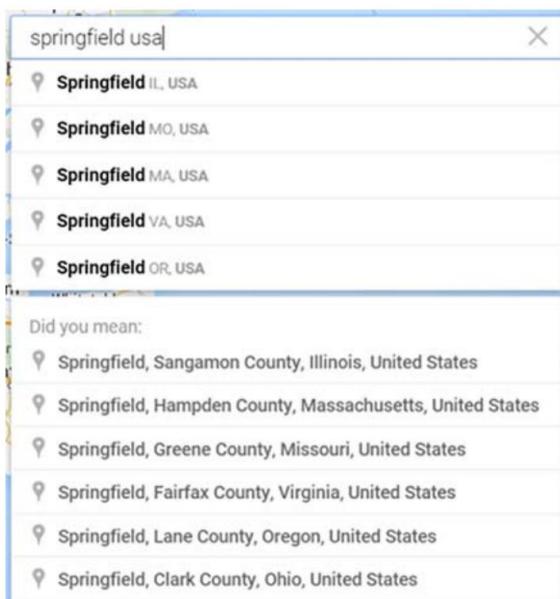
■ Phát hiện gian lận

■ Lọc thư rác ■

Phân tích tình cảm

Khai thác văn bản là hữu ích, nhưng nó có khó khăn? Xin lỗi để thất vọng: Vâng, đó là.

Khi xem các ví dụ về Wolfram Alpha và IBM Watson, bạn có thể có ấn tượng rằng việc khai thác văn bản rất dễ dàng. Thật đáng buồn không. Trong thực tế khai thác văn bản là một công việc phức tạp, thậm chí nhiều việc tưởng chừng như đơn giản cũng không thể thực hiện thỏa đáng. Chẳng hạn, nhận nhiệm vụ đoán đúng địa chỉ. Hình 8.6 cho thấy mức độ khó trả về kết quả chính xác với độ chắc chắn và cách Google Maps nhắc bạn cung cấp thêm thông tin khi tìm kiếm "Springfield". Trong trường hợp này, con người sẽ không làm tốt hơn nếu không có ngữ cảnh bổ sung, nhưng sự mơ hồ này là một trong nhiều vấn đề bạn gặp phải trong ứng dụng khai thác văn bản.



Hình 8.6 Google Maps yêu cầu bạn cung cấp thêm ngữ cảnh do truy vấn “Springfield” không rõ ràng.

Một vấn đề khác là lỗi chính tả và các dạng chính tả (đúng) khác nhau của một từ. Lấy ba tham chiếu sau đây đến New York: “NY,” “Neww York,” và “New York.” Cho một con người, thật dễ dàng để thấy tất cả họ đều đề cập đến thành phố New York. Bởi vì cách của chúng tôi bộ não diễn giải văn bản, hiểu văn bản có lỗi chính tả đến với chúng ta một cách tự nhiên; mọi người thậm chí có thể không nhận thấy chúng. Nhưng đối với máy tính, đây là những chuỗi không liên quan trừ khi chúng ta sử dụng các thuật toán để nói với nó rằng chúng đang đề cập đến cùng một thực thể. Có liên quan vấn đề là từ đồng nghĩa và việc sử dụng đại từ. Hãy thử chỉ định đúng người cho đại từ “cô ấy” trong các câu tiếp theo: “John tặng hoa cho bố mẹ Marleen khi anh ấy gặp bố mẹ cô ấy lần đầu tiên. Cô ấy rất hạnh phúc với cử chỉ này”. Vừa đủ dễ, Phải? Không phải cho một máy tính.

Chúng ta có thể giải nhiều bài toán tương tự một cách dễ dàng, nhưng chúng thường tỏ ra khó đối với một máy móc. Chúng ta có thể đào tạo các thuật toán hoạt động tốt trên một vấn đề cụ thể trong một phạm vi được xác định rõ ràng, nhưng các thuật toán tổng quát hơn hoạt động trong mọi trường hợp lại là một con thú khác. toàn bộ. Chẳng hạn, chúng ta có thể dạy máy tính nhận biết và truy xuất Hoa Kỳ số tài khoản từ văn bản, nhưng điều này không tổng quát tốt cho số tài khoản từ các nước khác.

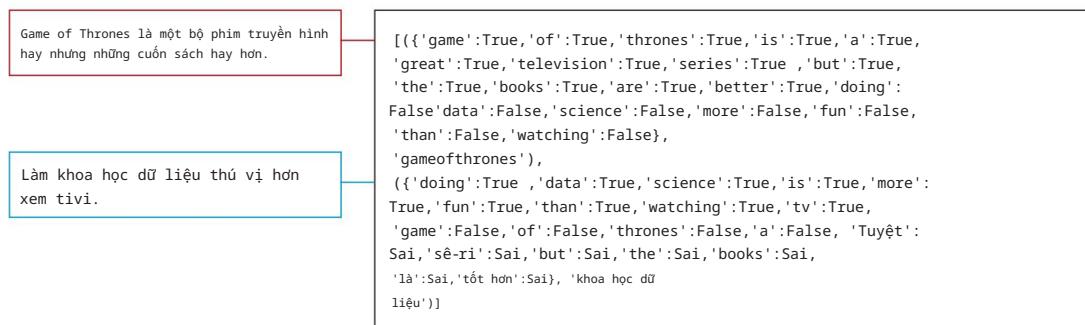
Các thuật toán ngôn ngữ cũng nhạy cảm với ngữ cảnh mà ngôn ngữ được sử dụng, thậm chí nếu bản thân ngôn ngữ vẫn giữ nguyên. Các mô hình tiếng Anh sẽ không hoạt động đối với tiếng Ả Rập và ngược lại ngược lại, nhưng ngay cả khi chúng tôi sử dụng tiếng Anh-một thuật toán được đào tạo cho dữ liệu Twitter cũng không có khả năng để thực hiện tốt các văn bản quy phạm pháp luật. Hãy ghi nhớ điều này khi chúng ta chuyển sang nghiên cứu điển hình trong chương này: không có giải pháp hoàn hảo, một kích cỡ phù hợp cho tất cả trong khai thác văn bản.

## 8.2 Kỹ thuật khai thác văn bản

Trong nghiên cứu điển hình sắp tới, chúng ta sẽ giải quyết vấn đề phân loại văn bản: tự động phân loại các văn bản chưa được phân loại thành các danh mục cụ thể. Để chuyển từ dữ liệu văn bản thô đến đích cuối cùng, chúng tôi sẽ cần một vài kỹ thuật khai thác dữ liệu yêu cầu thông tin cơ bản để chúng tôi sử dụng chúng một cách hiệu quả. Khái niệm quan trọng đầu tiên trong khai thác văn bản là “túi từ”.

### 8.2.1 Túi từ

Để xây dựng mô hình phân loại của chúng tôi, chúng tôi sẽ sử dụng cách tiếp cận túi từ. Túi từ là cách đơn giản nhất để cấu trúc dữ liệu văn bản: mọi tài liệu được biến thành một vectơ từ. Nếu một từ nào đó xuất hiện trong vec-tơ thì nó được gắn nhãn là “True”; những cái khác được dán nhãn “Sai”. Hình 8.7 cho thấy một ví dụ đơn giản về điều này, trong trường hợp chỉ có hai tài liệu: một về chương trình truyền hình Game of Thrones và một về khoa học dữ liệu. Hai vectơ từ cùng nhau tạo thành ma trận thuật ngữ tài liệu. Ma trận thuật ngữ tài liệu chứa một cột cho mọi thuật ngữ và một hàng cho mọi tài liệu. Các giá trị là của bạn để quyết định. Trong chương này, chúng ta sẽ sử dụng hệ nhị phân: thuật ngữ có mặt không? Đúng hay sai.



Hình 8.7 Một văn bản được chuyển đổi thành một túi từ bằng cách gắn nhãn cho mỗi từ (thuật ngữ) là “True” nếu nó có trong tài liệu và “False” nếu không.

Ví dụ từ hình 8.7 cung cấp cho bạn ý tưởng về dữ liệu có cấu trúc mà chúng ta sẽ cần để bắt đầu phân tích văn bản, nhưng nó rất đơn giản: không một từ nào được lọc ra và không áp dụng từ gốc (chúng ta sẽ đi sâu vào vấn đề này sau). Một kho ngữ liệu lớn có thể có hàng ngàn từ độc đáo. Nếu tất cả phải được gắn nhãn như thế này mà không có bất kỳ bộ lọc nào, thì dễ dàng nhận thấy rằng chúng tôi có thể nhận được một khối lượng lớn dữ liệu. Túi từ được mã hóa nhị phân như trong hình 8.7 chỉ là một cách để cấu trúc dữ liệu; các kỹ thuật khác tồn tại.

### Tần suất thuật ngữ–Tần suất tài liệu nghịch đảo (TF-IDF)

Một công thức nổi tiếng để điền vào ma trận thuật ngữ tài liệu là TF-IDF hoặc Tần suất thuật ngữ nhân với Tần số Tài liệu Nghịch đảo. Túi nhị phân của các từ chỉ định Đúng hoặc Sai (thuật ngữ có hay không), trong khi tần số đơn giản đếm số lần thuật ngữ xảy ra. TF-IDF phức tạp hơn một chút và tính đến số lần một thuật ngữ xảy ra trong tài liệu (TF). TF có thể là một số hạng đơn giản, một số nhị phân (Đúng hoặc Sai) hoặc số thuật ngữ được chia tỷ lệ logarit. Nó phụ thuộc vào những gì hoạt động tốt nhất cho bạn. Trong trường hợp TF là tần số của thuật ngữ, công thức của TF như sau:

$$TF = ft, d$$

TF là tần suất ( $f$ ) của thuật ngữ ( $t$ ) trong tài liệu ( $d$ ).

Nhưng TF-IDF cũng tính đến tất cả các tài liệu khác vì Nghịch đảo Tần suất tài liệu. IDF đưa ra ý tưởng về mức độ phổ biến của từ này trong toàn bộ văn bản: tần suất tài liệu càng cao thì càng có nhiều từ phổ biến và phổ biến hơn ít thông tin hơn. Ví dụ: từ "a" hoặc "the" không có khả năng cung cấp thông tin cụ thể về văn bản. Công thức của IDF với tỷ lệ logarit là nhất hình thức IDF thường được sử dụng:

$$IDF = \log(N / |d : t : d|)$$

với  $N$  là tổng số tài liệu trong kho văn bản và  $|d : t : d|$  là số tài liệu ( $d$ ) trong đó thuật ngữ ( $t$ ) xuất hiện.

Điểm số TF-IDF nói lên điều này về một thuật ngữ: từ này quan trọng như thế nào để phân biệt điều này tài liệu từ những người khác trong kho văn bản? Do đó, công thức của TF-IDF là

$$\frac{TF}{IDF} = \frac{f}{N} \log(\frac{N}{|d : t : d|})$$

Chúng tôi sẽ không sử dụng TF-IDF, nhưng khi thiết lập các bước tiếp theo của bạn trong khai thác văn bản, đây sẽ là một trong những điều đầu tiên bạn sẽ gặp phải. TF-IDF cũng là thứ đã được Elasticsearch sử dụng hậu trường trong chương 6. Đó là một cách hay nếu bạn muốn sử dụng TF-IDF cho phân tích văn bản; để việc khai thác văn bản cho phần mềm chuyên dụng như SOLR hoặc Tìm kiếm đàn hồi và lấy ma trận tài liệu/thuật ngữ để phân tích văn bản từ đó.

Trước khi đi đến túi từ thực tế, nhiều bước thao tác dữ liệu khác cần thực hiện địa điểm:

- Mã thông báo– Văn bản được cắt thành nhiều phần gọi là “mã thông báo” hoặc “điều khoản”. Những cái này mã thông báo là đơn vị thông tin cơ bản nhất mà bạn sẽ sử dụng cho mô hình của mình. Các thuật ngữ thường là từ nhưng đây không phải là điều cần thiết. Toàn bộ câu có thể được sử dụng cho Phân tích. Chúng tôi sẽ sử dụng unigram: thuật ngữ bao gồm một từ. Tuy nhiên, thường thì nó hữu ích để bao gồm bigram (hai từ cho mỗi mã thông báo) hoặc trigram (ba từ cho mỗi mã thông báo) để nắm bắt thêm ý nghĩa và tăng hiệu suất cho các mô hình của bạn.

Tuy nhiên, điều này phải trả giá vì bạn đang xây dựng các vector thuật ngữ lớn hơn bằng cách bao gồm bigram và/hoặc bát quái trong phương trình.

- **Ngừng lọc từ**-Mỗi ngôn ngữ đều có những từ ít có giá trị trong văn bản phân tích vì chúng được sử dụng quá thường xuyên. NLTK đi kèm với một danh sách tiếng Anh ngắn dừng từ chúng ta có thể lọc. Nếu văn bản được mã hóa thành từ, nó thường có ý nghĩa để loại bỏ vector từ của những từ dừng ít thông tin này.
- **Chữ thường**-Các từ có chữ in hoa xuất hiện ở đầu câu, những người khác vì chúng là danh từ hoặc tính từ riêng. Chúng tôi không nhận được giá trị gia tăng tạo sự khác biệt đó trong ma trận thuật ngữ của chúng tôi, vì vậy tất cả các thuật ngữ sẽ được đặt thành chữ thường.

Một kỹ thuật chuẩn bị dữ liệu khác là bắt nguồn. Điều này đòi hỏi nhiều công phu hơn.

#### 8.2.2 Từ gốc và từ vựng

**Bắt từ** là quá trình đưa các từ trở lại dạng gốc của chúng; theo cách này bạn kết thúc với ít phương sai hơn trong dữ liệu. Điều này có ý nghĩa nếu các từ có ý nghĩa tương tự nhau nhưng được viết khác đi bởi vì, ví dụ, one ở dạng số nhiều. nỗ lực xuất phát để thống nhất bằng cách cắt bỏ các phần của từ. Ví dụ: "máy bay" và "máy bay" đều trở thành "máy bay".

Một kỹ thuật khác, được gọi là **từ vựng hóa**, có cùng mục tiêu này nhưng thực hiện theo cách khác. cách nhạy cảm về mặt ngữ pháp. Ví dụ, trong khi cả gốc và từ vựng sẽ rút gọn "ô tô" thành "ô tô", việc bổ sung từ vựng cũng có thể đưa các động từ liên hợp trở lại thành dạng không liên hợp của chúng chẳng hạn như "are" thành "be". Cái nào bạn sử dụng phụ thuộc vào bạn trường hợp và từ vựng thu được nhiều lợi nhuận từ **Gắn thẻ POS** (Một phần của Gắn thẻ bài phát biểu). Gắn thẻ POS là quá trình gán nhãn ngữ pháp cho mọi phần của câu. Bạn có thể đã làm điều này một cách thủ công ở trường như một bài tập ngôn ngữ. Lấy ví dụ "Trò chơi vương quyền là một bộ phim truyền hình dài tập." Nếu chúng tôi áp dụng **Gắn thẻ POS** trên đó, chúng tôi sẽ nhận được

```
{"{"trò chơi":"NN"}, {"của":"IN"}, {"ngai vàng":"NNS"}, {"là":"VBZ"}, {"a":"DT"}, {"tivi ":"NN"}, {"sê-ri":"NN"}
```

NN là danh từ, IN là giới từ, NNS là danh từ ở dạng số nhiều, VBZ là ngôi thứ ba động từ số ít, và DT là một từ hạn định. Bảng 8.1 có danh sách đầy đủ.

Bảng 8.1 Danh sách tất cả các thẻ POS

Nhãn	Nghĩa	Nhãn	Nghĩa
CC	Phối hợp cùng	điều	bảng số
DT	định thức	SẢN TẠO	hiện sinh
FW	từ nước ngoài	TRONG	Giới từ hoặc liên từ phụ thuộc
JJ	Tính từ	JJR	Tính từ, so sánh
JJS	Tính từ, so sánh nhất	Ls	Đánh dấu mục danh sách
MD	phương thức	NN	Danh từ, số ít hoặc đại chúng

Bảng 8.1 Danh sách tất cả các thẻ POS (tiếp theo)

Nhãn	Nghĩa	Nhãn	Nghĩa
NNP	Danh từ, số nhiều	NNP	Danh từ riêng, số ít
NNPS	Danh từ riêng, số nhiều	PDT	định trước
này bắn hàng	kết thúc sở hữu	PRP	Đại từ nhân xưng
PRP\$	Đại từ sở hữu	RB	trạng từ
RBR	Trạng ngữ, so sánh	Trạng từ RBS, so sánh nhất	
RP	hạt	Biểu tượng SYM	
UH	thán từ	VB	Động từ, hình thức cơ sở
VBD	Động từ, thì quá khứ	VBG	Động từ, danh động từ hoặc hiện tại phân tử
VBN	Động từ, quá khứ phân tử	VBP	Động từ, hiện tại số ít không phải ngôi thứ 3
VBZ	Động từ, hiện tại ngôi thứ 3 số ít	WDT	Wh-xác định
WP	Wh-đại từ	WP\$	Sở hữu wh-đại từ
WRB	Wh-trạng từ		

Gắn thẻ POS là một trường hợp sử dụng mã hóa câu thay vì mã hóa từ.

Sau khi Gắn thẻ POS hoàn tất, bạn vẫn có thẻ tiếp tục mã hóa từ, nhưng

Trình gắn thẻ POS yêu cầu cả câu. Kết hợp Gắn thẻ POS và từ vựng là  
có khả năng cung cấp dữ liệu sạch hơn so với chỉ sử dụng một trình tạo gốc. Để đơn giản chúng ta sẽ  
gắn bó với việc bắt nguồn từ nghiên cứu điển hình, nhưng hãy coi đây là cơ hội để giải thích chi tiết hơn  
tập thẻ dực.

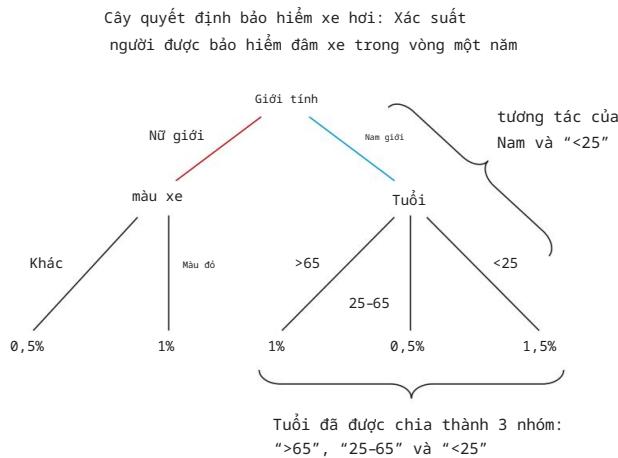
Bây giờ chúng tôi biết những điều quan trọng nhất mà chúng tôi sẽ sử dụng để làm sạch dữ liệu và  
thao tác (khai thác văn bản). Đối với phân tích văn bản của chúng tôi, hãy thêm trình phân loại cây quyết định  
đến tiêt mục của chúng tôi.

### 8.2.3 Trình phân loại cây quyết định

Phần phân tích dữ liệu trong nghiên cứu điển hình của chúng tôi cũng sẽ được giữ đơn giản. Chúng tôi sẽ kiểm tra một Naïve  
Bộ phân loại Bayes và bộ phân loại cây quyết định. Như đã thấy trong chương 3 Naïve Bayes  
bộ phân loại được gọi như vậy bởi vì nó coi mỗi biến đầu vào là độc lập với  
tất cả những thứ khác, điều này thật ngây thơ, đặc biệt là trong khai thác văn bản. Lấy ví dụ đơn giản về  
“khoa học dữ liệu”, “phân tích dữ liệu” hoặc “trò chơi vương quyền”. Nếu chúng ta cắt dữ liệu của mình bằng unigram  
chúng tôi nhận được các biến riêng biệt sau (nếu chúng tôi bỏ qua xuất phát và như vậy): “dữ liệu”, “khoa học”,  
“phân tích”, “trò chơi”, “của” và “ngai vàng”. Rõ ràng là các liên kết sẽ bị mất. Điều này có thể, trong  
biến, được khắc phục bằng cách tạo bigrams (khoa học dữ liệu, phân tích dữ liệu) và bất quái  
(trò chơi vương quyền).

Tuy nhiên, trình phân loại cây quyết định không coi các biến là độc lập với nhau và chủ động tạo các biến và  
nhóm tương tác . một sự tương tác

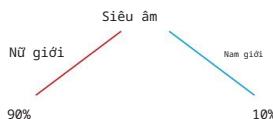
biến là một biến kết hợp các biến khác. Ví dụ “dữ liệu” và “khoa học” có thể là những yếu tố dự đoán tốt theo cách riêng của chúng nhưng có lẽ cả hai trong số chúng cùng xuất hiện trong cùng một văn bản có thể có giá trị riêng của nó. Một cái xô có phần ngược lại. Thay vì kết hợp hai biến, một biến được chia thành nhiều biến mới. Điều này có ý nghĩa đối với các biến số. Hình 8.8 cho thấy cây quyết định trông như thế nào và nơi bạn có thể tìm thấy sự tương tác và xô đẩy.



Hình 8.8 Mô hình cây quyết định hư cấu. Cây quyết định tự động tạo các nhóm và giả sử tương tác giữa các biến đầu vào.

Trong khi Naïve Bayes giả định tính độc lập của tất cả các biến đầu vào, cây quyết định được xây dựng dựa trên giả định về sự phụ thuộc lẫn nhau. Nhưng làm thế nào để nó xây dựng cấu trúc này? Cây quyết định có một số tiêu chí khả thi mà nó có thể sử dụng để chia thành các nhánh và quyết định xem biến nào quan trọng hơn (gần với gốc của cây) hơn các biến khác. Cái mà chúng ta sẽ sử dụng trong bộ phân loại cây quyết định NLTK là “thông tin nhận được.” Để hiểu mức tăng thông tin, trước tiên chúng ta cần xem xét entropy. Entropy là thước đo của sự không thể đoán trước hoặc sự hỗn loạn. Một ví dụ đơn giản là giới tính của em bé. Khi một người phụ nữ mang thai, giới tính của thai nhi có thể là nam hoặc nữ, nhưng chúng ta không biết đó là giới tính nào. Nếu bạn đoán, bạn có 50% cơ hội đoán đúng (cho hay nhận, vì phân bố giới tính không đồng đều 100%). Tuy nhiên, trong thời kỳ mang thai, bạn có cơ hội siêu âm để xác định giới tính của thai nhi. Siêu âm không bao giờ có thể kết luận chính xác 100%, nhưng càng xa trong quá trình phát triển của thai nhi, nó càng trở nên chính xác hơn. Mức tăng độ chính xác hoặc mức tăng thông tin này là do sự không chắc chắn hoặc entropy giảm xuống. Giả sử siêu âm khi thai 12 tuần có độ chính xác 90% trong việc xác định giới tính của em bé. Độ không chắc chắn 10% vẫn tồn tại, nhưng siêu âm đã làm giảm độ không chắc chắn

Xác suất thai nhi được xác định là  
nữ-siêu âm lúc 12 tuần



Hình 8.9 Cây quyết định với một biến: kết luận của bác sĩ khi xem siêu âm khi mang thai. Xác suất thai nhi là nữ là bao nhiêu?

từ 50% xuống còn 10%. Đó là một người phân biệt đối xử khá tốt. Một cây quyết định theo sau này cùng một nguyên tắc, như thể hiện trong hình 8.9.

Nếu một bài kiểm tra giới tính khác có sức mạnh dự đoán hơn, nó có thể trở thành gốc rễ của cây có kiểm tra siêu âm ở các nhánh và điều này có thể tiếp tục cho đến khi chúng tôi chạy ra khỏi các biến hoặc quan sát. Chúng ta có thể hết quan sát, bởi vì ở mọi chia nhánh, chúng tôi cũng chia dữ liệu đầu vào. Đây là một điểm yếu lớn của cây quyết định, bởi vì ở cấp độ lá của cây sẽ bị phá vỡ nếu có quá ít quan sát bên trái; cây quyết định bắt đầu overfit dữ liệu. Trang bị quá nhiều cho phép mô hình mắc lỗi tính ngẫu nhiên cho các tương quan thực. Để chống lại điều này, một cây quyết định được lược bỏ: các nhánh vô nghĩa bị loại khỏi mô hình cuối cùng.

Bây giờ chúng ta đã xem xét các kỹ thuật mới quan trọng nhất, hãy đi sâu vào nghiên cứu trường hợp.

### 8.3 Nghiên cứu điển hình: Phân loại bài đăng Reddit

Trong khai thác văn bản có nhiều ứng dụng, trong nghiên cứu tình huống của chương này, chúng tôi tập trung vào phân loại tài liệu. Như đã chỉ ra trước đó trong chương này, đây chính xác là những gì Google thực hiện khi nó sắp xếp email của bạn theo danh mục hoặc cố gắng phân biệt thư rác với email thông thường. Nó cũng được sử dụng rộng rãi bởi các trung tâm liên lạc xử lý các câu hỏi hoặc khiếu nại của khách hàng gửi đến: các khiếu nại bằng văn bản trước tiên sẽ đi qua bộ lọc phát hiện chủ đề để chúng có thể được chỉ định cho đúng người để xử lý. Tài liệu phân loại cũng là một trong những tính năng bắt buộc của hệ thống giám sát phương tiện truyền thông xã hội. Các tweet được theo dõi, các bài đăng trên diễn đàn hoặc Facebook, các bài báo, v.v.

các tài nguyên internet khác được gán nhãn chủ đề. Bằng cách này, chúng có thể được tái sử dụng trong báo cáo. Phân tích tinh cảm là một loại phân loại văn bản cụ thể: là tác giả của một bài đăng tiêu cực, tích cực, hoặc trung lập về một cái gì đó? "Cái gì đó" đó có thể được nhận ra với sự công nhận thực thể.

Trong nghiên cứu điển hình này, chúng tôi sẽ dựa trên các bài đăng từ Reddit, một trang web còn được gọi là "trang nhất của internet" tự xưng, và cố gắng đào tạo một mô hình có khả năng phân biệt liệu ai đó đang nói về "khoa học dữ liệu" hay "trò chơi vương quyền".

Kết quả cuối cùng có thể là một bản trình bày về mô hình của chúng tôi hoặc một ứng dụng tương tác toàn diện. Trong chương 9, chúng ta sẽ tập trung vào việc xây dựng ứng dụng cho người dùng cuối, vì vậy bây giờ chúng tôi sẽ tiếp tục trình bày mô hình phân loại của chúng tôi.

Để đạt được mục tiêu của mình, chúng tôi sẽ cần tất cả sự trợ giúp và công cụ mà chúng tôi có thể nhận được, và điều đó đã xảy ra Python một lần nữa sẵn sàng cung cấp chúng.

### 8.3.1 Làm quen với Bộ công cụ ngôn ngữ tự nhiên

Python có thể không phải là ngôn ngữ thực thi hiệu quả nhất trên trái đất, nhưng nó có một gói trưởng thành để khai thác văn bản và xử lý ngôn ngữ: Bộ công cụ ngôn ngữ tự nhiên (NLTK). NLTK là một tập hợp các thuật toán, chức năng và các tác phẩm được chú thích sẽ hướng dẫn bạn thực hiện những bước đầu tiên trong khai thác văn bản và xử lý ngôn ngữ tự nhiên. NLTK cũng được ghi lại một cách xuất sắc trên nltk.org. NLTK, tuy nhiên, không thường được sử dụng cho công việc cấp sản xuất, giống như các thư viện khác như scikit-learning.

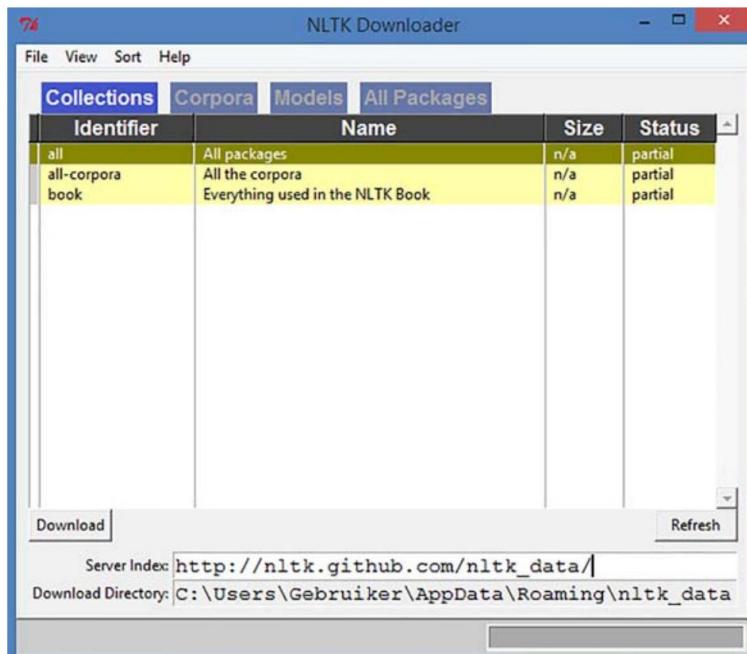
**Cài đặt NLTK và kho dữ liệu của nó**

Cài đặt NLTK bằng trình cài đặt gói yêu thích của bạn. Trong trường hợp bạn đang sử dụng Anaconda, nó được cài đặt với thiết lập Anaconda mặc định. Nếu không, bạn có thể sử dụng "pip" hoặc "Để cài đặt". Khi điều này được thực hiện, bạn vẫn cần cài đặt các mô hình và kho dữ liệu bao gồm để có nó được đầy đủ chức năng. Đối với điều này, hãy chạy mã Python sau:

- nhập nltk
- nltk.download()

Tùy thuộc vào cài đặt của bạn, điều này sẽ cung cấp cho bạn một cửa sổ bật lên hoặc nhiều tùy chọn dòng lệnh hơn. Hình 8.10 hiển thị hộp bật lên mà bạn nhận được khi thực hiện lệnh nltk.download().

Bạn có thể tải xuống tất cả kho ngôn ngữ nếu muốn, nhưng đối với chương này, chúng tôi sẽ chỉ sử dụng của "punkt" và "từ dừng". Tải xuống này sẽ được đề cập rõ ràng trong mã đi kèm với cuốn sách này.



Hình 8.10 Chọn Tất cả các Gói để hoàn tất cài đặt NLTK.

Hai tệp sổ tay IPython có sẵn cho chương này:

- Thu thập dữ liệu-Sẽ bao gồm phần thu thập dữ liệu trong nghiên cứu điển hình của chương này.
  - Chuẩn bị và phân tích dữ liệu- Dữ liệu được lưu trữ được đưa vào quá trình chuẩn bị dữ liệu và sau đó được đưa vào phân tích.

Tất cả mã trong nghiên cứu điển hình sắp tới có thể được tìm thấy trong hai tệp này theo cùng một trình tự và cũng có thể chạy như vậy. Ngoài ra, có sẵn hai biểu đồ tương tác để tải xuống:

- forceGraph.html—Đại diện cho 20 tính năng hàng đầu của mô hình Naïve Bayes của chúng tôi
  - Sunburst.html—Đại diện cho bốn nhánh hàng đầu của mô hình cây quyết định của chúng tôi

Để mở hai trang HTML này, cần có máy chủ HTTP mà bạn có thể sử dụng Python và một cửa sổ lệnh:

- Mở cửa sổ lệnh (Linux, Windows, bất cứ thứ gì bạn thích). ■ Di chuyển đến thư mục chứa các tệp HTML và tệp dữ liệu JSON của chúng : quyết định TreeData.json cho biểu đồ sunburst và NaiveBayesData.json cho biểu đồ lực. Điều quan trọng là các tệp HTML phải ở cùng vị trí với các tệp dữ liệu của chúng, nếu không bạn sẽ phải thay đổi JavaScript trong tệp HTML . ■ Tạo máy chủ HTTP Python bằng lệnh sau: python -m Simple

Máy chủ HTTP 8000

- Mở trình duyệt và truy cập localhost:8000; tại đây bạn có thể chọn các tệp HTML , như thể hiện trong hình 8.11.



## Directory listing for /

- [decisionTreeData.json](#)
  - [forceGraph.html](#)
  - [NaiveBayesData.json](#)
  - [sunburst.html](#)

Hình 8.11 Máy chủ HTTP Python phục vụ đầu ra của chương này

Các gói Python chúng ta sẽ sử dụng trong chương này:

- NLTK–Dành cho khai thác văn bản ■ PRAW–Cho phép tải xuống các bài đăng từ Reddit ■ SQLite3–Cho phép chúng tôi lưu trữ dữ liệu ở định dạng SQLite ■ Matplotlib–Thư viện đồ thị để trực quan hóa dữ liệu

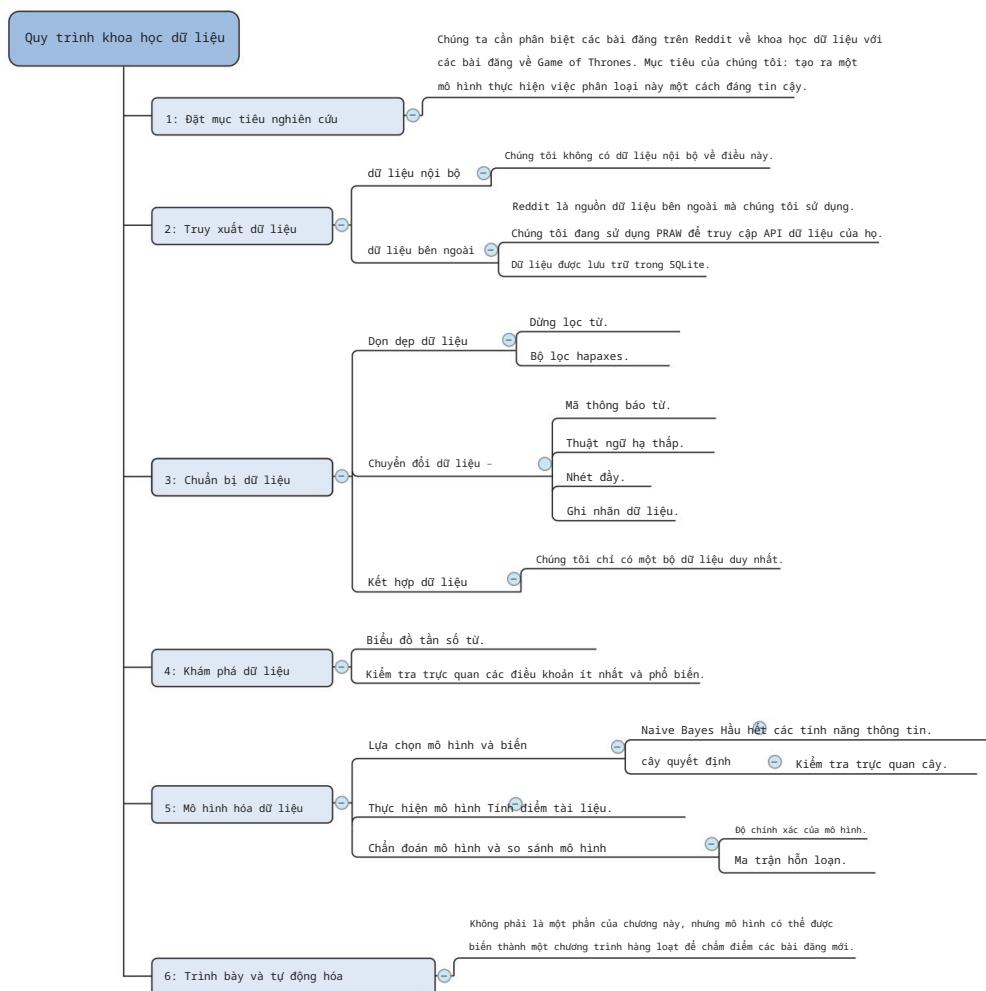
Đảm bảo cài đặt tất cả các thư viện và kho dữ liệu cần thiết trước khi tiếp tục. Tuy nhiên, trước khi đi sâu vào hành động, hãy xem xét các bước chúng ta sẽ thực hiện để đạt được mục tiêu tạo mô hình phân loại chủ đề.

### 8.3.2 Tổng quan về quy trình khoa học dữ liệu và bước 1: Mục tiêu nghiên cứu

Để giải bài tập khai thác văn bản này, một lần nữa chúng ta sẽ sử dụng quy trình khoa học dữ liệu.

Hình 8.12 cho thấy quy trình khoa học dữ liệu được áp dụng cho trường hợp phân loại Reddit của chúng tôi.

Không phải tất cả các yếu tố được mô tả trong hình 8.12 đều có ý nghĩa vào thời điểm này và phần còn lại của chương được dành để giải quyết vấn đề này trong thực tế khi chúng ta hướng tới mục tiêu nghiên cứu của mình: tạo ra một mô hình phân loại có khả năng phân biệt các bài viết về "khoa học dữ liệu" từ các bài đăng về "Trò chơi vương quyền". Không chần chừ gì nữa, hãy đi lấy dữ liệu của chúng ta.



Hình 8.12 Tổng quan về quy trình khoa học dữ liệu được áp dụng cho nghiên cứu tình huống phân loại chủ đề Reddit

### 8.3.3 Bước 2: Truy xuất dữ liệu

Chúng tôi sẽ sử dụng dữ liệu Reddit cho trường hợp này và đối với những người không quen thuộc với Reddit, hãy dành thời gian để tự làm quen với các khái niệm của nó tại [www.reddit.com](http://www.reddit.com).

Reddit tự gọi mình là “trang nhất của internet” vì người dùng có thể đăng mọi thứ họ thấy thú vị và/hoặc tìm thấy ở đâu đó trên internet và chỉ những thứ đó được nhiều người coi là thú vị được giới thiệu là “phổ biến” trên trang chủ của nó. Bạn có thể nói Reddit đưa ra cái nhìn tổng quan về những thứ thực hành trên internet. Bất kỳ người dùng nào có thể đăng trong một danh mục được xác định trước được gọi là “ subreddit”. Khi một bài viết được thực hiện, những người dùng khác có thể nhận xét về nội dung đó và có thể ủng hộ nội dung đó nếu họ thích nội dung đó hoặc không ủng hộ nội dung đó nếu họ không thích nội dung đó. Bởi vì một bài đăng luôn là một phần của subreddit, chúng tôi có dữ liệu meta này khi chúng tôi kết nối với API Reddit để lấy dữ liệu của mình. chúng tôi hiệu quả tìm nạp dữ liệu được gắn nhãn vì chúng tôi cho rằng một bài đăng trong “gameofthrones” của subreddit có liên quan gì đó đến “gameofthrones”.

Để truy cập dữ liệu của mình, chúng tôi sử dụng thư viện Reddit Python API chính thức có tên PRAW. Khi chúng tôi nhận được dữ liệu mình cần, chúng tôi sẽ lưu trữ dữ liệu đó trong một tệp giống như cơ sở dữ liệu nhẹ được gọi là SQLite. SQLite lý tưởng để lưu trữ một lượng nhỏ dữ liệu vì nó không yêu cầu bất kỳ thiết lập nào để sử dụng và sẽ trả lời các truy vấn SQL giống như bất kỳ quan hệ thông thường nào cơ sở dữ liệu nào. Bất kỳ phương tiện lưu trữ dữ liệu nào khác cũng được; nếu bạn thích cơ sở dữ liệu Oracle hoặc PostgreSQL, Python có một thư viện tuyệt vời để tương tác với chúng mà không cần để viết SQL. SQLAlchemy cũng sẽ hoạt động đối với các tệp SQLite. Hình 8.13 hiển thị dữ liệu bước truy xuất trong quy trình khoa học dữ liệu.



Hình 8.13 Bước truy xuất dữ liệu của quy trình khoa học dữ liệu cho trường hợp phân loại chủ đề Reddit

Mở trình thông dịch Python yêu thích của bạn; đã đến lúc hành động, như trong danh sách 8.1.

Trước tiên, chúng tôi cần thu thập dữ liệu của mình từ trang web Reddit. Nếu bạn chưa có, sử dụng pip install praw hoặc conda install praw (Anaconda) trước khi chạy tập lệnh hạ thấp theo dõi.

**LƯU Ý** Mã cho bước 2 cũng có thể được tìm thấy trong tệp IPython “Chương 8 thu thập dữ liệu.” Nó có sẵn trong phần tải xuống của cuốn sách này.

### Lịch kê 8.1 Thiết lập cơ sở dữ liệu SQLite và ứng dụng khách API Reddit

```

nhập praw nhập          | Nhập thư viện PRAW
sqlite3                  | và SQLite3.

conn = sqlite3.connect('reddit.db') c = conn.cursor()           | Thiết lập kết nối với
                                                               | cơ sở dữ liệu SQLite.

c.execute('''DROP TABLE IF EXISTS chủ đề''') c.execute('''DROP
TABLE IF EXISTS comments''') c.execute('''TAO TABLE chủ đề

(văn bản tiêu đề chủ đề, văn bản chủ đề, văn bản ID chủ đề,
topicCategory text)''')
c.execute('''CREATE TABLE comments

(văn bản commentText, văn bản commentID
topicVăn bản tiêu đề, topicText văn bản, topicID văn bản
chủ đềThể loại văn bản)'''')

user_agent = "Giới thiệu sách khoa học dữ liệu" r =
praw.Reddit(user_agent=user_agent)

subreddits = ['datascience', 'gameofthrones']

giới hạn = 1000

```

thực thi SQL  
tuyên bố để  
tạo chủ đề và bảng  
bình luận.

Tạo tác nhân người dùng PRAW để  
chúng tôi có thể sử dụng API Reddit.

Danh sách các subreddits của  
chúng tôi sẽ được đưa vào cơ  
sở dữ liệu SQLite của chúng tôi.

Trước tiên hãy nhập các thư viện cần thiết.

Bây giờ chúng ta có quyền truy cập vào các **khả năng** của SQLite3 và PRAW, chúng ta cần chuẩn bị cơ sở dữ liệu cục bộ nhỏ của chúng tôi cho dữ liệu sắp nhận được. Bằng cách xác định một kết nối đến một File SQLite chúng tôi tự động tạo nếu chưa có. Sau đó chúng tôi xác định một dữ liệu con trả về khả năng thực thi bất kỳ câu lệnh SQL nào, vì vậy chúng tôi sử dụng nó để xác định trước cấu trúc cơ sở dữ liệu của chúng tôi. Cơ sở dữ liệu sẽ chứa hai bảng: bảng chủ đề chứa các chủ đề Reddit, tương tự như ai đó bắt đầu một bài đăng mới trên diễn đàn và bảng thứ hai chứa các nhận xét và được liên kết với bảng chủ đề thông qua "topicID" cột. Hai bảng có mối quan hệ một (bảng chủ đề) với nhiều (bảng nhận xét). Đối với nghiên cứu điển hình, chúng tôi sẽ giới hạn bản thân trong việc sử dụng bảng chủ đề, nhưng phần thu thập dữ liệu sẽ kết hợp cả hai vì điều này cho phép bạn thử nghiệm thêm dữ liệu nếu bạn cảm thấy thích nó. Để trau dồi kỹ năng khai thác văn bản của bạn, bạn có thể thực hiện tinh cảm phân tích về các bình luận chủ đề và tìm ra chủ đề nào nhận được tiêu cực hoặc tích cực bình luận. Sau đó, bạn có thể tương quan điều này với các tính năng mô hình mà chúng tôi sẽ tạo ra bởi cuối chương này.

Chúng tôi cần tạo ứng dụng khách PRAW để có quyền truy cập vào dữ liệu. Mọi subreddit đều có thể được xác định bằng tên của nó và chúng tôi quan tâm đến "khoa học dữ liệu" và "trò chơi soán ngôi". Giới hạn thẻ hiện số lượng chủ đề tối đa (bài đăng, không phải bình luận) mà chúng tôi sẽ rút ra từ Reddit. Một nghìn cũng là số lượng tối đa mà API cho phép chúng tôi tìm nạp theo bất kỳ yêu cầu cụ thể nào, mặc dù sau này chúng tôi có thể yêu cầu nhiều hơn khi mọi người có

đăng những điều mới. Trên thực tế, chúng tôi có thể chạy yêu cầu API theo định kỳ và thu thập dữ liệu tăng ca. Mặc dù tại bất kỳ thời điểm nào, bạn bị giới hạn ở một nghìn bài đăng, nhưng không có gì dừng lại bạn khỏi việc phát triển cơ sở dữ liệu của riêng mình trong suốt nhiều tháng. Điều đáng chú ý là tập lệnh sau có thể mất khoảng một giờ để hoàn thành. Nếu bạn không muốn chờ đợi, vui lòng tiếp tục và sử dụng tệp SQLite có thể tải xuống. Ngoài ra, nếu bạn chạy nó bây giờ, bạn không có khả năng nhận được đầu ra chính xác như khi nó được chạy lần đầu tiên để tạo đầu ra thể hiện trong chương này.

Hãy xem chức năng truy xuất dữ liệu của chúng tôi, như được hiển thị trong danh sách sau.

#### Lịch kê 8.2 Truy xuất và lưu trữ dữ liệu Reddit trong SQLite

Các lĩnh vực cụ thể của chủ đề được thêm vào danh sách. Chúng tôi chỉ sử dụng tiêu đề và văn bản trong suốt bài tập nhưng ID chủ đề sẽ hữu ích cho việc xây dựng cơ sở dữ liệu chủ đề (lớn hơn) của riêng bạn.

```
def prawGetData(limit, subredditName):
    chủ đề = r.get_subreddit(subredditName).get_hot(limit=limit)
    commentInsert = []
```

```
    chủ đềChèn = []
    chủ đềNBR = 1
    cho chủ đề trong các chủ đề:
        nếu (float(topicNBR)/giới hạn)*100 trong xrange(1.100):
            in '***** CHỦ ĐỀ:' + str(topic.id)
+   ' *****HOÀN THÀNH: ' + str((float(topicNBR)/giới hạn)*100)
+   ' % *****'
        chủ đềNBR += 1
        thừ:
            topicInsert.append((topic.title, topic.selftext, topic.id,
                     subredditName))
    ngoại trừ:
        vượt qua
    thừ:
        để nhận xét trong chủ đề.comments:
            commentInsert.append((comment.body, comment.id,
                     topic.title, topic.selftext, topic.id, subredditName))
    ngoại trừ:
```

Từ  
subreddits, nhận 1.000  
chủ đề nóng nhất  
(trong trường hợp  
của chúng tôi).

Phản này là một bản in thông  
tin và không cần thiết để mã  
hoạt động. Nó chỉ thông báo  
cho bạn về tiến trình tải  
xuống.

Chèn tất cả  
các chủ đề vào  
SQLite  
cơ sở dữ liệu.

```
    vượt qua
    in '*****'
    in 'CHÈN DỮ LIỆU VÀO SQLITE'
    c.executemany('CHÈN VÀO chủ đề GIÁ TRỊ (?, ?, ?, ?, ?)', topicInsert)
    print 'CHÈN CHỦ ĐỀ'

    c.executemany('CHÈN VÀO nhận xét GIÁ TRỊ (?, ?, ?, ?, ?, ?)', commentInsert)
    print 'GIÁC NHẬN ĐƯỢC CHÈN'
```

Nối nhận xét vào một  
danh sách. Chúng không  
được sử dụng trong  
bài tập nhưng bây  
giờ bạn có chúng để  
thử nghiệm.

```
    conn.commit()

    cho chủ đề trong subreddits:
        prawGetData(limit=limit, subredditName=subject)
```

Cam kết thay đổi (chèn dữ liệu) vào cơ sở  
dữ liệu. Nếu không có cam kết, sẽ không  
có dữ liệu nào được chèn vào.

Chèn tất cả  
nhận xét vào  
cơ sở dữ liệu SQLite.

Hàm này được thực thi cho tất cả  
các subreddits mà chúng tôi  
đã chỉ định trước đó.

Hàm prawGetData () lấy các chủ đề “hot nhất” trong subreddit của nó, thêm vào này vào một mảng và sau đó nhận tất cả các nhận xét liên quan của nó. Điều này tiếp tục cho đến khi đạt đến một chủ đề hoặc không còn chủ đề nào tồn tại để tìm nạp và mọi thứ được lưu trữ trong cơ sở dữ liệu SQLite. Các báo cáo in ở đó để thông báo cho bạn về tiến trình của nó hướng tới việc thu thập hàng nghìn chủ đề. Tất cả những gì còn lại để chúng tôi làm là thực hiện chức năng cho mỗi subreddit.

Nếu bạn muốn phân tích này kết hợp nhiều hơn hai subreddits, đây là vấn đề về việc thêm một danh mục bổ sung vào mảng subreddits.

Với dữ liệu được thu thập, chúng tôi đã sẵn sàng chuyển sang chuẩn bị dữ liệu.

#### 8.3.4 Bước 3: Chuẩn bị dữ liệu

Như mọi khi, chuẩn bị dữ liệu là bước quan trọng nhất để có được kết quả chính xác. Để khai thác văn bản, điều này thậm chí còn đúng hơn vì chúng ta thậm chí không bắt đầu với dữ liệu có cấu trúc.

Mã sắp tới có sẵn trực tuyến dưới dạng tệp IPython “Chuẩn bị dữ liệu Chương 8 và phân tích.” Hãy bắt đầu bằng cách nhập các thư viện cần thiết và chuẩn bị SQLite cơ sở dữ liệu, như thể hiện trong danh sách sau đây.

**Liệt kê 8.3 Khai thác văn bản, thư viện, phụ thuộc kho văn bản và kết nối cơ sở dữ liệu SQLite**

```
nhập sqlite3 nhập
nltk nhập
matplotlib.pyplot dưới dạng plt từ bộ sưu tập
nhập OrderedDict nhập ngẫu nhiên
```

Nhập tất cả  
các thư  
viện cần thiết

```
nltk.download('punkt')
nltk.download('stopwords')
```

Tải xuống kho dữ liệu  
chúng tôi sử dụng

```
conn = sqlite3.connect('reddit.db') c = conn.cursor()
```

Tạo kết nối với cơ sở dữ liệu SQLite chứa dữ  
liệu Reddit của chúng tôi

Trong trường hợp bạn chưa tải xuống kho ngữ liệu NLTK đầy đủ, bây giờ chúng tôi sẽ tải xuống một phần của nó chúng tôi sẽ sử dụng. Để lo lắng nếu bạn đã tải xuống, tập lệnh sẽ phát hiện nếu kho dữ liệu của bạn được cập nhật.

Dữ liệu của chúng tôi vẫn được lưu trữ trong tệp Reddit SQLite, vì vậy hãy tạo kết nối với nó.

Ngay cả trước khi khám phá dữ liệu của mình, chúng tôi đã biết ít nhất hai điều chúng tôi phải làm để làm sạch dữ liệu: ngừng lọc từ và viết thường.

Một chức năng lọc từ tổng hợp sẽ giúp chúng ta lọc bỏ những phần chưa trong sạch. Hãy tạo ra một trong danh sách sau đây.

Liệt kê 8.4 Chức năng lọc từ và viết thường

```
def wordFilter(không bao gồm, wordrow):
    đã lọc = [từng từ trong hàng từ nếu từ không bị loại trừ]
    lọc trả lại

stopwords = nltk.corpus.stopwords.words('english') def LowerCaseArray(wordrow):
Lowercased = [word.lower() cho từ trong wordrow]

    trả lại chữ thường

    hàm LowerCaseArray()
    chuyển đổi bất kỳ số hạng nào thành số hạng của nó
    phiên bản chữ thường

    Biến từ dùng chứa
    các từ dùng tiếng Anh
    trên mỗi từ mặc
    định có trong NLTK
```

Các từ dùng tiếng Anh sẽ là từ đầu tiên rời khỏi dữ liệu của chúng tôi. Đoạn mã sau sẽ cung cấp cho chúng ta những từ dùng này:

```
từ khóa = nltk.corpus.stopwords.words('tiếng Anh')  
in từ dừng
```

Hình 8.14 hiển thị danh sách các từ dừng tiếng Anh trong NLTk.

```
stopwords = nltk.corpus.stopwords.words('english')
print stopwords
```

[u'i', u'me', u'my', u'myself', u'we', u'our', u'ours', u'ourselves', u'you', u'your', u'yours', u'yourself', u'yourselves', u'he', u'him', u'his', u'himsel f', u'she', u'her', u'hers', u'herself', u'it', u'its', u'itself', u'they', u'them', u'their', u'theirs', u'themselves', u'what', u'which', u'who', u'whom', u't his', u'that', u'these', u'those', u'am', u'is', u'are', u'was', u'were', u'be', u'been', u'being', u'have', u'has', u'had', u'having', u'do', u'does', u'did', u'doing', u'a', u'an', u'the', u'and', u'but', u'if', u'or', u'because', u'as', u'until', u'while', u'of', u'at', u'by', u'for', u'with', u'about', u'against', u'between', u'into', u'through', u'during', u'before', u'after', u'above', u'bel ow', u'to', u'from', u'up', u'down', u'in', u'out', u'on', u'off', u'over', u'un der', u'again', u'further', u'then', u'once', u'here', u'there', u'when', u'where', u'why', u'how', u'all', u'any', u'both', u'each', u'few', u'more', u'most', u'other', u'some', u'such', u'no', u'nor', u'not', u'only', u'own', u'same', u's o', u'than', u'too', u'very', u's', u't', u'can', u'will', u'just', u'don', u'should', u'now']

Hình 8.14 Danh sách từ dùng tiếng Anh trong NLTK

Với tất cả các thành phần cần thiết đã sẵn sàng, chúng ta hãy xem chức năng xử lý dữ liệu đầu tiên của chúng ta trong danh sách sau.



Hàm `data_processing()` của chúng tôi nhận một câu lệnh SQL và trả về ma trận thuật ngữ tài liệu. Nó thực hiện điều này bằng cách lặp qua một mục nhập dữ liệu (chủ đề Reddit) tại một thời gian và kết hợp tiêu đề chủ đề và nội dung chủ đề thành một vectơ từ duy nhất với việc sử dụng mã thông báo từ. Trình mã thông báo là tập lệnh xử lý văn bản cắt văn bản thành miếng. Bạn có nhiều cách khác nhau để mã hóa một văn bản: bạn có thể chia nó thành các nghĩa hoặc từ, bạn có thể chia nó theo dấu cách và dấu chấm câu, hoặc bạn có thể tính đến các ký tự khác, v.v. Ở đây, chúng tôi đã chọn mã thông báo từ NLTK tiêu chuẩn.

Từ mã thông báo này rất đơn giản; tất cả những gì nó làm là chia văn bản thành các thuật ngữ nếu có khoảng trắng giữa các từ. Sau đó, chúng tôi viết thường vectơ và lọc ra các từ dừng. Ghi chú thứ tự ở đây quan trọng như thế nào; một từ dừng ở đầu câu sẽ không được lọc nếu trước tiên chúng ta lọc các từ dừng trước khi viết thường. Ví dụ trong "Tôi thích Game of Thrones," chữ "I" sẽ không được viết thường và do đó sẽ không được lọc ngoài. Sau đó, chúng tôi tạo một ma trận từ (ma trận tài liệu thuật ngữ) và một danh sách chứa tất cả từ. Lưu ý cách chúng tôi mở rộng danh sách mà không cần lọc gấp đôi; theo cách này chúng ta có thể tạo biểu đồ về số lần xuất hiện từ trong quá trình khám phá dữ liệu. Hãy thực hiện chức năng cho hai loại chủ đề của chúng tôi.

Hình 8.15 cho thấy vectơ từ đầu tiên của danh mục "khoa học dữ liệu".

```
in dữ liệu['datascience']['wordMatrix'][0]
```

```
print data['datascience']['wordMatrix'][0]

[u'data', u'science', u'freelancing', u"'m", u'currently', u'master
s', u'program', u'studying', u'business', u'analytics', u"'m", u'try
ing', u'get', u'data', u'freelancing', u'.', u"'m", u'still', u'lear
ning', u'skill', u'set', u'typically', u'see', u'right', u"'m", u'fa
irly', u'proficient', u'sql', u'know', u'bit', u'r.', u'freelancer
s', u'find', u'jobs', u'?']
```

Hình 8.15 Vectơ từ đầu tiên của danh mục “khoa học dữ liệu” sau lần thử xử lý dữ liệu đầu tiên

Điều này chắc chắn có vẻ bị ô nhiễm: dấu câu được giữ dưới dạng các thuật ngữ riêng biệt và một số từ thậm chí còn chưa được chia. Khám phá dữ liệu sâu hơn sẽ làm rõ một số điều cho chúng tôi.

### 8.3.5 Bước 4: Khám phá dữ liệu

Bây giờ chúng tôi đã tách tất cả các điều khoản của mình, nhưng kích thước tuyệt đối của dữ liệu cần trở thành chúng tôi nắm rõ liệu nó có đủ sạch để sử dụng thực tế hay không. Bằng cách nhìn vào một vectơ, mặc dù vậy, chúng tôi đã phát hiện ra một số vấn đề: một số từ chưa được phân tách chính xác và vectơ chứa nhiều thuật ngữ có một ký tự. Thuật ngữ ký tự đơn có thể là yếu tố phân biệt chủ đề tốt trong một số trường hợp nhất định. Ví dụ, một văn bản kinh tế sẽ chứa nhiều ký hiệu \$, £ và © hơn văn bản y tế. Nhưng trong hầu hết các trường hợp, các thuật ngữ một ký tự này là vô ích. Đầu tiên, chúng ta hãy xem phân phối tần số của các điều khoản của chúng tôi.

```
wordfreqs_cat1 = nltk.FreqDist(data['datascience']['all_words'])
plt.hist(wordfreqs_cat1.values(), bins = range(10))
plt.show()
wordfreqs_cat2 = nltk.FreqDist(data['gameofthrones']['all_words'])
plt.hist(wordfreqs_cat2.values(), bins = range(20))
plt.show()
```

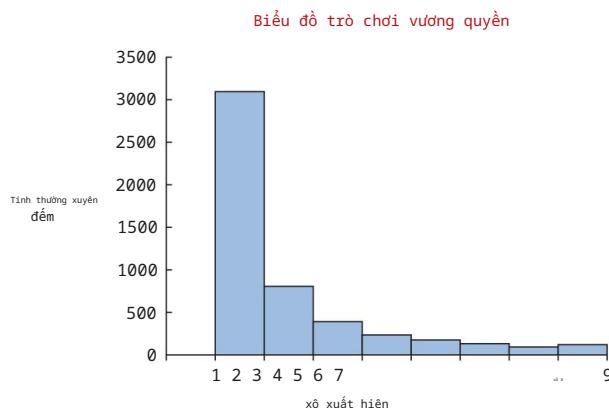
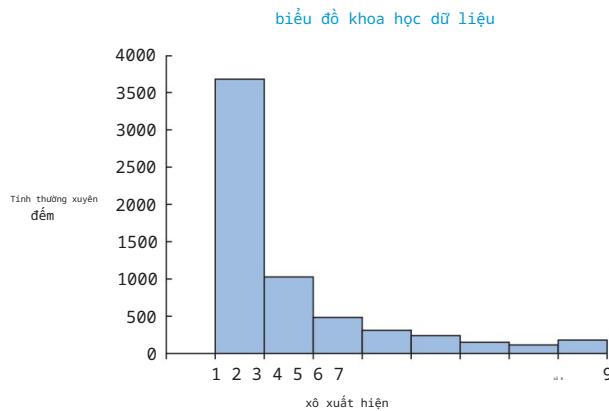
Bằng cách vẽ biểu đồ phân bố tần suất (hình 8.16), chúng ta nhanh chóng nhận thấy rằng phần lớn các thuật ngữ của chúng tôi chỉ xuất hiện trong một tài liệu.

Các thuật ngữ xuất hiện một lần như thế này được gọi là hapaxes và theo mô hình, chúng là vô dụng vì một lần xuất hiện của một tính năng không bao giờ đủ để xây dựng một người mẫu. Đây là tin tốt cho chúng tôi; tất những hapax này ra sẽ thu nhỏ đáng kể dữ liệu của chúng tôi mà không làm tổn hại đến mô hình cuối cùng của chúng tôi. Chúng ta hãy nhìn vào một vài trong số này thuật ngữ xảy ra.

```
in wordfreqs_cat1.hapaxes()
in wordfreqs_cat2.hapaxes()
```

Các thuật ngữ chúng tôi thấy trong hình 8.17 có ý nghĩa và nếu chúng tôi có nhiều dữ liệu hơn thì chúng có thể xảy ra thường xuyên hơn.

```
in wordfreqs_cat1.hapaxes()
in wordfreqs_cat2.hapaxes()
```



Hình 8.16 Biểu đồ tần suất thuật ngữ này cho thấy cả ma trận thuật ngữ "khoa học dữ liệu" và "trò chơi vương quyền" có hơn 3.000 thuật ngữ xảy ra một lần.

### Least frequent terms within data science posts

```
print wordfreqs_cat1.hapaxes()
[u'post-grad', u'marching', u'cytoscape', u'wizardry', u"'pure", u'i
m mature', u'socrata', u'filenotfoundexception', u'side-by-side', u'b
ringing', u'non-experienced', u'zestimate', u'formatting*', u'sustai
```

### Least frequent terms within Game of Thrones posts

```
print wordfreqs_cat2.hapaxes()
[u'hordes', u'woods', u'comically', u'pack', u'seventy-seven', u"'co
n text", u'shaving', u'kennels', u'differently', u'screaming', u'her-
', u'complainers', u'sailed', u'contributed', u'payoff', u'hallucina
```

Hình 8.17 Thuật ngữ xuất hiện đơn lẻ "Khoa học dữ liệu" và "trò chơi vương quyền" (hapax)

Nhiều thuật ngữ trong số này viết sai chính tả của những thuật ngữ hữu ích khác, chẳng hạn như: Jaimie là Jaime (Lannister), Miliandre sẽ là Melisandre, v.v. Một trò chơi đáng hoàng của Từ điển đồng nghĩa dành riêng cho Thrones có thể giúp chúng tôi tìm và thay thế những lỗi chính tả này bằng một thuật toán tìm kiếm mở. Điều này chứng tỏ việc làm sạch dữ liệu trong khai thác văn bản có thể tiếp tục vô thời hạn nếu bạn muốn; giữ nỗ lực và kết quả cân bằng là rất quan trọng ở đây.

Bây giờ chúng ta hãy xem những từ thường xuyên nhất.

```
in wordfreqs_cat1.most_common(20)
in wordfreqs_cat2.most_common(20)
```

Hình 8.18 cho thấy kết quả của việc yêu cầu 20 từ thông dụng nhất cho mỗi loại.

#### Most frequent words within data science posts

```
print wordfreqs_cat1.most_common(20)
[(u'.', 2833), (u', 2831), (u'data', 1882), (u'?', 1190), (u'science', 887), (u')', 812), (u'(', 739), (u'"m", 566), (u':', 548), (u'would', 427), (u'"s", 323), (u'like', 321), (u'n't", 288), (u'get', 252), (u'know', 225), (u'"ve", 213), (u'scientist', 211), (u'!', 209), (u'work', 204), (u'job', 199)]
```

#### Most frequent words within Game of Thrones posts

```
print wordfreqs_cat2.most_common(20)
[(u'.', 2909), (u', 2478), (u'[', 1422), (u']', 1420), (u'?', 1139), (u'"s", 886), (u'"n't", 494), (u')', 452), (u'(', 426), (u'"ss', 399), (u':', 380), (u'"spoilers', 332), (u'"show', 325), (u'"would', 311), (u'"", 305), (u'"``', 276), (u'"think', 248), (u'"season', 244), (u'"like', 243), (u'"one', 238)]
```

Hình 8.18 Top 20 từ thường dùng nhất cho các bài đăng “khoa học dữ liệu” và “trò chơi vương quyền”

Bây giờ điều này có vẻ đáng khích lệ: một số từ phổ biến có vẻ cụ thể đối với chủ đề của chúng. Những từ như “dữ liệu”, “khoa học” và “mùa vụ” có khả năng trở thành những yếu tố phân biệt tốt. Một điều quan trọng khác cần lưu ý là sự phong phú của các thuật ngữ ký tự đơn chẳng hạn như “.” và “,”; chúng ta sẽ loại bỏ những thứ này.

Với kiến thức bổ sung này, hãy sửa lại tập lệnh chuẩn bị dữ liệu của chúng ta.

#### 8.3.6 Xem lại bước 3: Chuẩn bị dữ liệu phù hợp

Khám phá dữ liệu ngắn này đã thu hút sự chú ý của chúng tôi đến một số điều chỉnh rõ ràng chúng tôi có thể làm để cải thiện văn bản của chúng tôi. Một điều quan trọng khác là bắt nguồn từ các điều khoản.

Danh sách sau đây cho thấy một thuật toán tạo gốc đơn giản có tên là “kết hợp gốc quả cầu tuyệt”. Những người bắt quả cầu tuyệt này có thể dành riêng cho ngôn ngữ, vì vậy chúng tôi sẽ sử dụng tiếng Anh một; tuy nhiên, nó hỗ trợ nhiều ngôn ngữ.

## Liệt kê 8.6 Quá trình xử lý dữ liệu Reddit được sửa đổi sau khi khám phá dữ liệu

```

gốc = nltk.SnowballStemmer("tiếng Anh") def wordStemmer(wordrow):
    gốc = [stemmer.stem(word) cho từ trong wordrow]
    trả lại xuất phát

manual_stopwords = ['.', ',', '!', '(', ')', 'm', 'n', 't', 'eg', 've', '#', '/',
    's', '!', 'r', 'l', '=', '[', 's', '&', '%', '*', '...', '1', '2', '3', '4',
    '5', '6', '7', '8', '9', '10', '-', '...', ';', '!', ':']

def data_processing(sql,manual_stopwords): #tạo con trỏ tới dữ liệu
    c.execute(sql)
    dữ liệu = {'wordMatrix':[], 'all_words':[]}
    interWordMatrix = []
    interWordList = []

    hàng = c.fetchone() trong
    khi hàng không phải là Không có:
        tokenizer = nltk.tokenize.RegexpTokenizer(r'\w+|[^\\w\\s]+')

        wordrow = tokenizer.tokenize(row[0] + " " + row[1]) wordrow_lowercased =
       LowerCaseArray(wordrow)
        wordrow_nostopwords = wordFilter(từ dừng,wordrow_lowercased)

        wordrow_nostopwords =
        Bộ lọc từ(manual_stopwords,wordrow_nostopwords)
        wordrow_stemmed = wordStemmer(wordrow_nostopwords)

        interWordList.extend(wordrow_stemmed)
        interWordMatrix.append(wordrow_stemmed)

    hàng = c.fetchone()

    wordfreqs = nltk.FreqDist(interWordList) s hapaxes = wordfreqs.hapaxes()

    cho wordvector trong interWordMatrix:
        wordvector_nohapaxes = wordFilter(hapaxes,wordvector)
        data['wordMatrix'].append(wordvector_nohapaxes)
        data['all_words'].extend(wordvector_nohapaxes)

    trả về dữ liệu

    cho chủ đề trong subreddits:
        data[subject] = data_processing(sql='''SELECT
            chủ đềTiêu đề, chủ đềVăn bản, chủ đềThể loại TỪ chủ đề
        WHERE topicCategory = ''' + "+subject+" ''',
        manual_stopwords=manual_stopwords)

```

**Khởi tạo gốc từ thư viện NLTK.**

**Mảng từ dừng xác định các thuật ngữ cần loại bỏ/bỏ qua.**

**Bây giờ chúng tôi xác định chuẩn bị dữ liệu sửa đổi của chúng tôi.**

**Tìm nạp dữ liệu (các bài đăng trên reddit) từng cái một từ cơ sở dữ liệu SQLite.**

**Danh sách từ tạm thời được sử dụng để loại bỏ các hapaxes sau này.**

**Nhận chủ đề mới.**

**Thực hiện phân phối tần số của tất cả các điều khoản.**

**Nhận danh sách hapaxes.**

**Xóa hapaxes trong mỗi vectơ từ.**

**Mở rộng danh sách tất cả các thuật ngữ với vectơ từ đã sửa.**

**Chạy chức năng xử lý dữ liệu mới cho cả hai subreddits.**

**Lưu ý những thay đổi kể từ hàm data\_processing() cuối cùng . Tokenizer của chúng tôi bây giờ là một mã thông báo biểu thức chính quy. Biểu thức chính quy không phải là một phần của cuốn sách này và**

thường được coi là thách thức để thành thạo, nhưng tất cả những gì đơn giản này làm là cắt văn bản thành từ. Đối với các từ, bất kỳ tổ hợp chữ và số nào cũng được phép (\w), vì vậy không còn nữa ký tự đặc biệt hoặc dấu câu. Chúng tôi cũng áp dụng từ gốc và loại bỏ một danh sách các từ dừng thêm. Và, tất cả các hapaxes sẽ được loại bỏ vào cuối bởi vì mọi thứ cần phải được ngăn chặn trước. Hãy chạy lại quá trình chuẩn bị dữ liệu của chúng tôi.

Nếu chúng ta thực hiện phân tích thăm dò tương tự như trước đây, chúng ta sẽ thấy nó có ý nghĩa hơn, và chúng tôi không còn hapax nữa.

```
in wordfreqs_cat1.hapaxes()
in wordfreqs_cat2.hapaxes()
```

Hãy lấy lại 20 từ hàng đầu của mỗi loại (xem hình 8.19).

### Top 20 most common "Data Science" terms after more intense data cleansing

```
wordfreqs_cat1 = nltk.FreqDist(data['datascience']['all_words'])
print wordfreqs_cat1.most_common(20)

[(u'data', 1971), (u'scienc', 955), (u'would', 418), (u'work', 368), (u'use', 347), (u'program', 343), (u'learn', 342), (u'like', 341), (u'get', 325), (u'scientist', 310), (u'job', 268), (u'cours', 265), (u'look', 257), (u'know', 239), (u'statist', 228), (u'want', 225), (u've', 223), (u'python', 205), (u'year', 204), (u'time', 196)]
```

### Top 20 most common "Game of Thrones" terms after more intense data cleansing

```
wordfreqs_cat2 = nltk.FreqDist(data['gameofthrones']['all_words'])
print wordfreqs_cat2.most_common(20)

[(u's5', 426), (u'spoiler', 374), (u'show', 362), (u'episod', 300), (u'think', 289), (u'would', 287), (u'season', 286), (u'like', 282), (u'book', 271), (u'one', 249), (u'get', 236), (u'sansa', 232), (u'scene', 216), (u'cersei', 213), (u'know', 192), (u'go', 188), (u'king', 183), (u'throne', 181), (u'see', 177), (u'charact', 177)]
```

Hình 8.19 Top 20 từ thường gặp nhất trong các bài đăng trên Reddit "khoa học dữ liệu" và "trò chơi vương quyền" sau khi chuẩn bị dữ liệu

Chúng ta có thể thấy trong hình 8.19 chất lượng dữ liệu đã được cải thiện đáng kể như thế nào. Ngoài ra, thông báo làm thế nào một số từ được rút ngắn do chúng tôi áp dụng từ gốc. Ví dụ, "khoa học" và "khoa học" đã trở thành "khoa học;" "các khóa học" và "khóa học" đã trở thành "cours," v.v. Các thuật ngữ kết quả không phải là từ thực tế nhưng vẫn có thể hiểu được. Nếu bạn nhấn mạnh vào các thuật ngữ của mình là các từ thực tế, thì việc bổ sung từ vựng sẽ là cách đi.

Với quá trình làm sạch dữ liệu “đã hoàn thành” (nhận xét: bài tập làm sạch khai thác văn bản hầu như không bao giờ có thể được hoàn thành đầy đủ), tất cả những gì còn lại là một vài chuyển đổi dữ liệu để lấy dữ liệu ở định dạng túi từ.

Trước tiên, hãy gắn nhãn tất cả dữ liệu của chúng tôi và cũng tạo một mẫu gồm 100 quan sát mỗi danh mục, như thể hiện trong danh sách sau đây.

#### Lịch kê 8.7 Chuyển đổi dữ liệu cuối cùng và chia tách dữ liệu trước khi lập mô hình

Mẫu tổ chức bao gồm dữ liệu chưa được gắn nhãn từ hai subreddits: 100 quan sát từ mỗi tập dữ liệu. Các nhãn được giữ trong bộ dữ liệu riêng biệt.

```
holdoutLength = 100
```

```
nhãn_data1 = [(từ, 'datascience') cho từ trong
               data['datascience']['wordMatrix'][holdoutLength:]]
nhãn_data2 = [(từ, 'gameofthrones') cho từ trong
               data['gameofthrones']['wordMatrix'][holdoutLength:]]
được gắn nhãn_data = []
được gắn nhãn_data.extend(được gắn nhãn_dữ liệu1) được
gắn nhãn_data.extend(được gắn nhãn_data2)
```

```
holdout_data = data['datascience']['wordMatrix'][:holdoutLength]
holdout_data.extend(data['gameofthrones']['wordMatrix'][:holdoutLength])
holdout_data_labels = ((('datascience') cho
                       - trong xrange(holdoutLength)) + ((('gameofthrones') cho
                       xrange(holdoutLength)))
```

```
data['datascience']['all_words_dedup'] =
list(OrderedDict.fromkeys( data['datascience']
[ 'all_words'])) data['gameofthrones']['all_words_dedup'] =
list(OrderedDict.fromkeys( data ['gameofthrones'][ 'all_words']))
all_words = []
all_words.extend(data['datascience']['all_words_dedup'])

all_words.extend(data['gameofthrones']['all_words_dedup']) all_words_dedup = danh sách
(OrderedDict. fromkeys (all_words))
```

Dữ liệu cho  
người mẫu  
đào tạo  
Và  
thử nghiệm lâ  
Đầu tiên  
xáo trộn.

```
chuẩn bị_data = [{từ: (từ trong x[0]) cho từ trong all_words_dedup}, x[1])
cho x trong dữ liệu được gắn nhãn] chuẩn bị_sẵn_holdout_data = [{(từ: (từ
trong x[0]) cho từ trong all_words_dedup }) cho x trong holdout_data]
```

```
random.shuffle(prepared_data) train_size =
int(len(prepared_data) * 0,75) train = chuẩn bị_data[:train_size]
test = chuẩn bị_data[train_size:]
```

Mẫu tổ chức sẽ được sử dụng để xác định lỗi của mô hình bằng cách xây dựng ma trận nhầm lẫn.

Chúng tôi tạo một tập dữ liệu duy nhất với mọi vectơ từ được gắn thẻ là ‘khoa học dữ liệu’ hoặc ‘gameofthrones’. Chúng tôi giữ một phần dữ liệu sang một bên cho mẫu nắm giữ.

- TRONG

Một danh sách tất cả các thuật ngữ duy nhất được tạo để xây dựng túi dữ liệu từ chúng tôi như câu đàm thoại hoặc chấm điểm một mô hình.

Dữ liệu được biến thành một túi định dạng túi nhị phân.

Kích thước của dữ liệu đào tạo sẽ là 75% tổng số và 25% còn lại sẽ được sử dụng để kiểm tra hiệu suất của mô hình.

Mẫu giữ lại sẽ được sử dụng cho thử nghiệm cuối cùng của chúng tôi về mô hình và tạo ra một ma trận hỗn loạn. Ma trận nhằm lẩn là một cách để kiểm tra xem một mô hình đã hoạt động tốt như thế nào trên dữ liệu chưa từng thấy trước đó. Ma trận cho thấy có bao nhiêu quan sát là chính xác và phân loại không chính xác.

Trước khi tạo hoặc đào tạo và kiểm tra dữ liệu, chúng ta cần thực hiện một bước cuối cùng: rót dữ liệu vào một định dạng túi từ trong đó mọi thuật ngữ được cung cấp "Đúng" hoặc "Sai" nhẫn tùy thuộc vào sự hiện diện của nó trong bài đăng cụ thể đó. Chúng ta cũng cần phải làm điều này cho mẫu holdout không được dán nhãn.

Dữ liệu đã chuẩn bị của chúng ta hiện chứa mọi số hạng cho mỗi vector, như thể hiện trong hình 8.20.

in đã chuẩn bị\_data[0]

```
print prepared_data[0]
({u'sunspear': False, u'profici': False, u'pardon': False, u'selye
s': False, u'four': False, u'davo': False, u'sleev': False, u'slee
:
u'daeron': False, u'portion': False, u'emerg': False, u'fifti': Fals
e, u'decemb': False, u'defend': False, u'sincer': False}, 'datascien
ce')
```

Hình 8.20 Một túi nhị phân các từ đã sẵn sàng để lập mô hình là dữ liệu rất thưa thớt.

Chúng tôi đã tạo ra một ma trận lớn nhưng thưa thớt, cho phép chúng tôi áp dụng các kỹ thuật từ chương 5 nếu nó quá lớn để xử lý trên máy của chúng tôi. Tuy nhiên, với một chiếc bàn nhỏ như vậy, không cần điều đó ngay bây giờ và chúng tôi có thể tiến hành trộn và chia dữ liệu thành một khóa đào tạo và Tập kiểm tra.

Mặc dù phần lớn nhất trong dữ liệu của bạn phải luôn được chuyển đến quá trình đào tạo mô hình, nhưng vẫn tồn tại một tỷ lệ phân chia tối ưu. Ở đây, chúng tôi đã chọn chia tỷ lệ 3-1, nhưng hãy thoải mái chơi với điều này. Các bạn có nhiều quan sát hơn, bạn càng có nhiều tự do hơn ở đây. Nếu bạn có ít quan sát, bạn sẽ cần phân bổ tương đối nhiều hơn cho việc huấn luyện mô hình. Bây giờ chúng tôi đã sẵn sàng để chuyển sang phần bổ ích nhất: phân tích dữ liệu.

### 8.3.7 Bước 5: Phân tích dữ liệu

Đối với phân tích của chúng tôi, chúng tôi sẽ điều chỉnh hai thuật toán phân loại cho dữ liệu của mình: Naïve Bayes và quyết định cây. Naïve Bayes đã được giải thích trong chương 3 và cây quyết định trước đó trong chương này.

Trước tiên, hãy kiểm tra hiệu suất của bộ phân loại Naïve Bayes của chúng tôi. NLTK đi kèm với một phân loại, nhưng hãy thoải mái sử dụng các thuật toán từ các gói khác như SciPy.

bộ phân loại = nltk.NaiveBayesClassifier.train(đào tạo)

Với trình phân loại được đào tạo, chúng tôi có thể sử dụng dữ liệu thử nghiệm để đo lường độ chính xác tổng thể.

nltk.classify.accuracy(phân loại, kiểm tra)

```
nltk.classify.accuracy(classifier, test)
0.9681528662420382
```

Hình 8.21 Độ chính xác của phân loại là thước đo thể hiện phần trăm quan sát được phân loại chính xác trên dữ liệu thử nghiệm.

Độ chính xác của dữ liệu thử nghiệm được ước tính là lớn hơn 90%, như thể hiện trong hình 8.21. Độ chính xác phân loại là số quan sát được phân loại chính xác theo tỷ lệ phần trăm trong tổng số quan sát. Tuy nhiên, hãy lưu ý rằng điều này có thể khác trong trường hợp của bạn nếu bạn sử dụng dữ liệu khác nhau.

```
nltk.classify.accuracy(phân loại, kiểm tra)
```

Đó là một con số tốt. Bây giờ chúng ta có thể ngả lưng và thư giãn, phải không? Không thật sự lầm. Hãy kiểm tra lại nó trên mẫu giữ lại 200 quan sát và lần này tạo ra một ma trận nhầm lẫn.

```
dã phân loại_data = classifier.classify_many(prepared_holdout_data)
cm = nltk.ConfusionMatrix(holdout_data_labels,Classified_data)
in cm
```

Ma trận nhầm lẫn trong hình 8.22 cho chúng ta thấy 97% có lẽ là vượt trội bởi vì chúng tôi có 28 (23 + 5) trường hợp bị phân loại sai. Một lần nữa, điều này có thể khác với dữ liệu bạn tự điền vào tệp SQLite.

	g	
	a	
d	m	
a	e	
t	o	
a	f	
s	t	
c	h	
i	r	
e	o	
n	n	
c	e	
e	s	
-----+-----+		
datascience	<77>23	
gameofthrones	5<95>	
-----+-----+		
(row = reference; col = test)		

Hình 8.22 Ma trận nhầm lẫn  
mô hình Naïve Bayes cho thấy 28  
(23 + 5) quan sát trong số 200  
quan sát bị phân loại sai

28 phân loại sai có nghĩa là chúng tôi có độ chính xác 86% trên mẫu giữ lại. Điều này cần được so sánh với việc chỉ định ngẫu nhiên một bài đăng mới cho "khoa học dữ liệu" hoặc nhóm "gameofthrones". Nếu chúng tôi chỉ định chúng một cách ngẫu nhiên, chúng tôi có thể mong đợi một

độ chính xác là 50% và mô hình của chúng tôi dường như hoạt động tốt hơn thế. Hãy nhìn vào những gì nó sử dụng để xác định các danh mục bằng cách đào sâu vào các tính năng mô hình nhiều thông tin nhất.

```
in(classifier.show_most_informative_features(20))
```

Hình 8.23 cho thấy 20 thuật ngữ hàng đầu có khả năng phân biệt giữa hai loại.

#### Most Informative Features

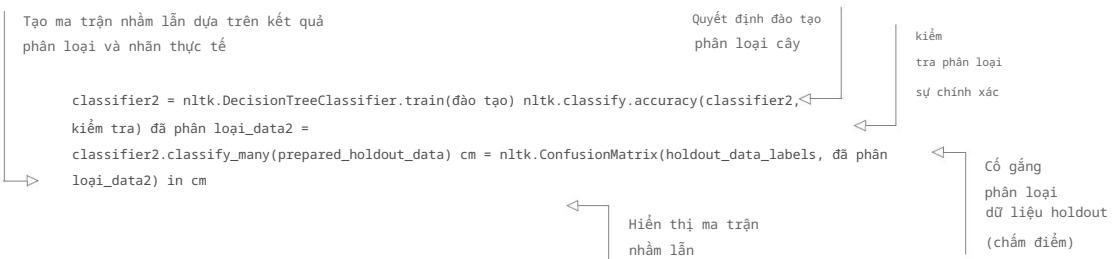
data = True	datasc : gameof =	365.1 : 1.0
scene = True	gameof : datasc =	63.8 : 1.0
season = True	gameof : datasc =	62.4 : 1.0
king = True	gameof : datasc =	47.6 : 1.0
tv = True	gameof : datasc =	45.1 : 1.0
kill = True	gameof : datasc =	31.5 : 1.0
compani = True	datasc : gameof =	28.5 : 1.0
analysi = True	datasc : gameof =	27.1 : 1.0
process = True	datasc : gameof =	25.5 : 1.0
appli = True	datasc : gameof =	25.5 : 1.0
research = True	datasc : gameof =	23.2 : 1.0
episod = True	gameof : datasc =	22.2 : 1.0
market = True	datasc : gameof =	21.7 : 1.0
watch = True	gameof : datasc =	21.6 : 1.0
man = True	gameof : datasc =	21.0 : 1.0
north = True	gameof : datasc =	20.8 : 1.0
hi = True	datasc : gameof =	20.4 : 1.0
level = True	datasc : gameof =	19.1 : 1.0
learn = True	datasc : gameof =	16.9 : 1.0
job = True	datasc : gameof =	16.6 : 1.0

Hình 8.23 Các thuật ngữ quan trọng nhất trong mô hình phân loại Naïve Bayes

Thuật ngữ “dữ liệu” được coi trọng và dường như là chỉ số quan trọng nhất của chủ đề có thuộc danh mục khoa học dữ liệu hay không. Các thuật ngữ như “cảnh”, “mùa,” “king”, “tv” và “kill” là những dấu hiệu tốt cho thấy chủ đề là Game of Thrones chứ không phải khoa học dữ liệu. Tất cả những điều này đều có ý nghĩa hoàn hảo, vì vậy mô hình đã vượt qua cả kiểm tra độ chính xác và độ chính xác.

Naïve Bayes hoạt động tốt, vì vậy chúng ta hãy xem cây quyết định trong danh sách sau đây.

#### Lịch kê 8.8 Huấn luyện và đánh giá mô hình cây quyết định



```
nltk.classify.accuracy(classifier2, test)
```

```
0.9333333333333333
```

Hình 8.24 Độ chính xác của mô hình cây quyết định

Như thể hiện trong hình 8.24, độ chính xác được đưa ra là 93%.

Bây giờ chúng ta biết rõ hơn là chỉ dựa vào thử nghiệm đơn lẻ này, vì vậy, một lần nữa chúng ta chuyển sang ma trận nhầm lẫn trên tập dữ liệu thứ hai, như thể hiện trong hình 8.25.

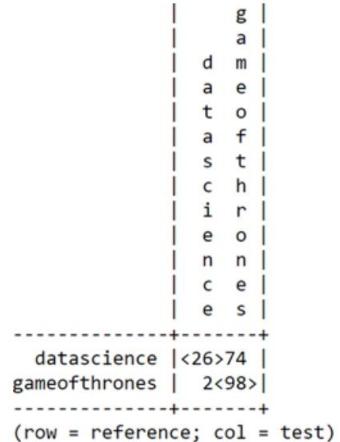
Hình 8.25 cho thấy một câu chuyện khác. Trên 200 quan sát mẫu lưu giữ này, mô hình cây quyết định có xu hướng phân loại tốt khi bài đăng về Game of Thrones nhưng lại thất bại thảm hại khi đối mặt với các bài đăng về khoa học dữ liệu. Có vẻ như người mẫu thích Game of Thrones hơn, và bạn có thể dễ dàng cho điều đó không? Hãy xem mô hình thực tế, mặc dù trong trường hợp này chúng ta sẽ sử dụng Naïve Bayes làm mô hình cuối cùng.

```
in (phân loại2.mã giả (độ sâu = 4))
```

Cây quyết định, như tên gợi ý, là một mô hình dạng cây, như thể hiện trong hình 8.26.

Naïve Bayes xem xét tất cả các thuật ngữ và có trọng số được quy cho, nhưng mô hình cây quyết định đi qua chúng một cách tuần tự, theo đường dẫn từ gốc đến các nhánh và lá bên ngoài. Hình 8.26 chỉ hiển thị bốn lớp trên cùng, bắt đầu với thuật ngữ "dữ liệu". Nếu "dữ liệu" xuất hiện trong bài đăng, thì đó luôn là khoa học dữ liệu. Nếu không tìm thấy "dữ liệu", nó sẽ kiểm tra cụm từ "học" và cứ thế tiếp tục. Một lý do có thể khiến cây quyết định này không hoạt động tốt là do thiếu sự cắt tia. Khi một cây quyết định được xây dựng, nó có nhiều lá, thường là quá nhiều. Sau đó, một cái cây được cắt tia ở một mức độ nhất định để giảm thiểu việc trang bị quá mức.

Một lợi thế lớn của cây quyết định là hiệu ứng tương tác ngầm giữa các từ mà nó



Hình 8.25 Ma trận nhầm lẫn trên mô hình cây quyết định

```
if data == False:
    if learn == False:
        if python == False:
            if tool == False: return 'gameofthrones'
            if tool == True: return 'datascience'
        if python == True: return 'datascience'
    if learn == True:
        if go == False:
            if wrong == False: return 'datascience'
            if wrong == True: return 'gameofthrones'
        if go == True:
            if upload == False: return 'gameofthrones'
            if upload == True: return 'datascience'
    if data == True: return 'datascience'
```

Hình 8.26 Biểu diễn cấu trúc cây mô hình cây quyết định

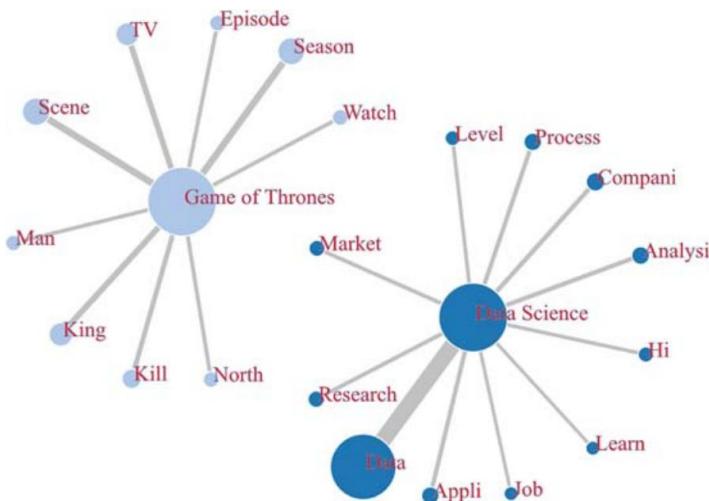
tính đến khi xây dựng các nhánh. Khi nhiều điều khoản với nhau tạo ra một phân loại mạnh hơn so với các điều khoản đơn lẻ, cây quyết định sẽ thực sự vượt trội hơn so với Naïve Bayes. Chúng tôi sẽ không đi vào chi tiết về điều đó ở đây, nhưng hãy xem xét điều này các bước tiếp theo bạn có thể thực hiện để cải thiện mô hình.

Hiện tại, chúng tôi có hai mô hình phân loại giúp chúng tôi hiểu rõ hơn về sự khác biệt giữa hai nội dung của các subreddits. Bước cuối cùng là chia sẻ thông tin mới tìm được này với những người khác.

### 8.3.8 Bước 6: Trình bày và tự động hóa

Bước cuối cùng, chúng ta cần sử dụng những gì đã học và biến nó thành một ứng dụng hữu ích hoặc trình bày kết quả của chúng tôi cho người khác. Chương cuối cùng của cuốn sách này thảo luận về việc xây dựng một ứng dụng tương tác, vì bản thân đây là một dự án. Bây giờ chúng tôi sẽ hài lòng với một tốt đẹp cách để truyền đạt những phát hiện của chúng tôi. Một biểu đồ đẹp hoặc tốt hơn nữa là một biểu đồ tương tác, có thể nắm bắt mắt; đó là lớp kem trên chiếc bánh thuyết trình. Mặc dù thật dễ dàng và hấp dẫn để đại diện cho các con số như vậy hoặc nhiều nhất là biểu đồ thanh, có thể tốt hơn nếu tiến thêm một bước nữa.

Ví dụ, để biểu diễn mô hình Naïve Bayes, chúng ta có thể sử dụng biểu đồ lực (hình 8.27), trong đó kích thước bong bóng và liên kết biểu thị mức độ liên quan chặt chẽ của một từ với subreddits "trò chơi vương quyền" hoặc "khoa học dữ liệu". Lưu ý cách các từ trên bong bóng thường bị cắt bỏ; hãy nhớ rằng điều này là do chúng tôi đã áp dụng từ gốc.



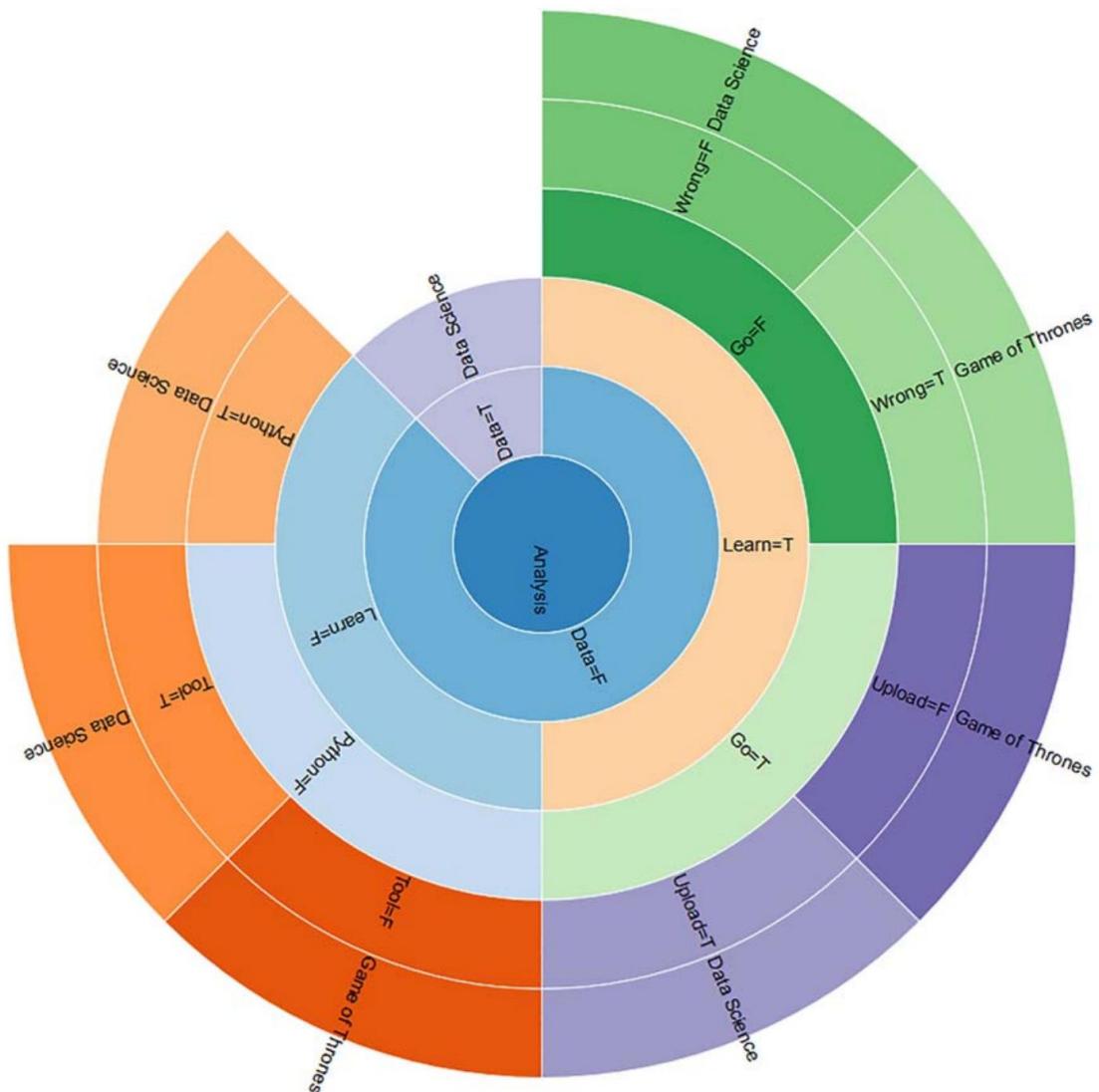
Hình 8.27 Biểu đồ lực tương tác với 20 số hạng có nghĩa hàng đầu của Naïve Bayes và trọng số của chúng

Trong khi bản thân hình 8.27 là tĩnh, bạn có thể mở tệp HTML "forceGraph.html" để tận hưởng hiệu ứng biểu đồ lực lượng d3.js như đã giải thích trước đó trong chương này. d3.js nằm ngoài phạm vi của cuốn sách này nhưng bạn không cần phải có kiến thức phức tạp về d3.js để sử dụng nó. MỘT tập hợp các ví dụ mở rộng có thể được sử dụng với các điều chỉnh tối thiểu đối với mã được cung cấp tại <https://github.com/mbostock/d3/wiki/Gallery>. Tất cả những gì bạn cần là ý thức chung và

kiến thức nhỏ về JavaScript. Bạn có thể tìm thấy mã cho ví dụ về biểu đồ lực tại <http://bl.ocks.org/mbostock/4062045>.

Chúng ta cũng có thể biểu diễn cây quyết định của mình theo một cách khá độc đáo. Chúng ta có thể sử dụng một phiên bản ưa thích của sơ đồ cây thực tế, nhưng sơ đồ sunburst sau đây nguyên bản hơn và không kém phần thú vị khi sử dụng.

Hình 8.28 cho thấy lớp trên cùng của biểu đồ sunburst. Có thể phóng to bằng cách nhấp vào một đoạn vòng tròn. Bạn có thể thu nhỏ lại bằng cách nhấp vào vòng tròn trung tâm. Bạn có thể tìm thấy mã cho ví dụ này tại <http://bl.ocks.org/metmajer/5480307>.

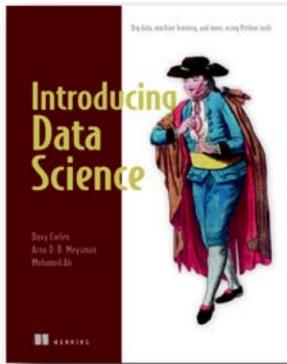


Hình 8.28 Biểu đồ Sunburst được tạo từ bốn nhánh trên cùng của mô hình cây quyết định

Hiển thị kết quả của bạn theo cách ban đầu có thể là chìa khóa cho một dự án thành công. Mọi người không bao giờ đánh giá cao nỗ lực bạn đã bỏ ra để đạt được kết quả của mình nếu bạn không thể truyền đạt chúng và chúng có ý nghĩa với họ. Một trực quan hóa dữ liệu ban đầu ở đây và chắc chắn sẽ giúp với điều này.

## 8.4 Tóm tắt

- Khai thác văn bản được sử dụng rộng rãi cho những thứ như nhận dạng thực thể, đạo văn phát hiện, xác định chủ đề, dịch thuật, phát hiện gian lận, lọc thư rác, và hơn thế nữa.
  - Python có bộ công cụ trưởng thành để khai thác văn bản được gọi là NLTK, hoặc ngôn ngữ tự nhiên bộ công cụ. NLTK rất tốt để chơi xung quanh và tìm hiểu các sơ đồ; cho cuộc sống thực tuy nhiên, các ứng dụng, Scikit-learning thường được coi là “sẵn sàng sản xuất” hơn. Scikit-learning được sử dụng rộng rãi trong các chương trước.
  - Việc chuẩn bị dữ liệu của dữ liệu văn bản chuyên sâu hơn dữ liệu số chuẩn bị và liên quan đến các kỹ thuật bổ sung, chẳng hạn như - **Bắt đầu từ–Cắt phần cuối** của một từ theo cách thông minh để từ đó có thể khớp với nhau với một số phiên bản liên hợp hoặc số nhiều của từ này.
    - Lemmatization–Giống như từ gốc, nó có nghĩa là loại bỏ các từ kép, nhưng không giống như bắt nguồn, nó nhìn vào nghĩa của từ.
    - **Dừng lọc từ–Một số** từ xuất hiện quá thường xuyên nên không hữu ích và lọc chúng ra có thể cải thiện đáng kể các mô hình. Các từ dừng thường là ngữ liệu cụ thể.
    - **Tokenization–Cắt** văn bản thành từng mảnh. Mã thông báo có thể là các từ đơn lẻ, tổ hợp các từ (n-gram) hoặc thậm chí cả câu.
    - **Gắn thẻ POS –Gắn thẻ** một phần của bài phát biểu. Đôi khi nó có thể hữu ích để biết những gì chức năng của một từ nhất định trong một câu là để hiểu nó tốt hơn.
  - Trong nghiên cứu điển hình của chúng tôi, chúng tôi đã cố gắng phân biệt các bài đăng Reddit trên “Trò chơi vương quyền” so với các bài đăng trên “khoa học dữ liệu”. Trong nỗ lực này, chúng tôi đã thử cả Naïve Bayes và phân loại cây quyết định. Naïve Bayes giả định tất cả các tính năng là độc lập của nhau; bộ phân loại cây quyết định giả định sự phụ thuộc, cho phép mô hình khác nhau.
  - Trong ví dụ của chúng tôi, Naïve Bayes mang lại mô hình tốt hơn, nhưng thường thì bộ phân loại cây quyết định thực hiện công việc tốt hơn, thường là khi có nhiều dữ liệu hơn. ■
- Chúng tôi đã xác định sự khác biệt về hiệu suất bằng cách sử dụng ma trận nhầm lẫn mà chúng tôi tính toán muộn sau khi áp dụng cả hai mô hình trên dữ liệu mới (nhưng được dán nhãn). ■ Khi trình bày những phát hiện cho người khác, có thể hữu ích khi đưa vào một trực quan hóa dữ liệu có khả năng truyền đạt kết quả của bạn một cách đáng nhớ.



Khoa học dữ liệu đã trở thành một trong những lĩnh vực nóng nhất trong công nghệ. Các công ty trên toàn thế giới đang tranh giành để tìm các nhà phát triển có kỹ năng khoa học dữ liệu để làm việc trên các dự án khác nhau, từ tiếp thị truyền thông xã hội đến học máy, nhưng kiến thức và kinh nghiệm tiên quyết cho sự nghiệp này có vẻ hoang mang. Cuốn sách này được thiết kế để giúp bạn bắt đầu.

Về cốt lõi, khoa học dữ liệu là một tập hợp các khái niệm và kỹ thuật để trích xuất ý nghĩa và sự rõ ràng từ các tập dữ liệu khổng lồ được lưu trữ hoặc các luồng dữ liệu chuyển động nhanh. Dữ liệu các nhà khoa học viết chương trình để diễn giải những dữ liệu này. Các Ngôn ngữ lập trình Python là một công cụ yêu thích của dữ liệu các nhà khoa học vì nó dễ đọc và viết, đồng thời nó cung cấp một số thư viện có giá trị cao giúp đơn giản hóa các tác vụ cốt lõi như thống kê, thuật toán máy học và toán học.

### Có gì bên trong

- Làm quen với các khái niệm khoa học dữ liệu quan trọng nhất
- Sử dụng Python để làm việc với dữ liệu phổ biến—và không quá phổ biến—lưu trữ định dạng
- Viết thuật toán bằng Python
- Sử dụng các công cụ Python như iPython để hiểu dữ liệu lớn
- Trải nghiệm thực tế với các thư viện khoa học dữ liệu Python phổ biến nhất chẳng hạn như Scikit-learn và StatsModels
- Sử dụng khoa học dữ liệu trong thế giới dữ liệu lớn

Cuốn sách này giả định rằng bạn cảm thấy thoải mái khi đọc mã bằng Python hoặc một ngôn ngữ tương tự, chẳng hạn như C, Ruby hoặc JavaScript. Không yêu cầu kinh nghiệm trước về khoa học dữ liệu.

# phụ thuộc mô hình với Bayesian và mạng Markov

xác suất các mô hình mạng phù hợp với các nhiệm vụ mà bạn phải suy ra nhiều trạng thái bên trong (hoặc không quan sát được) của thế giới từ các quan sát bên ngoài, hoặc khi bạn muốn mô phỏng các quy trình cho kịch bản “nếu như”. Chương sau đây cung cấp phần giới thiệu tuyệt vời về các mô hình mạng và lý luận xác suất làm nền tảng cho chúng. Các chi tiết thực hiện có thể khó khăn để làm theo nếu bạn không quen thuộc với ngôn ngữ lập trình xác suất Figaro, nhưng các ví dụ thúc đẩy giải thích rõ ràng các khái niệm cơ bản. Bạn cũng sẽ hiểu rõ về các loại quy trình ra quyết định mà Figaro có thể đại diện bức bối.

Chương 5 từ Lập trình xác suất thực tế  
của Avi Pfeffer

# phụ thuộc mô hình với Bayesian và mạng Markov

## Chương này bao gồm

- Các loại mối quan hệ giữa các biến trong một mô hình xác suất và cách thức các mối quan hệ này chuyển thành các yếu tố phụ thuộc
- Cách thể hiện các loại phụ thuộc khác nhau này trong Figaro
- Mạng Bayes: các mô hình mã hóa sự phụ thuộc có hướng giữa các biến
- Mạng Markov: các mô hình mã hóa phụ thuộc vô hướng giữa các biến
- Ví dụ thực tế về mạng Bayesian và Markov

Trong chương 4, bạn đã học về mối quan hệ giữa các mô hình xác suất và các chương trình xác suất, và bạn cũng đã thấy các thành phần của một mô hình xác suất, đó là các biến, phụ thuộc, dạng hàm và tham số số.

Chương này tập trung vào hai khung mô hình hóa: mạng Bayesian và Markov các mạng. Mỗi khung dựa trên một cách mã hóa phụ thuộc khác nhau.

Phụ thuộc nắm bắt mối quan hệ giữa các biến. Tìm hiểu các loại của các mối quan hệ và cách chúng chuyển thành các phụ thuộc trong một mô hình xác suất

là một trong những kỹ năng quan trọng nhất mà bạn có thể có được để xây dựng mô hình. Theo đó, bạn sẽ tìm hiểu tất cả về các loại mối quan hệ và sự phụ thuộc. Nói chung, có hai loại phụ thuộc giữa các biến: phụ thuộc trực tiếp, mà biểu thị các mối quan hệ bất đối xứng và các phụ thuộc vô hướng, biến thành các mối quan hệ số liệu đối xứng. Các mạng Bayes mã hóa các phụ thuộc có hướng, trong khi Markov mạng mã hóa các phụ thuộc vô hướng. Bạn cũng sẽ học cách mở rộng các mạng Bayesian truyền thống với các khả năng của ngôn ngữ lập trình để hưởng lợi từ sức mạnh của lập trình xác suất.

Sau khi bạn hiểu tài liệu trong chương này, bạn sẽ có kiến thức vững chắc về các yếu tố cần thiết của lập trình xác suất. Tất cả các mô hình xác suất đều rút gọn thành một tập hợp các phụ thuộc có hướng và không có hướng. Bạn sẽ biết khi nào nên giới thiệu một sự phụ thuộc giữa các biến, cho dù nó được định hướng hay vô hướng, và, nếu nó được định hướng, nó nên đi theo hướng nào. Chương 6 dựa trên kiến thức này để tạo ra các mô hình phức tạp hơn bằng cách sử dụng cấu trúc dữ liệu và chương 7 sẽ tiếp tục mở rộng kỹ năng của bạn với mô hình hướng đối tượng.

Chương này giả định rằng bạn đã có kiến thức cơ bản về Figaro, như đã xuất hiện trong chương 2. Đặc biệt, bạn nên làm quen với Chain, là cơ sở để định hướng phụ thuộc, điều kiện và ràng buộc, là cơ sở cho vô hướng phụ thuộc. Bạn cũng sẽ sử dụng các phần tử CPD và RichCPD xuất hiện trong chương 4. Đừng lo lắng nếu bạn không nhớ những khái niệm này; Tôi sẽ nhắc bạn về chúng ở đây khi bạn nhìn thấy chúng.

## 5.1 Phụ thuộc mô hình hóa

Lý luận xác suất là tất cả về việc sử dụng các phụ thuộc giữa các biến. Hai biến là phụ thuộc nếu kiến thức về một biến cung cấp thông tin về biến kia. Biến đối. Ngược lại, nếu biết điều gì đó về một biến không cho bạn biết gì về biến kia, các biến là độc lập.

Hãy xem xét một ứng dụng chẩn đoán hệ thống máy tính mà bạn đang cố gắng suy luận về các lỗi trong quá trình in. Giả sử bạn có hai biến, Một nút nguồn máy in và Trạng thái máy in. Nếu bạn quan sát thấy nút nguồn tắt, bạn có thể suy ra rằng trạng thái máy in bị hỏng. Ngược lại, nếu bạn quan sát thấy tình trạng máy in bị hỏng, bạn có thể suy luận rằng nút nguồn có thể bị tắt. Hai biến này rõ ràng sự phụ thuộc.

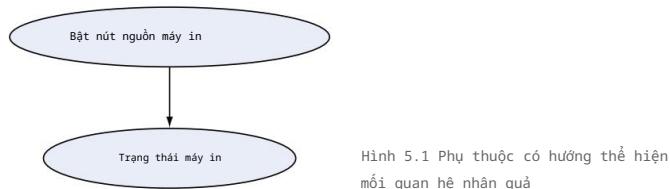
Các phụ thuộc được sử dụng để mô hình hóa các biến có liên quan theo một cách nào đó. Nhiều tồn tại các loại quan hệ giữa các biến, nhưng chỉ có hai loại phụ thuộc:

- Phụ thuộc có hướng đi từ biến này sang biến khác. Thông thường, những mô hình này quan hệ nhân quả giữa các biến.
- Mối quan hệ phụ thuộc vô hướng mô hình giữa các biến không có hướng ảnh hưởng rõ ràng giữa chúng.

Hai tiêu mục tiếp theo mô tả chi tiết cả hai loại phụ thuộc và đưa ra rất nhiều ví dụ.

### 5.1.1 Phụ thuộc có hướng

Các phụ thuộc có hướng dẫn từ biến này sang biến khác và thường được sử dụng để biểu diễn các mối quan hệ nhân quả. Ví dụ: nút nguồn của máy in bị tắt khiến máy in ngừng hoạt động, do đó tồn tại sự phụ thuộc trực tiếp giữa Bật nút nguồn máy in và Trạng thái máy in. Hình 5.1 minh họa sự phụ thuộc này, với một cạnh có hướng giữa hai biến. (Cạnh là thuật ngữ thông thường cho một mũi tên giữa hai nút trong biểu đồ.) Biến đầu tiên (trong trường hợp này là Bật nút nguồn máy in) được gọi là biến chính và biến thứ hai (Trạng thái máy in) được gọi là biến con.



Hình 5.1 Phụ thuộc có hướng thể hiện  
mối quan hệ nhân quả

Tại sao mũi tên đi từ nguyên nhân đến kết quả? Một lý do đơn giản là các nguyên nhân có xu hướng xảy ra trước các kết quả. Sâu xa hơn, câu trả lời có liên quan chặt chẽ với khái niệm mô hình tổng quát được khám phá trong chương 4. Hãy nhớ rằng, mô hình tổng quát mô tả một quá trình tạo ra các giá trị của tất cả các biến trong mô hình của bạn. Thông thường, quy trình tổng quát mô phỏng một quy trình trong thế giới thực. Nếu một nguyên nhân dẫn đến một kết quả, trước tiên bạn muốn tạo giá trị của nguyên nhân và sử dụng giá trị đó khi bạn tạo kết quả. Trong ví dụ của chúng tôi, nếu bạn tạo một mô hình máy in và tưởng tượng việc tạo các giá trị cho tất cả các biến trong mô hình, trước tiên bạn sẽ tạo một giá trị cho Bật nút nguồn máy in và sau đó sử dụng giá trị này để tạo Trạng thái máy in.

Bây giờ, cần nhắc lại rằng hướng của sự phụ thuộc không nhất thiết phải là hướng của lý luận. Bạn có thể suy luận từ nút nguồn máy in đang tắt đến trạng thái máy in không hoạt động, nhưng bạn cũng có thể suy luận theo hướng ngược lại: nếu trạng thái máy in hoạt động, bạn biết chắc rằng nút nguồn không tắt. Nhiều người mắc sai lầm khi xây dựng các mô hình của họ theo hướng mà họ dự định lập luận. Trong một ứng dụng chẩn đoán, bạn có thể quan sát thấy rằng máy in không hoạt động và cố gắng xác định nguyên nhân của nó, vì vậy bạn sẽ suy luận từ Trạng thái máy in sang Bật nút nguồn máy in. Bạn có thể muốn làm cho mũi tên chuyển từ Trạng thái máy in sang Bật nút nguồn máy in. Điều này là không chính xác. Mũi tên phải thể hiện quá trình sinh sản, theo sau nguyên nhân

hướng tác dụng.

Tôi đã nói rằng các phụ thuộc có hướng thường mô hình hóa các mối quan hệ nhân quả. Trên thực tế, nguyên nhân-kết quả chỉ là một ví dụ về một loại chung của các mối quan hệ bất đối xứng giữa các biến. Chúng ta hãy xem xét kỹ hơn các loại tàu có quan hệ bất đối xứng–đầu tiên là quan hệ nhân quả, sau đó là các loại khác.

**CÁC MỐI QUAN HỆ NHÂN QUẢ**

Dưới đây là một số loại mối quan hệ nhân quả:

- Điều gì xảy ra trước với điều xảy ra tiếp theo—Loại quan hệ nhân quả rõ ràng nhất là giữa một điều dẫn đến một điều khác sau đó. Ví dụ, nếu ai đó tắt nguồn máy in, thì sau đó, máy in sẽ ngừng hoạt động. Mối quan hệ thời gian này là một đặc điểm chung của mối quan hệ nhân quả mà bạn có thể nghĩ rằng tất cả các mối quan hệ nhân quả đều liên quan đến thời gian, nhưng tôi không đồng ý với điều này.
- Nguyên nhân-kết quả của các trạng thái—Đôi khi bạn có thể có hai biến đại diện cho các khía cạnh khác nhau của trạng thái của tình huống tại một thời điểm nhất định. Ví dụ: bạn có thể có một biến đại diện cho việc nút nguồn của máy in có tắt hay không và một biến khác đại diện cho việc máy in có bị hỏng hay không. Cả hai đều là những trạng thái tồn tại cùng một lúc. Trong ví dụ này, nút nguồn máy in bị tắt khiến máy in không hoạt động, vì nó làm cho máy in không có nguồn.
- Giá trị thực để đo lường—Bắt đầu khi nào một biến là thước đo của giá trị của một biến khác, bạn nói rằng giá trị thực là nguyên nhân của phép đo. Ví dụ: giả sử bạn có biến Đèn báo nguồn sáng cho biết đèn LED nguồn của máy in có sáng hay không. Mối quan hệ không đối xứng tồn tại từ Bật nút nguồn máy in đến Đèn báo nguồn sáng. Thông thường, các phép đo được tạo ra bởi các cảm biến và có thể có nhiều hơn một phép đo có cùng giá trị. Ngoài ra, các phép đo thường được quan sát và bạn muốn suy luận từ các phép đo thành giá trị thực, vì vậy đây là một ví dụ khác về hướng của các yếu tố phụ thuộc khác với hướng suy luận. ■ Tham số cho biến sử dụng tham số—Ví dụ: xem xét độ lệch của một đồng xu, biểu thị xác suất mà một lần tung sẽ ra mặt ngửa, và một lần tung đồng xu đó. Việc tung sử dụng độ lệch để xác định kết quả. Rõ ràng là

xu hướng được tạo ra trước tiên và chỉ sau đó là tung cá nhân. Và khi có nhiều lần tung cùng một đồng xu, tất cả chúng đều được tạo ra sau xu hướng.

**CÁC MỐI QUAN HỆ BỎ SUNG KHÔNG ĐỐI XỨNG** Các trường hợp

trước cho đến nay là quan trọng nhất và ít mơ hồ nhất. Nếu bạn hiểu những trường hợp này, bạn sẽ đi được 95% trong việc xác định hướng phụ thuộc chính xác. Bây giờ chúng ta hãy đi sâu hơn bằng cách xem xét một loạt các mối quan hệ khác, mặc dù rõ ràng là không đối xứng, nhưng lại mơ hồ về hướng phụ thuộc.

Tôi sẽ liệt kê các mối quan hệ này và sau đó mô tả một quy tắc chung có thể giúp bạn giải quyết sự mơ hồ.

- Từng phần—Thông thường, các thuộc tính của một phần của đối tượng dẫn đến các thuộc tính của toàn bộ đối tượng. Ví dụ, hãy xem xét một máy in có mực và khay nạp giấy. Các lỗi của bộ nạp mực hoặc giấy, là các bộ phận của máy in, có thể dẫn đến lỗi toàn bộ máy in. Những lần khác, các thuộc tính của toàn bộ có thể xác định các thuộc tính của một phần. Ví dụ, nếu máy in

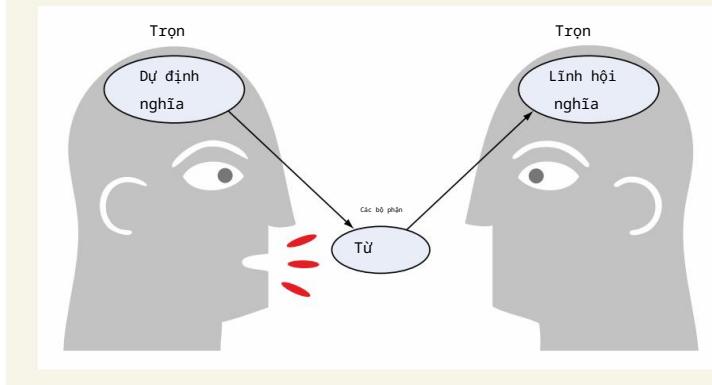
nói chung là kém, cả khay nạp giấy và hộp mực đều có khả năng bị hỏng. cách làm tồi.

- Cụ thể đến chung—Cái này cũng có thể đi cả hai chiều. Người dùng có thể trải nghiệm một sự cố máy in theo nhiều cách, chẳng hạn như kẹt giấy hoặc chất lượng in kém. Nếu người dùng gặp phải bất kỳ sự cố cụ thể nào trong số này, người dùng đó sẽ gặp phải vấn đề chung hơn về trải nghiệm in kém. Trong trường hợp này, cụ thể gây ra cái chung. Mặt khác, hãy tưởng tượng quá trình tạo ra một sự vật. Thông thường, bạn tạo các thuộc tính chung của đối tượng trước khi tinh chỉnh chúng với các chi tiết cụ thể. Ví dụ: khi tạo máy in, trước tiên bạn có thể quyết định xem đó là máy in laser hay máy in phun trước khi tạo riêng của cải. Thật vậy, không có ý nghĩa gì khi tạo ra các thuộc tính cụ thể, có thể chỉ phù hợp với một loại máy in cụ thể, trước khi bạn biết đó là loại máy in nào.
- Cụ thể/chi tiết đến trừu tượng/tóm tắt—Một ví dụ về mối quan hệ cụ thể-trừu tượng là giữa điểm bài kiểm tra và điểm chữ. Nhiều điểm số tương ứng đến cùng một lớp chữ cái. Rõ ràng, giáo viên căn cứ vào điểm chữ trong bài kiểm tra điểm chữ không phải ngược lại, vì vậy điểm bài kiểm tra gây ra điểm chữ. TRÊN mặt khác, hãy xem xét quá trình tạo ra một học sinh và kết quả kiểm tra. Trước tiên, bạn có thể tạo loại sinh viên trừu tượng (ví dụ: sinh viên A hoặc sinh viên B), sau đó tạo điểm kiểm tra cụ thể.

### Phân biệt mối quan hệ nhân quả

Như bạn có thể thấy từ các ví dụ trước, mặc dù rõ ràng là một bối cảnh xứng sự phụ thuộc tồn tại trong những trường hợp này, trêu chọc hướng của sự phụ thuộc có thể trả nên khó khăn. Đây là một ý tưởng có thể giúp đỡ. Hãy tưởng tượng rằng ai đó đang nói một câu bằng tiếng Anh cho người khác, như thể hiện trong hình minh họa sau đây. Người nói có một ý nghĩa nhất định trong tâm trí cho cả câu. Từ ý nghĩa này, người nói tạo ra các từ. Đây là một mối quan hệ toàn bộ, từ câu đến từ của nó.

Sau đó, câu được nghe bởi người khác. Người này tập hợp ý nghĩa nhận thức của câu từ các từ, tạo ra mối quan hệ một phần-toàn bộ.



Truyền thông của một câu: ý nghĩa dự định của người nói về toàn bộ câu tạo ra các từ (các phần), từ đó tạo ra ý nghĩa nhận thức của người nghe về toàn bộ câu.

## (còn tiếp)

Nếu bạn xem kỹ ví dụ này, bạn có thể thấy rằng trong quá trình tạo câu, nghĩa của cả câu được tạo ra trước các từ riêng lẻ.

Nhưng trong quá trình linh hôi câu, nghĩa của câu được linh hôi sau các từ riêng lẻ. Đây không phải là một quy tắc cứng nhắc, nhưng thường thì khi bạn làm một thứ gì đó, trước tiên bạn tạo ra cái chung/trùu tượng/toàn bộ và sau đó tinh chỉnh nó để tạo ra thứ cụ thể/cụ thể/chi tiết/nhiều phần. Khi bạn tạo một máy in, trước tiên bạn tạo lớp chung của toàn bộ máy in. Sau đó, bạn tạo loại máy in cụ thể và cả thông tin chi tiết về các thành phần của máy in. Tương tự, khi tạo học sinh, trước tiên bạn tạo lớp trùu tượng của học sinh và sau đó điền vào điểm kiểm tra cụ thể. Mặt khác, khi bạn nhận thức và báo cáo một cái gì đó, trước tiên bạn nhận thức thông tin cụ thể/cụ thể/chi tiết về các bộ phận của nó, sau đó rút ra và tóm tắt các thuộc tính chung/trùu tượng về toàn bộ của nó. Ví dụ: người dùng gặp sự cố máy in cụ thể có thể tóm tắt sự cố đó theo cách chung chung hoặc giáo viên chấm điểm học sinh sẽ quan sát điểm bài kiểm tra trước khi báo cáo điểm chữ cái. Vì vậy, đây là quy tắc ngón tay cái:

Nếu bạn đang lập mô hình việc tạo hoặc định nghĩa các thuộc tính, các yếu tố phụ thuộc sẽ đi từ khái niệm chung, trùu tượng hoặc toàn bộ đến khái niệm cụ thể, cụ thể hoặc các bộ phận. Nếu bạn đang lập mô hình nhận thức và báo cáo về các thuộc tính, thì các yếu tố phụ thuộc sẽ đi từ cụ thể, cụ thể hoặc các bộ phận đến tổng quát, trùu tượng hoặc toàn bộ.

Như tôi đã nói trước đó, nếu bạn hiểu các mối quan hệ nhân quả chính, thì hầu như lúc nào bạn cũng sẽ nhận được chỉ dẫn đúng đắn. Trong các trường hợp ngoại lệ đối với quy tắc này, ngay cả những người lập mô hình có kinh nghiệm cũng có thể không đồng ý về hướng thích hợp của các thành phần phụ thuộc. Tôi hy vọng rằng quy tắc ngón tay cái mà tôi đã cung cấp cho bạn sẽ cung cấp một số hướng dẫn để giúp bạn xây dựng các mô hình.

**SỰ PHỤ THUỘC CÓ ĐỊNH HƯỚNG**

**TRONG FIGARO** Hãy nhớ bốn thành phần của một mô hình xác suất? Chúng là các biến, các thuộc tính phụ thuộc, các dạng hàm và các tham số số. Cho đến bây giờ, bạn đã giả định rằng các biến được đưa ra và tập trung vào các thành phần phụ thuộc. Khi bạn muốn thể hiện các phụ thuộc này trong Figaro, bạn cần cung cấp một dạng chức năng và chỉ định các tham số số.

Bạn có thể thể hiện các phụ thuộc có hướng trong Figaro theo nhiều cách khác nhau. Nguyên tắc chung là sử dụng một Chuỗi nào đó làm dạng chức năng. Hãy nhớ rằng Chuỗi có hai đối số: ■

Phần tử cha ■ Hàm chuỗi

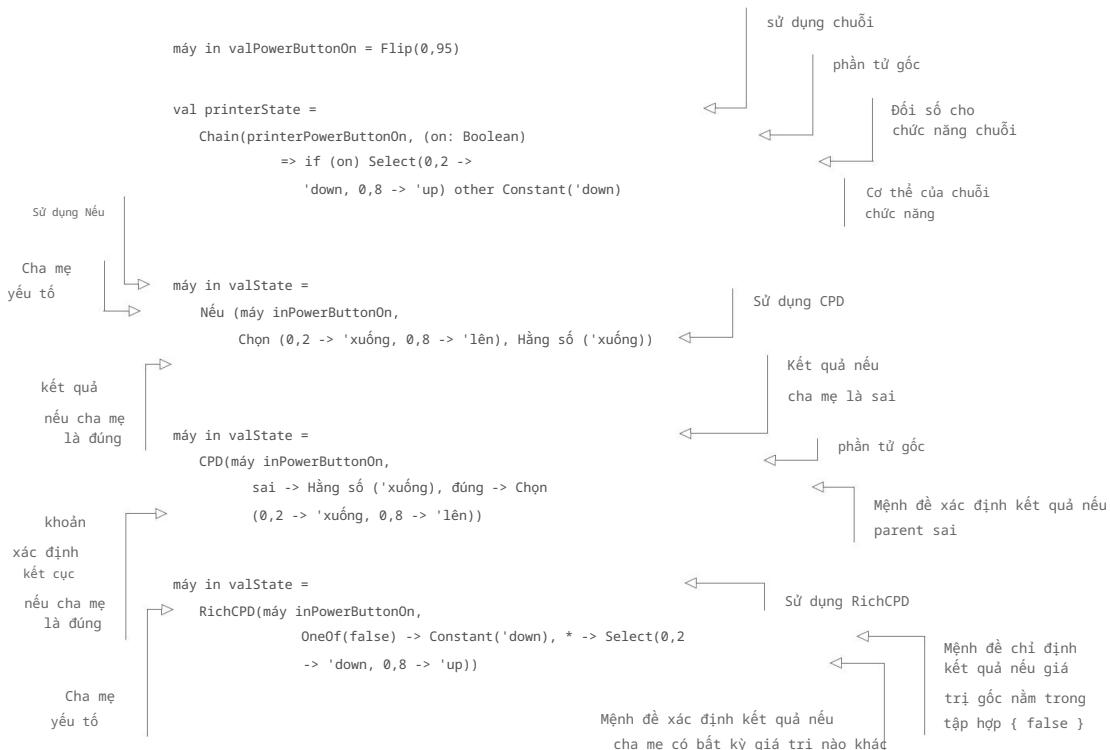
từ giá trị của phần tử cha đến phần tử kết quả

Chain(parentElement, chainFunction) xác định quy trình tổng quát sau: lấy giá trị cho cha mẹ từ parentElement, sau đó áp dụng chainFunction để lấy resultElement và cuối cùng nhận giá trị từ resultElement.

Khi thể hiện một phụ thuộc có hướng bằng cách sử dụng Chain , parentElement đương nhiên là cha mẹ. Hàm chuỗi chỉ định phân phối xác suất trên

con cho mỗi giá trị của cha mẹ. Figaro có rất nhiều cấu trúc sử dụng Chuỗi, vì vậy trong nhiều trường hợp bạn đang sử dụng Chuỗi ngay cả khi nó không giống như vậy.

Dưới đây là một số ví dụ tương đương, tất cả đều thể hiện rằng nếu nút nguồn máy in tắt thì máy in chắc chắn bị sập, nhưng nếu bật nút nguồn thì máy in có thể lên hoặc xuống:



Như đã đề cập trước đó, một loại quan hệ bất đối xứng là giữa một tham số và một biến phụ thuộc vào tham số, chẳng hạn như độ lệch của đồng xu và tung đồng xu đó. Một lần nữa, điều này có thể được thể hiện bằng cách sử dụng Chuỗi hoặc sử dụng phiên bản ghép của một nguyên tố nguyên tử. Dưới đây là một ví dụ tương đương:

```

val tung = Chain(bias, (d: Double) => Flip(d))

val tung = Lật (thiên vị)

```

### 5.1.2 Phụ thuộc vô hướng

Bạn đã thấy rằng các phụ thuộc có hướng có thể đại diện cho nhiều mối quan hệ bất đối xứng. Mỗi quan hệ mô hình phụ thuộc vô hướng giữa các biến trong đó không có hướng rõ ràng giữa chúng. Những loại quan hệ này được gọi là quan hệ đối xứng. Nếu bạn có các biến tương quan với nhau, nhưng không có kết quả rõ ràng

quá trình tổng quát theo đó một biến được tạo ra trước biến kia, một sự phụ thuộc vô hướng có thể phù hợp. Mỗi quan hệ đối xứng có thể phát sinh theo hai cách:

- Hai tác động của cùng một nguyên nhân, trong đó nguyên nhân không được lập mô hình rõ ràng-Ví dụ: hai phép đo có cùng giá trị, khi bạn không có biến cho giá trị hoặc hai hậu quả của cùng một sự kiện, khi bạn không có biến cho sự kiện. Rõ ràng, nếu bạn không biết giá trị cơ bản của một sự kiện, thì hai phép đo hoặc hệ quả có liên quan với nhau. Hãy tưởng tượng, trong scénario của máy in, bạn có các biến riêng biệt cho chất lượng in và tốc độ in. Nếu bạn không có một biến đại diện cho trạng thái của máy in, thì hai biến này sẽ có liên quan với nhau, bởi vì chúng là hai khía cạnh của quá trình in có thể có cùng một nguyên nhân cơ bản.

Bạn có thể hỏi, tại sao chúng ta không bao gồm một biến nguyên nhân trong mô hình của mình? Một câu trả lời có thể là nguyên nhân phức tạp hơn nhiều so với các hiệu ứng và sẽ khó mô hình hóa chính xác. Trong chương này, bạn sẽ thấy một ví dụ về tái tạo hình ảnh. Hình ảnh là hiệu ứng hai chiều của một cảnh ba chiều phức tạp. Có thể khó tạo mô hình xác suất chính xác của cảnh ba chiều hơn là tạo mô hình mối quan hệ giữa các pixel trong ảnh.

- Hai nguyên nhân của cùng một hệ quả đã biết-Điều này thật thú vị. Thông thường, không có mối quan hệ nào giữa hai nguyên nhân của cùng một hệ quả. Ví dụ: trạng thái khay nạp giấy của máy in và mức bột mực đều ảnh hưởng đến toàn bộ trạng thái của máy in, nhưng trạng thái khay nạp giấy và mức bột mực là độc lập. Nhưng nếu bạn biết rằng máy in không in tốt, độ nhiên tình trạng khay nạp giấy và mức mực trớn nên phụ thuộc. Nếu bạn biết rằng mức sáp hết, điều đó có thể khiến bạn tin rằng lý do khiến chất lượng in kém là do mức sáp hết chứ không phải do dòng giấy bị tắc. Trong ví dụ này, trạng thái tổng thể của máy in là ảnh hưởng, mức mực in và trạng thái khay nạp giấy là những nguyên nhân có thể xảy ra và hai nguyên nhân trở nên phụ thuộc khi ảnh hưởng được biết đến. Đây là một ví dụ về sự phụ thuộc cảm ứng mà bạn sẽ tìm hiểu chi tiết hơn trong phần 5.2.3. Nếu bạn không có một biến số cho kết quả, điều này sẽ trở thành một mối quan hệ đối xứng giữa hai nguyên nhân. Nhưng ít khi loại bỏ ảnh hưởng chung của hai nguyên nhân ra khỏi mô hình.

#### THỂ HIỆN SỰ PHỤ THUỘC KHÔNG ĐỊNH HƯỚNG TRONG FIGARO

Bạn có thể thể hiện các mối quan hệ bất đối xứng trong Figaro theo hai cách: sử dụng các ràng buộc và sử dụng các điều kiện. Mỗi cái đều có ưu điểm và nhược điểm. Ưu điểm của phương pháp ràng buộc là nó đơn giản hơn về mặt khái niệm. Nhưng các số nằm trong các giới hạn được mã hóa cứng và không thể học được trong Figaro vì chúng không thể truy cập được bằng thuật toán học. Ưu điểm của phương pháp sử dụng điều kiện là có thể học được các con số.

Nguyên tắc cơ bản đằng sau hai cách tiếp cận là tương tự nhau. Phần 5.5 mô tả cách mã hóa chi tiết các phụ thuộc vô hướng, nhưng đây là phiên bản ngắn.

Khi tồn tại một sự phụ thuộc vô hướng giữa hai biến, một số giá trị chung của hai biến được ưa thích hơn những biến khác. Điều này có thể đạt được bằng cách gán trọng số cho các giá trị khớp khác nhau. Một ràng buộc mã hóa các trọng số bằng cách chỉ định một hàm từ một giá trị chung của hai biến thành một số thực biểu thị trọng số của biến đó giá trị. Trong cách tiếp cận điều kiện, về cơ bản cùng một thông tin được mã hóa nhưng trong một cách phức tạp hơn.

Ví dụ: hãy mã hóa mối quan hệ giữa hai pixel liền kề trong một hình ảnh. Những mối quan hệ này có bản chất “tất cả đều bình đẳng”. Ví dụ, bạn có thể tin rằng, nếu tất cả những thứ khác đều bằng nhau, thì hai pixel có khả năng có cùng màu với chúng để có các màu khác nhau. Trong thực tế, nhiều mối quan hệ có thể ảnh hưởng đến màu sắc của hai pixel, vì vậy khả năng chúng không thực sự cao gấp ba lần có cùng màu.

Đây là cách thể hiện mối quan hệ này trong Figaro bằng cách sử dụng phương pháp ràng buộc. Hãy gọi hai màu pixel là color1 và color2. Để giữ cho ví dụ đơn giản, bạn sẽ giả sử rằng màu sắc là Boolean. Hãy nhớ rằng một ràng buộc là một chức năng từ giá trị của một phần tử thành Double. Bạn cần tạo một phần tử đại diện cho cặp color1 và color2 để bạn có thể chỉ định một ràng buộc trên cặp. Bạn có thể làm được việc này như sau:

```
nhập com.cra.figaro.library.compound.^^  
cặp val = ^^(color1, color2)
```

^  
là hàm tạo  
cặp Figaro.

Bây giờ bạn phải xác định chức năng thực hiện ràng buộc:

```
def sameColorConstraint(cặp: (Boolean, Boolean)) =  
    if (cặp._1 == cặp._2) 0,3; khác 0,1
```

Thử nghiệm kiểm tra xem  
thành phần đầu tiên  
của cặp bằng  
thành phần thứ hai.

Cuối cùng, bạn áp dụng ràng buộc cho cặp màu:

```
cặp.setConstraint(sameColorConstraint _)
```

← Dấu gạch dưới chỉ ra rằng bạn muốn chính  
chức năng đó chứ không phải áp dụng nó.

Figaro diễn giải ràng buộc chính xác như bạn mong đợi. Tất cả những thứ khác đều bình đẳng, một trạng thái trong đó hai thành phần của cặp bằng nhau (nói cách khác, cả hai màu đều bằng nhau) sẽ có khả năng cao gấp ba lần so với trường hợp chúng không bằng nhau. Những ràng buộc này đến với nhau trong việc xác định phân phối xác suất theo cách chính xác, như được định nghĩa trong mục 5.5. Tuy nhiên, thật không may, các số 0,3 và 0,1 bị chôn vùi bên trong chức năng ràng buộc, vì vậy chúng không thể học được từ dữ liệu.

Đối với phương pháp điều kiện, tôi sẽ chỉ cho bạn cách nó hoạt động trong mã và sau đó giải thích tại sao nó đúng. Đầu tiên, bạn xác định một phần tử Boolean phụ trợ, mà bạn sẽ gọi cùng một ColorConstraintValue. Sau đó, bạn xác định nó để xác suất nó xuất hiện

true bằng với giá trị của ràng buộc. Cuối cùng, bạn sẽ thêm một điều kiện nói rằng phần tử phụ trợ này phải đúng. Điều này có thể đạt được bằng cách sử dụng một quan sát:

```
val sameColorConstraintValue =
    Chuỗi (màu1, màu2, (b1:
        Boolean, b2: Boolean) => if (b1 == b2)
            Lật (0,3); khác Lật (0,1))
    sameColorConstraintValue.observe(true)
```

Mã này tương đương với phiên bản ràng buộc. Để thấy điều này, hãy nhận ra rằng một điều kiện khiến bất kỳ giá trị nào vi phạm điều kiện có xác suất bằng 0; mặt khác, nó có xác suất 1. Xác suất chung của bất kỳ trạng thái nào có được bằng cách kết hợp các xác suất do định nghĩa các phần tử và từ các điều kiện hoặc ràng buộc.

Ví dụ, giả sử rằng hai màu bằng nhau. Sử dụng định nghĩa của Chuỗi, sameColorConstraintValue sẽ cho kết quả đúng với xác suất 0,3 và sai với xác suất 0,7. Nếu sameColorConstraintValue cho kết quả đúng, điều kiện sẽ có xác suất là 1, trong khi nếu sai, điều kiện sẽ có xác suất là 0. Do đó, xác suất kết hợp của điều kiện là  $(0,3 \times 1) + (0,7 \times 0) = 0,3$ . Tương tự, nếu hai màu không bằng nhau, xác suất kết hợp là  $(0,1 \times 1) + (0,9 \times 0) = 0,1$ .

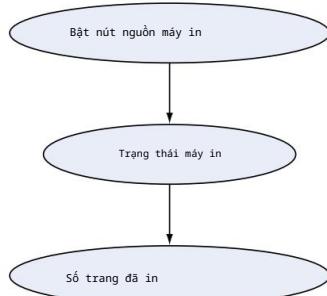
Vì vậy, bạn thấy rằng các màu giống nhau có khả năng cao gấp ba lần, tất cả các màu khác đều bằng nhau, vì các màu khác nhau. Đây chính xác là kết quả giống như kết quả bạn nhận được với ràng buộc.

Việc xây dựng này sử dụng các điều kiện là đủ tổng quát để bao gồm tất cả các mối quan hệ bất đối xứng. Ưu điểm của phương pháp này, như bạn có thể thấy, là các số 0,3 và 0,1 nằm bên trong các phần tử Flip, vì vậy bạn có thể làm cho chúng phụ thuộc vào các khía cạnh khác của mô hình. Ví dụ: giả sử bạn muốn tìm hiểu mức độ mà hai pixel liền kề có nhiều khả năng có cùng màu hơn là khác nhau. Bạn có thể tạo một phần tử, có thể được xác định bằng phân phối beta, để biểu thị trọng số. Sau đó, bạn có thể sử dụng phần tử này bên trong Flip thay vì 0,3. Sau đó, với dữ liệu đã cho, bạn có thể tìm hiểu sự phân phối theo giá trị của trọng số.

### 5.1.3 Sự phụ thuộc trực tiếp và gián tiếp

Tước khi bạn tiếp tục xem xét các mạng Bayesian và Markov, tôi muốn nhấn mạnh một điểm quan trọng. Một mô hình xác suất điển hình có nhiều cặp biến số, và do đó, kiến thức về một biến số sẽ thay đổi niềm tin của bạn về biến số kia. Theo định nghĩa, đây là tất cả các trường hợp phụ thuộc giữa các biến. Nhưng hầu hết các phụ thuộc này là gián tiếp: chúng không đi trực tiếp giữa hai biến mà thay vào đó đi qua một số biến trung gian. Nói một cách chính xác, lý do kiến thức về biến thứ nhất thay đổi niềm tin của bạn về biến thứ hai là vì kiến thức về biến thứ nhất thay đổi niềm tin của bạn về các biến trung gian, từ đó thay đổi niềm tin của bạn về biến thứ hai.

Ví dụ, hãy xem hình 5.2, trong đó có ba biến: Bật nút nguồn máy in, Trạng thái máy in và Số trang đã in. Rõ ràng, Bật nút nguồn máy in và Số trang đã in có mối quan hệ phụ thuộc. Nếu bạn biết rằng nút nguồn



Hình 5.2 Bật nút nguồn máy in có mối quan hệ gián tiếp với Số trang đã in thông qua biến trung gian Trạng thái máy in.

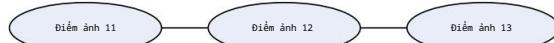
tắt, bạn sẽ tin rằng số trang đã in bằng không. Nhưng trong hình, sự phụ thuộc từ Bật nút nguồn máy in đến Số lượng trang đã in vượt qua thông qua một biến trung gian là Printer State. Điều này có nghĩa là lý do kiến thức về Bật nút nguồn máy in thay đổi niềm tin của bạn về Số Các trang được in trước tiên thay đổi niềm tin của bạn về Trạng thái máy in và thay đổi đến lượt niềm tin về Trạng thái máy in thay đổi niềm tin của bạn về Số lượng bản in Trang. Nếu bạn biết nút nguồn đang tắt, điều đó cho bạn biết rằng máy in đang bị hỏng, khiến bạn tin rằng sẽ không có trang nào được in.

**CẢNH BÁO THUẬT NGỮ** Trước đây tôi đã nói về các phụ thuộc có hướng và không có hướng, và bây giờ tôi đang nói về các phụ thuộc trực tiếp và gián tiếp.

Mặc dù tên giống nhau, nhưng chúng có ý nghĩa khác nhau. Một sự phụ thuộc trực tiếp đi trực tiếp giữa hai biến; từ trái nghĩa của nó là gián tiếp, đi thông qua các biến trung gian. Một phụ thuộc có hướng có hướng từ biến này sang biến khác, trái ngược với sự phụ thuộc vô hướng không có phương hướng. Bạn có thể có một phụ thuộc vô hướng trực tiếp và một phụ thuộc gián tiếp sự phụ thuộc có định hướng.

Hãy xem xét một ví dụ khác, lần này liên quan đến các phụ thuộc vô hướng. Trước đó, tôi đã đưa ra một ví dụ về các pixel liền kề trong một hình ảnh có sự phụ thuộc vô hướng giữa họ. Còn các pixel không liền kề thì sao? Xét ví dụ trong hình 5.3. Nếu như bạn biết rằng pixel 11 có màu đỏ, điều đó sẽ khiến bạn tin rằng pixel 12 có khả năng là đỏ, điều này sẽ khiến bạn tin rằng pixel 13 có thể có màu đỏ. Đây là ví dụ rõ ràng về sự phụ thuộc gián tiếp, bởi vì kiến thức về pixel 11 ảnh hưởng đến niềm tin về pixel 13 chỉ thông qua biến trung gian pixel 12.

Điều quan trọng là phải nhận ra những phụ thuộc nào trong miền của bạn là trực tiếp và đó là gián tiếp. Trong cả mạng Bayesian và Markov, bạn tạo một biểu đồ với



Hình 5.3 Pixel 11 có mối quan hệ gián tiếp với pixel 13 thông qua biến trung gian pixel 12.

các cạnh giữa các biến để biểu thị các phụ thuộc. Trong mạng Bayes, đây là các cạnh có hướng, trong khi trong mạng Markov, chúng không có hướng. Bạn chỉ vẽ một cạnh cho các phụ thuộc trực tiếp. Nếu hai biến chỉ có một phụ thuộc gián tiếp, bạn không vẽ một cạnh giữa chúng.

Tiếp theo, bạn sẽ xem xét các mạng Bayesian, là các mô hình đại diện cho các phụ thuộc có hướng, và sau đó là các mạng Markov, là các mô hình đại diện cho các phụ thuộc không được chỉ định. Tuy nhiên, tôi muốn chỉ ra rằng chỉ vì có các khung mô hình riêng biệt cho các phụ thuộc có hướng và không có hướng không có nghĩa là bạn phải chọn cái này hay cái kia cho mô hình của mình. Các khung khác kết hợp cả hai loại phụ thuộc trong một mô hình duy nhất. Thật dễ dàng để làm điều đó trong một chương trình xác suất: bạn sử dụng định nghĩa chung của các phần tử để mã hóa các phụ thuộc có hướng và thêm bất kỳ ràng buộc nào bạn muốn thể hiện các phụ thuộc không có hướng.

## 5.2 Sử dụng mạng Bayes

Bạn đã thấy rằng mã hóa các mối quan hệ giữa các biến là điều cần thiết để lập mô hình xác suất. Trong phần này, bạn sẽ tìm hiểu về các mạng Bayesian, là khuôn khổ tiêu chuẩn để mã hóa các mối quan hệ bắt đầu xứng bằng cách sử dụng các phụ thuộc có hướng. Bạn đã thấy các mạng Bayes trong chương 4, trong ngữ cảnh của ví dụ Rembrandt. Phần này cung cấp cách xử lý kỹ lưỡng hơn, bao gồm định nghĩa đầy đủ và giải thích về các mẫu lập luận mà bạn có thể sử dụng.

### 5.2.1 Định nghĩa mạng Bayesian

Mạng Bayesian là một đại diện của một mô hình xác suất bao gồm ba thành phần:

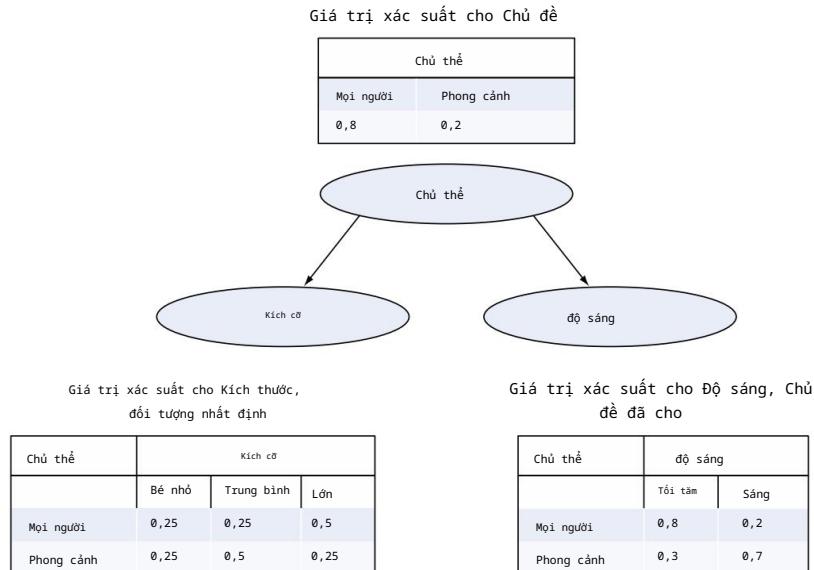
- Một tập hợp các biến với các miền tương ứng của chúng
- Đồ thị tuần hoàn có hướng trong đó mỗi biến là một nút
- Đối với mỗi biến, phân phối xác suất có điều kiện (CPD) trên biến có thể, với cha mẹ của nó

#### TẬP HỢP CÁC BIẾN VỚI MIỀN TƯƠNG ỨNG

Ví dụ trong hình 5.4 hiển thị ba biến: Đồi tượng, Kích thước và Độ sáng. Miền của một biến chỉ định những giá trị nào có thể có cho biến đó. Miền của Chủ thể là {People, Landscape}, miền Kích thước là {Small, Medium, Large} và miền Độ sáng là {Dark, Bright}.

#### BIỂU ĐỒ CHUYỂN ĐỔI CÓ HƯỚNG

Có hướng nghĩa là mỗi cạnh trong đồ thị có một hướng; nó đi từ biến này sang biến khác. Biến đầu tiên được gọi là cha mẹ và biến thứ hai được gọi là con. Trong hình 5.4 Chủ thể là cha của cả Kích thước và Độ sáng. Từ acyclic có nghĩa là không có chu trình nào trong biểu đồ: không có chu trình có hướng nào đi theo hướng mũi tên; bạn không thể bắt đầu tại một nút, đi theo các mũi tên và kết thúc tại cùng một nút. Nhưng bạn có thể có một chu trình vô hướng, nó sẽ là một chu trình nếu bạn bỏ qua

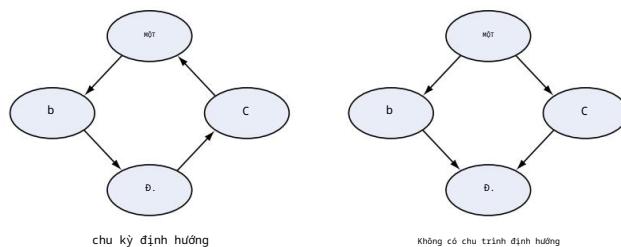


Hình 5.4 Mạng Bayesian ba nút

hướng của các cạnh. Khái niệm này được minh họa trong hình 5.5. Đồ thị bên trái có chu trình có hướng ABDCA. Trong biểu đồ bên phải, chu trình ABDCA đôi khi chạy ngược chiều mũi tên, vì vậy đó là chu trình vô hướng.

Do đó, biểu đồ bên trái không được phép nhưng biểu đồ bên phải được phép.

Điểm này rất quan trọng, bởi vì sau này, khi bạn cho phép các cạnh vô hướng thể hiện các phụ thuộc đối xứng, bạn sẽ được phép có các chu trình vô hướng.



Hình 5.5 Một chu trình có hướng là một chu trình đi theo các mũi tên và kết thúc tại nơi nó bắt đầu.

#### PHÂN PHỐI XÁC SUẤT CÓ ĐIỀU KIỆN TRÊN BIẾN

CPD chỉ định phân phối xác suất trên biến con, được cung cấp các giá trị của cha mẹ của nó. CPD này xem xét mọi khả năng gán giá trị cho cấp độ gốc, khi giá trị của cấp độ gốc có thể là bất kỳ giá trị nào trong miền của nó. Đối với mỗi nhiệm vụ, nó xác định một

phân phối xác suất cho đứa trẻ. Trong hình 5.6, mỗi biến có một CPD. Đối tượng là gốc của mạng, vì vậy nó không có cha mẹ. Khi một biến không có cha mẹ, CPD chỉ định một phân phối xác suất duy nhất cho biến đó. Trong ví dụ này, Đối tượng lấy giá trị Người có xác suất 0,8 và Phong cảnh có xác suất 0,2. Kích thước có cha là Chủ đề, vì vậy CPD của nó có một hàng cho mỗi giá trị của Chủ đề. CPD nói rằng khi Đối tượng có giá trị Con người, phân phối trên Kích thước làm cho Kích thước có giá trị Nhỏ với xác suất 0,25, Trung bình với xác suất 0,25 và Lớn với xác suất 0,5.

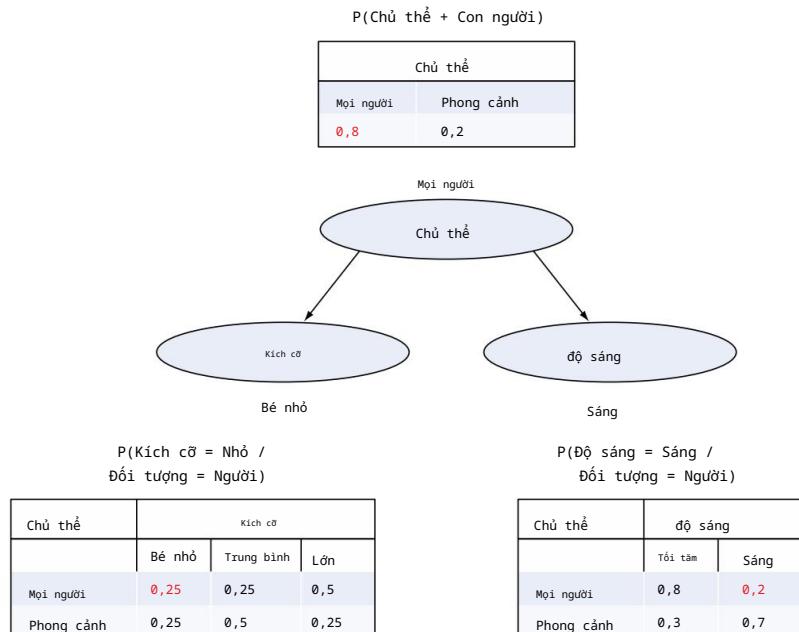
Khi Chủ đề có giá trị Cảnh, Kích thước có phân phối khác. Cuối cùng, Độ sáng cũng có Chủ đề gốc và CPD của nó cũng có một hàng cho mỗi giá trị của Chủ đề.

### 5.2.2 Cách mạng Bayes xác định phân phối xác suất

Đó là tất cả những gì định nghĩa về mạng Bayesian. Nay hãy xem mạng Bayes xác định phân bố xác suất như thế nào. Điều đầu tiên bạn cần làm là xác định các thế giới có thể. Đối với mạng Bayesian, một thế giới khả thi bao gồm việc gán giá trị cho từng biến, đảm bảo giá trị của từng biến nằm trong miền của nó.

Ví dụ: <Chủ đề = Con người, Kích thước = Nhỏ, Độ sáng = Sáng> là một thế giới khả thi.

Tiếp theo, bạn xác định xác suất của một thế giới có thể xảy ra. Cái này đơn giản. Tất cả những gì bạn phải làm là xác định mục nhập trong CPD của từng biến khớp với giá trị của cha mẹ và con cái trong thế giới có thể. Quá trình này được minh họa trong hình 5.6. Vì



$$P(\text{Đối tượng} = \text{Người}, \text{Kích thước} = \text{Nhỏ}, \text{Độ sáng} = \text{Sáng}) = 0,8 \times 0,25 \times 0,2 = 0,04$$

Hình 5.6 Tính toán xác suất của một thế giới có thể xảy ra bằng cách nhân các mục thích hợp trong mỗi CPD

ví dụ: đối với thế giới có thể có <Chủ đề = Mọi người, Kích thước = Nhỏ, Độ sáng = Sáng>, mục nhập cho Chủ đề là 0,8, là từ cột có nhãn Chủ đề. Đối với Size, bạn nhìn vào hàng tương ứng với Subject = People và cột tương ứng với Size = Small và lấy mục 0,25. Cuối cùng, đối với Độ sáng, bạn nhìn lại hàng tương ứng với Chủ đề = Mọi người và lần này lấy cột có nhãn Sáng để lấy mục 0,2. Cuối cùng, bạn nhân tất cả các mục này với nhau để có được xác suất của thế giới có thể xảy ra, là  $0,8 \times 0,25 \times 0,2 = 0,04$ .

Nếu bạn trải qua quá trình này cho mọi thế giới có thể, xác suất sẽ cộng lại thành 1, giống như chúng được cho là vậy. Đây luôn là trường hợp đối với mạng Bayesian. Vì vậy, bạn đã thấy cách mạng Bayes xác định phân phối xác suất hợp lệ. Bây giờ bạn đã hiểu chính xác mạng Bayes bao gồm những gì và ý nghĩa của nó, hãy xem cách sử dụng mạng Bayes để lấy niềm tin về một số biến, dựa trên kiến thức về các biến khác.

### 5.2.3 Lập luận với mạng Bayes

Một mạng Bayesian mã hóa rất nhiều tính độc lập giữa các biến.

Hãy nhớ lại rằng sự độc lập giữa hai biến có nghĩa là việc tìm hiểu điều gì đó về một biến không cho bạn biết bất cứ điều gì mới về biến kia. Từ ví dụ trước, bạn có thể thấy rằng Số trang đã in và Bật nút nguồn máy in không độc lập. Khi bạn biết rằng không có trang nào được in, điều đó làm giảm khả năng nút nguồn được bật.

Độc lập có điều kiện cũng tương tự. Hai biến là độc lập có điều kiện với biến thứ ba nếu sau khi biết biến thứ ba, việc tìm hiểu điều gì đó về biến thứ nhất không cho bạn biết điều gì mới về biến thứ hai. Một tiêu chí được gọi là phân tách d xác định khi nào hai biến trong mạng Bayes độc lập về mặt điều kiện với tập biến thứ ba. Tiêu chí có một chút liên quan, vì vậy tôi sẽ không cung cấp định nghĩa chính thức. Thay vào đó, tôi sẽ mô tả các nguyên tắc cơ bản và cho bạn xem một vài ví dụ.

Ý tưởng cơ bản là lập luận chảy dọc theo một con đường từ biến này sang biến khác. Trong ví dụ của hình 5.4, lý luận có thể chuyển từ Kích thước sang Độ sáng dọc theo đường dẫn Kích thước-Chủ đề-Độ sáng. Bạn đã thấy sơ qua về ý tưởng này trong phần 5.1.3 về sự phụ thuộc trực tiếp và gián tiếp. Trong một sự phụ thuộc gián tiếp, lý luận chảy từ biến này sang biến khác thông qua các biến trung gian khác. Trong ví dụ này, Chủ đề là biến trung gian giữa Kích thước và Độ sáng. Trong mạng Bayesian, lập luận có thể chạy dọc theo một đường dẫn miễn là đường dẫn đó không bị chặn tại một số biến.

Trong hầu hết các trường hợp, một đường dẫn bị chặn tại một biến nếu biến đó được quan sát. Vì vậy, nếu Chủ đề được quan sát, con đường Kích thước-Chủ đề-Độ sáng bị chặn. Điều này có nghĩa là nếu bạn quan sát Kích thước, nó sẽ không thay đổi niềm tin của bạn về Độ sáng nếu Chủ đề cũng được quan sát. Một cách khác để nói điều này là Kích thước độc lập có điều kiện với Độ sáng, đối tượng đã cho. Trong mô hình của chúng tôi, sự lựa chọn đối tượng của họa sĩ xác định kích thước và độ sáng, nhưng sau khi chọn đối tượng, kích thước và độ sáng được tạo ra một cách độc lập.

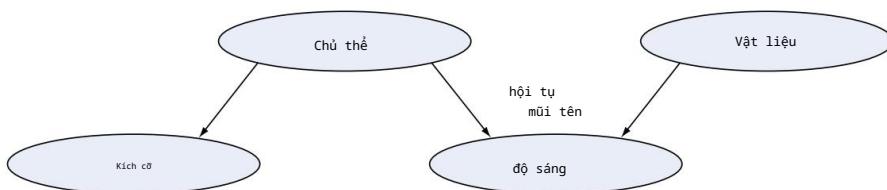
### CÁC MŨI TÊN HỘI TỤ VÀ CÁC PHỤ THUỘC GÂY RA Một

trường hợp khác thuật nghe có vẻ phản trực giác. Trong trường hợp này, một đường dẫn bị chặn tại một biến nếu biến đó không được quan sát và sẽ không bị chặn nếu biến đó được quan sát.

Tôi sẽ minh họa tình huống này bằng cách mở rộng ví dụ của chúng ta, như thể hiện trong hình 5.7. Bạn có một biến mới gọi là Chất liệu, có thể là dầu hoặc màu nước hoặc một số thứ khác.

Đương nhiên, Chất liệu là nguyên nhân của Độ sáng (có lẽ tranh sơn dầu sáng hơn màu nước), nên mạng lưới có hướng từ Chất liệu đến Độ sáng.

Ở đây, bạn có hai cha mẹ của cùng một đứa trẻ. Đây được gọi là **mẫu mũi tên hội tụ**, bởi vì các cạnh từ Đôi tượng và Vật liệu hội tụ tại Độ sáng.



Hình 5.7 Ví dụ về tranh mở rộng, bao gồm các mũi tên hội tụ giữa hai cha mẹ của cùng một đứa trẻ

Bây giờ, hãy suy nghĩ về lập luận giữa Chủ thẻ và Vật chất. Theo mô hình, Chủ đề và Tài liệu được tạo độc lập. Vì vậy, đây là sự thật:

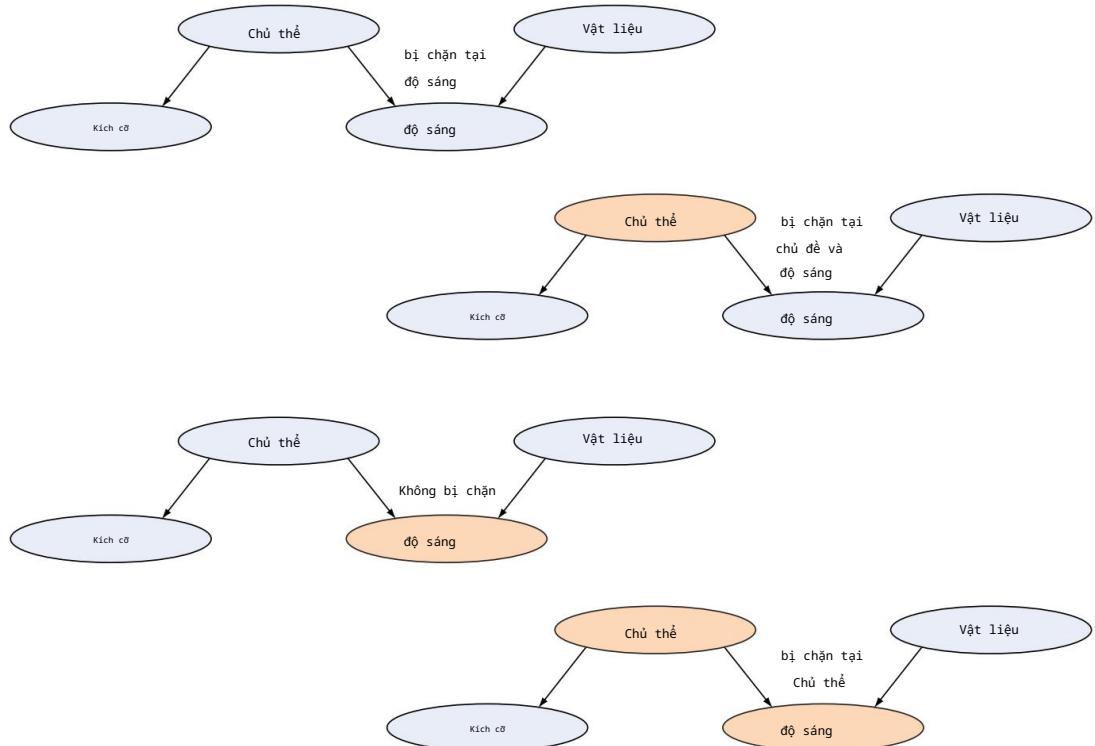
(1) **Đôi tượng và Vật liệu độc lập** khi không có gì được quan sát.

Nhưng điều gì xảy ra khi bạn quan sát thấy bức tranh sáng? Theo mô hình của chúng tôi, phong cảnh có xu hướng sáng hơn tranh người. Sau khi quan sát thấy bức tranh có màu sáng, bạn sẽ suy ra rằng bức tranh có nhiều khả năng là phong cảnh. Giả sử sau đó bạn quan sát bức tranh là tranh sơn dầu, bức tranh nào cũng có xu hướng sáng. Quan sát này cung cấp một lời giải thích khác về độ sáng của bức tranh của chúng tôi. Do đó, khả năng bức tranh là phong cảnh bị giảm đi phần nào so với xác suất sau khi bạn quan sát thấy bức tranh có màu sáng nhưng trước khi bạn quan sát thấy đó là bức tranh sơn dầu. Bạn có thể thấy rằng lý luận đang chuyển từ Vật chất sang Chủ thẻ dọc theo con đường Vật chất-Độ sáng-Chủ thẻ. Vì vậy, bạn nhận được tuyên bố sau:

(2) **Đôi tượng và Vật liệu không độc lập** về mặt điều kiện, được cung cấp Độ sáng.

Những gì bạn có ở đây là **mẫu đôi lập** với thông thường. Bạn có một đường dẫn bị chặn khi biến trung gian không được quan sát và trở nên không bị chặn khi biến được quan sát. Loại tình huống này được gọi là **sự phụ thuộc cảm ứng**- **sự phụ thuộc** giữa hai biến được tạo ra bằng cách quan sát biến thứ ba. Bất kỳ mẫu mũi tên hội tụ nào trong mạng Bayes đều có thể dẫn đến sự phụ thuộc cảm ứng.

Một đường dẫn giữa hai biến có thể bao gồm cả các mẫu thông thường và các mũi tên hội tụ. Suy luận chỉ chảy dọc theo đường dẫn nếu nó không bị chặn tại bất kỳ nút nào trên



Hình 5.8 Ví dụ về các đường dẫn bị chặn và không bị chặn kết hợp một mẫu bình thường với một mẫu mũi tên hội tụ. Mỗi hình hiển thị đường dẫn từ Kích thước đến Chất liệu, với Đối tượng và Độ sáng không được quan sát hoặc được quan sát.

con đường. Hình 5.8 hiển thị bốn ví dụ cho đường dẫn Kích thước-Chú thể-Độ sáng-Vật liệu. TRONG ví dụ trên cùng bên trái, cả Chủ thể và Độ sáng đều không được quan sát và đường dẫn là bị chặn ở Độ sáng vì nó có các mũi tên hội tụ. Trong ví dụ tiếp theo, trên phải, Đối tượng hiện được quan sát, vì vậy đường dẫn bị chặn ở cả Đối tượng và Độ sáng. Trong ví dụ tiếp theo, ở bên trái, Đối tượng không được quan sát và Độ sáng được quan sát, đó chính xác là điều kiện cần thiết để đường dẫn không bị chặn ở Đối tượng hoặc Độ sáng. Cuối cùng, ở dưới cùng bên phải, Đối tượng được quan sát ngoài Độ sáng, vì vậy con đường bị chặn ở đó.

### 5.3 Khám phá một ví dụ về mạng Bayes

Bây giờ bạn đã học các khái niệm cơ bản về mạng Bayesian, hãy xem xét một ví dụ về khắc phục sự cố máy in. Trước tiên tôi sẽ chỉ cho bạn cách thiết kế mạng và sau đó chỉ cho bạn tất cả các cách lập luận với mạng. Tôi sẽ tiết kiệm một thảo luận về việc tìm hiểu các tham số của mạng cho chương 12, nơi bạn sẽ khám phá một mẫu thiết kế hữu ích cho việc học tham số.

### 5.3.1 Thiết kế mô hình chẩn đoán hệ thống máy tính

Hãy tưởng tượng rằng bạn đang thiết kế một ứng dụng bàn trợ giúp để được hỗ trợ kỹ thuật. Bạn muốn để giúp nhân viên hỗ trợ kỹ thuật xác định nguyên nhân lỗi nhanh nhất có thể. Bạn có thể sử dụng hệ thống lý luận xác suất cho ứng dụng này—bằng chứng đã cho, bao gồm các báo cáo từ người dùng và các thử nghiệm chẩn đoán, bạn muốn xác định nguyên nhân bên trong trạng thái của hệ thống. Đôi với ứng dụng này, việc sử dụng mạng Bayesian để lập lại mô hình xác suất đã gửi là điều tự nhiên.

Khi bạn thiết kế một mạng Bayesian, bạn thường trải qua ba bước: chọn các biến và miền tương ứng của chúng, chỉ định cấu trúc mạng, và mã hóa các CPD. Bạn sẽ thấy cách thực hiện điều này trong Figaro.

Trong thực tế, bạn thường không đi qua tất cả các bước theo kiểu tuyến tính, chọn tất cả của các biến, xây dựng toàn bộ cấu trúc mạng, sau đó viết ra CPD. Thông thường, bạn sẽ xây dựng mạng từng chút một, tinh chỉnh nó khi bạn tiếp tục. bạn sẽ thực hiện cách tiếp cận đó ở đây. Đầu tiên, bạn sẽ xây dựng một mạng lưới cho mô hình lỗi in chung và sau đó đi sâu vào một mô hình chi tiết hơn của máy in.

#### MÔ HÌNH LỖI IN CHUNG: BIẾN

Bạn muốn lập mô hình các báo cáo có thể có từ người dùng, các lỗi có thể liên quan, và các yếu tố hệ thống có thể dẫn đến những lỗi đó. Bạn sẽ giới thiệu các biến cho tất cả những thứ này.

Bạn bắt đầu với Tóm tắt kết quả in. Điều này thể hiện trải nghiệm tổng thể của người dùng về kết quả in ở mức độ trừu tượng cao. Khi người dùng lần đầu gọi trợ giúp desk, người dùng đó chỉ có thể cung cấp một bản tóm tắt cao như thế này. Ba kết quả có thể xảy ra: (1) quá trình in diễn ra hoàn hảo (Bạn sẽ đánh giá điều này là xuất sắc); (2) in bút hap, nhưng không hoàn toàn đúng (kém); (3) hoàn toàn không in (none).

Tiếp theo, bạn xem xét các khía cạnh cụ thể khác nhau của kết quả in. Đây là Số của các trang đã in, có thể là số không, một số trang hoặc tất cả các trang; bản in Nhanh chóng, một biến Boolean cho biết việc in có diễn ra trong thời gian hợp lý hay không; và Chất lượng in tốt, một biến Boolean khác. Một lý do để lập mô hình cho mỗi những khía cạnh này riêng lẻ là chúng phân biệt giữa các lỗi khác nhau. Ví dụ: nếu không in được trang nào, có thể do mạng bị hỏng. Nhưng nếu một số trang không phải tất cả của các trang được in, vấn đề ít có khả năng là do mạng và có nhiều khả năng là do lỗi người dùng.

Vì vậy, bạn xem xét tất cả các yếu tố của hệ thống có thể ảnh hưởng đến kết quả in. Chúng bao gồm Trạng thái Máy in, có thể tốt, kém hoặc không hoạt động; trạng thái phần mềm, cái nào có thể chính xác, trực trắc hoặc bị lỗi; Trạng thái mạng, có thể tăng lên, không liên tục, hoặc xuống; và User Command Correct, là một biến Boolean.

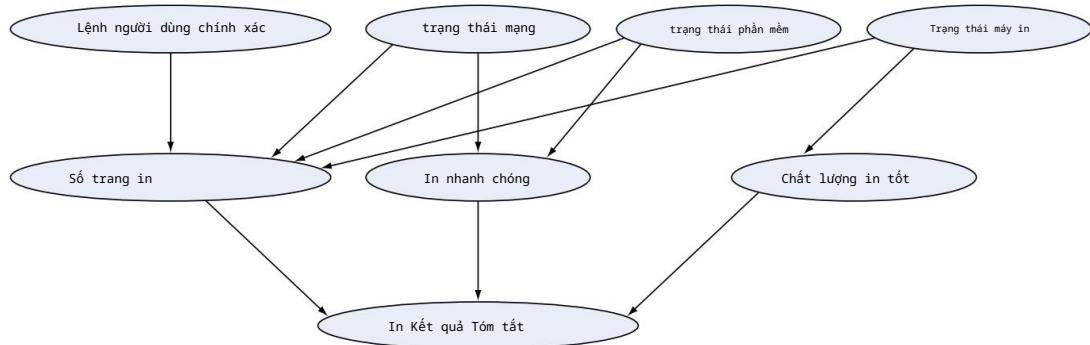
#### MÔ HÌNH LỖI IN CHUNG: CẤU TRÚC MẠNG

Xem xét các biến bạn đã xác định, có ba nhóm: In trừu tượng Tóm tắt kết quả, các khía cạnh cụ thể khác nhau của kết quả in và trạng thái hệ thống

ánh hưởng đến kết quả in. Theo đó, thật hợp lý khi thiết kế mạng theo Ba lớp. Thứ tự của các lớp nên là gì?

Mỗi quan hệ nhân quả tồn tại giữa các biến trạng thái hệ thống và bê tông in các biến kết quả. Ví dụ: Trạng thái Mạng ngừng hoạt động là nguyên nhân gây ra lỗi In Nhanh. Ngoài ra, các mối quan hệ cụ thể-trừu tượng tồn tại giữa các biến kết quả in riêng lẻ và Tóm tắt kết quả in tổng thể. Trong phần 5.1.1, Tôi đã nói rằng những mối quan hệ này có thể đi theo một trong hai hướng. Trong ứng dụng của chúng tôi, bạn đang lập mô hình trải nghiệm của người dùng và báo cáo kết quả in, do đó, theo quy tắc ngón tay cái mà tôi đã giới thiệu ở đó, hướng đi đúng là đi từ biến kết quả in cụ thể thành bản tóm tắt trừu tượng. Vì vậy, thứ tự của các lớp trong của chúng tôi mạng là (1) biến trạng thái hệ thống, (2) biến kết quả in cụ thể, (3) In Kết quả tổng kết.

Cấu trúc mạng được thể hiện trong hình 5.9. Bạn có thể thấy ba lớp, nhưng không phải lúc nào cũng có lợi thế từ mọi biến trong một lớp này sang các biến trong lớp tiếp theo lớp. Điều này là do một số biến kết quả in chỉ phụ thuộc vào trạng thái của một số thành phần của hệ thống. Ví dụ: chất lượng in có tốt không phụ thuộc vào trạng thái của máy in nhưng không phụ thuộc vào mạng. Tương tự, tốc độ của in theo mô hình của chúng tôi chỉ phụ thuộc vào mạng và phần mềm. Liệu những tuyên bố này có đúng hay không là điều cần tranh luận; điểm chính là trong bất kỳ ứng dụng, các đối số có thể được thực hiện để loại bỏ một số cạnh. Lợi ích của loại bỏ các cạnh là CPDs nhỏ hơn.



Hình 5.9 Cấu trúc mạng cho phần lỗi in chung của mô hình chẩn đoán hệ thống máy tính của chúng tôi

#### LỖI IN CHUNG MÔ HÌNH: CPDS

Tôi sẽ chỉ cho bạn thiết kế CPD thông qua mã Figaro, đảm bảo giải thích rõ nhất các mặt hàng thú vị. Phần 5.1.1 đã chỉ cho bạn nhiều cách khác nhau để định nghĩa CPD trong Figaro. Bạn sẽ sử dụng nhiều loại ở đây.

### Lịch kê 5.1 Triển khai mô hình lỗi in chung trong Figaro

Cái này đến từ chi tiết máy in người mẫu đang tới Ké tiếp.

```

    máy in valState =
    phần mềm valState =
        Chọn (0,8 -> 'đúng, 0,15 -> 'trục trặc, 0,05 -> 'bị lỗi)
    mạng valState =
        Chọn (0,7 -> 'lên, 0,2 -> 'không liên tục, 0,1 -> 'xuống)
val userCommandCorrect =
    Lật (0,65)

```

Đối với các biến gốc, chúng ta có các CPD nguyên tử như Select hoặc Flip.

```

val numPrintedPages =
    RichCPD(userCommandCorrect, networkState,
            softwareState, printerState, (*, *, *),
            OneOf('out)) -> Constant('zero), (*, *, OneOf('crashed), *) ->
            Constant('zero), (*, OneOf('xuống), *, *) -> Hàng số ('không),
            (OneOf(false), *, *, *) -> Chọn (0,3 -> 'không, 0,6 -> 'một số, 0,1 ->
            'tất cả ),
            (OneOf(true), *, *, *) ->
            Chọn (0,01 -> 'không, 0,01 -> 'một số, 0,98 -> 'tất cả))

```

numPrintedPages sử dụng RichCPD để thể hiện sự phụ thuộc vào lệnh của người dùng và trạng thái mạng, phần mềm và máy in.

```

val printQuickly =
    Chain(networkState, softwareState, (mạng: Biểu tượng,
    phần mềm: Biểu tượng) =>
        if (mạng == 'xuống || phần mềm == 'bị lỗi)
            Hàng số (sai) khác nếu
        (mạng == 'không liên tục || phần mềm == 'trục trặc)
            Lật (0,5) khác
        Lật (0,9))

```

printQuickly sử dụng Chuỗi để thể hiện sự phụ thuộc vào trạng thái mạng và phần mềm.

```

val goodPrintQuality =
    CPD(printerState, 'tốt ->
        Lật (0,95), 'kém -> Lật (0,3),
        'ra -> Hàng số (sai))

```

goodPrintQuality sử dụng một CPD đơn giản để thể hiện sự phụ thuộc vào trạng thái máy in.

```

val printResultSummary =
    Áp dụng(numPrintedPages, PrintQuickly, goodPrintQuality,
            (trang: Biểu tượng, nhanh chóng: Boolean, chất lượng: Boolean) => if
            (trang == 'không) 'không có gì
            khác nếu (trang == 'một số || !nhanh chóng || !chất lượng) 'kém khác'
            xuất sắc)

```

Bởi vì nó hoàn toàn được xác định bởi cha mẹ của nó, printResult Summary sử dụng Apply.

Mã này sử dụng một số kỹ thuật để thể hiện sự phụ thuộc xác suất của một đứa trẻ trên cha mẹ của nó:

- numPrintedPages sử dụng RichCPD, thực thi logic sau: Nếu trạng thái máy in không hoạt động hoặc trạng thái mạng bị hỏng hoặc phần mềm bị hỏng, không trạng thái nào sẽ được in, bất kể trạng thái của các phụ huynh khác. Nếu không thì, nếu người dùng đưa ra lệnh sai, không chắc tất cả các trạng thái

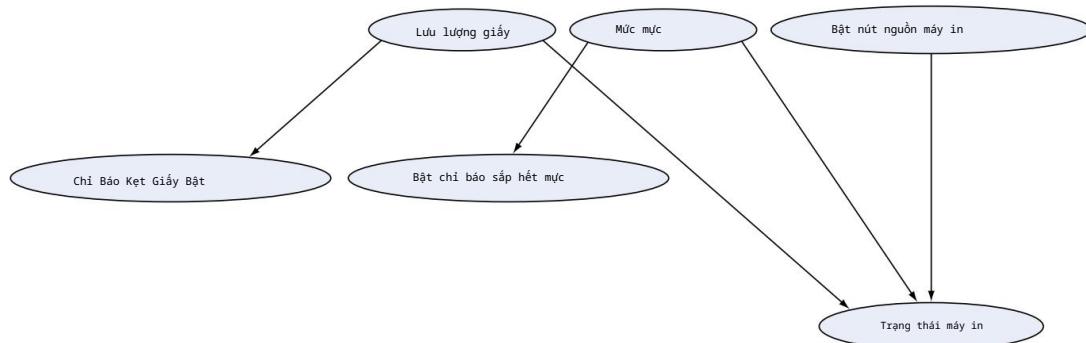
được in, nhưng nếu người dùng đưa ra đúng lệnh, rất có thể tất cả các trang sẽ được in.

- `printQuickly` sử dụng Chain để thực hiện logic sau: Nếu mạng bị hỏng hoặc phần mềm bị lỗi, nó chắc chắn sẽ không in nhanh. Mặt khác, nếu mạng bị gián đoạn hoặc phần mềm bị trực trặc, việc in nhanh sẽ trở nên khó khăn. Nếu cả mạng và phần mềm đều ở trạng thái tốt thì thường sẽ in nhanh (nhưng không đảm bảo).
- `goodPrintQuality` sử dụng CPD đơn giản. Nếu máy in bị tắt, chắc chắn chất lượng in sẽ không tốt. Nếu máy in ở tình trạng kém, có thể sẽ không có chất lượng in tốt. Ngay cả khi máy in ở trạng thái tốt, chất lượng in tốt cũng không được đảm bảo (vì đó là cách của máy in).
- `printSummary` là một biến xác định: Nó hoàn toàn được xác định bởi cha mẹ của nó mà không có bất kỳ sự không chắc chắn nào. Bạn có thể sử dụng Áp dụng thay vì Chuỗi cho biến tic xác định.

#### MÔ HÌNH MÁY IN CHI TIẾT

Phần này đi qua mô hình máy in chi tiết nhanh hơn, bởi vì nhiều nguyên tắc giống nhau. Cấu trúc mạng được thể hiện trong hình 5.10. Ba yếu tố ảnh hưởng đến trạng thái của máy in: Dòng giấy, Mức mực và Bật nút nguồn máy in. Mô hình thêm một loại biến mới mà bạn chưa từng thấy, một chỉ số hoặc phép đo. Chỉ báo két giấy Bật là phép đo Dòng chảy của giấy và Bật chỉ báo sắp hết mực là phép đo Mức mực. Như đã thảo luận trong phần 5.1.1, mối quan hệ giữa giá trị thực và phép đo của nó là một loại mối quan hệ nhân quả, vì vậy bạn có lợi thế từ Dòng chảy giấy đến Bật chỉ báo két giấy và từ Mức mực đến Bật chỉ báo sắp hết mực.

Đây là mã xác định CPD, phần lớn là đơn giản.



Hình 5.10 Cấu trúc mạng cho mô hình máy in chi tiết

## Liệt kê 5.2 Mô hình máy in chi tiết trong Figaro

```

máy in valPowerButtonOn = Flip(0,95)
val tonerLevel = Select(0,7 -> 'cao, 0,2 -> 'thấp, 0,1 -> 'ra) val tonerLowIndicatorOn = _____
    ↑
    Nhớ lại rằng một dấu nháy đơn
    ký tự cho biết
    Loại biểu tượng Scala.

    Nếu (máy inPowerButtonOn,
        CPD(PaperFlow,
            'cao -> Lật (0,2),
            'thấp -> Lật (0,6),
            'ra -> Lật (0,99)),
        Hàng số (sai))
    val paperFlow = Select(0,6 -> 'tròn trịa, 0,2 -> 'không đều, 0,2 -> _bi_kết)

    giấy valJamIndicatorOn =
        Nếu (máy inPowerButtonOn,
            CPD(mức mực in,
                'cao -> Lật (0,1), 'thấp ->
                Lật (0,3), 'ra -> Lật (0,99)),
            _____
        Hàng số (sai))

    máy in valState =
        Áp dụng(printerPowerButtonOn, tonerLevel, paperFlow,
            (sức mạnh: Boolean, mục: Biểu tượng, giấy: Biểu tượng) => {
                nếu (sức mạnh) {
                    if (mục == 'cao && giấy == 'mịn') 'tốt other if (mục == 'out || paper == 'out)
                        'out other' kém
                    } khác 'ra
                }
            })
    ↑
    Một CPD được lồng bên trong một If.
    Nếu nút nguồn máy in đang
    bật, chỉ báo mức sắp hết phụ thuộc
    vào mức mực. Nếu nút nguồn tắt, chỉ
    báo mức sẽ tắt vì không có điện,
    bất kể mức mực.

    ↑
    Trạng thái máy
    in là một bản
    tóm tắt xác
    định của các yếu tố
    dựng lên.

```

Để tóm tắt ví dụ, cấu trúc mạng Bayes đầy đủ được thể hiện trong hình 5.11.

Bạn cũng có thể xem toàn bộ chương trình trong mã của cuốn sách dưới chap05/PrinterProbability.scala.

## 5.3.2 Lập luận với mô hình chẩn đoán hệ thống máy tính

Phần này cho thấy cách lập luận với mô hình chẩn đoán hệ thống máy tính mà bạn vừa được xây dựng. Cơ chế lập luận của Figaro rất đơn giản, nhưng các mẫu lập luận mà bạn thoát khỏi nó thật thú vị và minh họa cho các khái niệm được giới thiệu trong phần 5.2.3. Tất cả của các mẫu lý luận trong phần này được chia thành các bước riêng biệt trong mã cho chương. Nhận xét các bước bạn không muốn thực hiện, để đánh dấu từng bước như nó được trình bày.

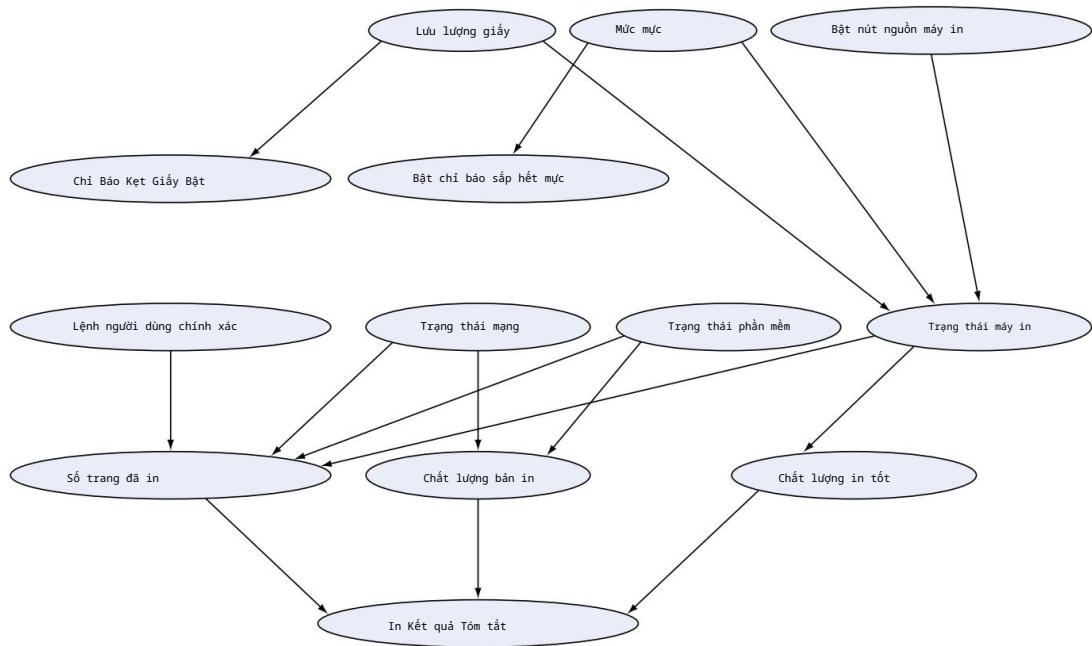
## TRẮC NGHIỆM XÁC SUẤT TRƯỚC

Đầu tiên, hãy truy vấn khả năng nút nguồn của máy in đang bật mà không có bất kỳ bằng chứng nào. Đây là xác suất trước. Bạn có thể sử dụng đoạn mã sau:

```

val answerWithNoEvidence =
    VariableElimination.probability(printerPowerButtonOn, true)
    println("Xác suất trước nút nguồn máy in đang bật =
        trả lờiKhông có bằng chứng)
    ↑
    tính toán
    P( Máy in
    Nút nguồn
    Bật = đúng)

```



Hình 5.11 Mạng đầy đủ cho ví dụ chẩn đoán hệ thống máy tính

Điều này in kết quả sau:

Xác suất trước nút nguồn máy in được bật = 0,95

Nếu bạn nhìn lại mô hình, bạn sẽ thấy máy inPowerButtonOn được xác định bằng cách sử dụng dòng sau:

máy in valPowerButtonOn = Flip(0,95)

Bạn có thể thấy rằng câu trả lời cho truy vấn chính xác là những gì bạn nhận được nếu bạn bỏ qua toàn bộ mô hình ngoại trừ định nghĩa này. Đây là một ví dụ về quy tắc chung: mạng lưới công việc xuôi dòng từ một biến chỉ liên quan đến biến đó nếu nó có bằng chứng. TRONG đặc biệt, đối với xác suất trước đó, không có bằng chứng, vì vậy bạn không quan tâm đến mạng hạ lưu.

#### **ĐẶT VẤN ĐỀ VỚI BẰNG CHỨNG**

Điều gì xảy ra nếu bạn giới thiệu bằng chứng? Hãy truy vấn mô hình cho xác suất rằng nút nguồn máy in đang bật, do kết quả in kém, có nghĩa là có là một số kết quả, nhưng đó không phải là điều người dùng muốn. Bạn có thể sử dụng đoạn mã sau:

```

printResultSummary.observe('poor) val
answerIfPrintResultPoor =
  VariableElimination.probability(printerPowerButtonOn, true) println("Khả năng nút nguồn máy in
được bật kém + answerIfPrintResultPoor)
+ "quả =
  
```

Tính  $P(\text{Máy in}\ |\ \text{Bật nút nguồn} = \text{đúng} \mid \text{In Kết quả} = \text{kém})$

Điều này in kết quả sau:

Xác suất bật nút nguồn máy in cho kết quả kém = 1,0

Điều này có thể gây ngạc nhiên! Xác suất sẽ cao hơn nếu bạn không có bất kỳ bằng chứng nào về việc in án. Nếu bạn nghĩ về mô hình, bạn có thể hiểu tại sao. Một người nghèo kết quả in chỉ có thể xảy ra nếu ít nhất một số trang được in, điều này sẽ không là trường hợp nếu tắt nguồn. Do đó, xác suất bật nguồn là 1.

Bây giờ hãy truy vấn bằng chứng rằng không có gì được in ra:

```
printResultSummary.observe('none) val
answerIfPrintResultNone =
    VariableElimination.probability(printerPowerButtonOn, true) println("Khả năng nút nguồn
máy in được bật trống + answerIfPrintResultNone)
+ "quá =
" + "quá =
```

Tính P( Máy in  
Bật nút nguồn  
= đúng | máy in  
Kết quả = không có )

Bây giờ kết quả như sau:

Xác suất nút nguồn máy in được bật cho kết quả trống = 0,8573402523786461

Đây là như bạn mong đợi. Nút nguồn bị tắt là một lời giải thích hợp lý cho việc trống in kết quả, vì vậy xác suất của nó tăng lên dựa trên bằng chứng.

#### ĐỘC LẬP VÀ CHẶN

Mục 5.2.3 đã giới thiệu khái niệm đường đi bị chặn và mối quan hệ của đường dẫn này khái niệm độc lập có điều kiện. Khái niệm này có thể được minh họa bằng ba biến số: Tóm tắt kết quả in, Bật nút nguồn máy in và Trạng thái máy in. Trong hình 5.11, bạn thấy rằng đường dẫn từ Bật nút nguồn máy in đến Tóm tắt kết quả in đi thông qua Trạng thái Máy in. Vì đây không phải là mẫu mũi tên hội tụ nên đường đi là bị chặn nếu Trạng thái máy in được quan sát. Thật vậy, đây là trường hợp, như bạn sẽ thấy bây giờ:

```
printResultSummary.unobserve()
printerState.observe('out) val
answerIfPrinterStateOut =
    VariableElimination.probability(printerPowerButtonOn, true)
println("Khả năng nút nguồn máy in được bật " + answerIfPrinterStateOut)
+ "ra trạng thái máy in =
" + "ra trạng thái máy in

printResultSummary.observe('none) val
answerIfPrinterStateOutAndResultNone =
    VariableElimination.probability(printerPowerButtonOn, true)
println("Khả năng nút nguồn máy in được bật "
+ "ra trạng thái máy in và kết quả trống =
" + answerIfPrinterStateOutAndResultNone)
```

Tính P( Máy in  
Bật nút nguồn =
true | máy in  
Nhà nước = ra)

Tính P( Máy in  
Bật nút nguồn =
thật | Trạng thái máy  
in = out, kết quả in  
Tóm tắt = không có )

Điều này in như sau:

Khả năng nút nguồn máy in được bật ở trạng thái máy in đã cho = 0,6551724137931032

Khả năng nút nguồn máy in được bật ở trạng thái máy in đã cho và kết quả trống = 0,6551724137931033

Bạn có thể thấy rằng việc biết rằng kết quả in trống không thay đổi xác suất rằng nút nguồn của máy in đang bật, sau khi bạn đã biết rằng trạng thái của máy in là ngoài. Điều này là do Tóm tắt kết quả in độc lập về mặt điều kiện với Máy in Bật nút nguồn, Trạng thái máy in nhất định.

#### LÝ DO GIỮA CÁC TÁC DỤNG KHÁC NHAU CỦA CÙNG MỘT NGUYÊN NHÂN

Tất cả các con đường lý luận mà bạn đã thấy cho đến nay đều đi thẳng lên mạng. Bạn cũng có thể kết hợp cả hai hướng khi lập luận. Để dàng nhận thấy điều này khi bạn cân nhắc xem phép đo cho bạn biết điều gì về giá trị mà nó đang đo và điều đó có thể lần lượt thông báo. Trong ví dụ của chúng tôi, Toner Low Indicator On là phần tử con của Toner Level, và Mức mực in là cấp độ gốc của Trạng thái máy in. Nếu mức mực thấp, ít có khả năng tình trạng máy in tốt. Trong khi đó, đèn báo sắp hết mực đang bật là dấu hiệu cho thấy mức mực thấp. Lý do là nếu bạn quan sát thấy đèn báo sắp hết mực đang bật, thì khả năng máy in hoạt động tốt sẽ giảm đi. Bạn có thể thấy rằng đây là trường hợp với đoạn mã sau:

```
printResultSummary.unobserve()
printerState.unobserve() val
printerStateGoodPrior =
    VariableElimination.probability(printerState, 'good') println("Xác suất trước
trạng thái máy in tốt = " + printerStateGoodPrior)
```

tính toán  
 $P(\text{Máy in})$   
 $\text{Nhà nước} = \text{tốt}$

```
tonerLowIndicatorOn.observe(true) val
printerStateGoodGivenTonerLowIndicatorOn =
    VariableElimination.probability(printerState, 'good') println("Xác suất trạng
thái máy in tốt khi mực thấp + chỉ báo = "
    " + máy inStateGoodGivenTonerLowIndicatorOn)
```

tính toán  
 $P(\text{Trạng thái máy in} = \text{tốt} | \text{Mực in} \\ \text{Chi số thấp})$   
 $\text{Bật} = \text{đúng}$

Điều này in như sau:

```
Xác suất trước trạng thái máy in tốt = 0,3989999999999997
Xác suất trạng thái máy in tốt với chỉ báo mực thấp = 0,23398328690807796
```

Bạn có thể thấy rằng khả năng máy in ở trạng thái tốt sẽ giảm khi bạn quan sát chỉ báo mực thấp, như mong đợi.

#### LÝ DO GIỮA NHỮNG NGUYÊN NHÂN KHÁC NHAU CỦA CÙNG MỘT TÁC DỤNG: SỰ PHỤ THUỘC GÂY RA

Như đã thảo luận trong phần 5.2.3, lập luận giữa các nguyên nhân khác nhau của cùng một hệ quả là khác với các loại lý luận khác, bởi vì nó liên quan đến các mũi tên hội tụ, mà dẫn đến một sự phụ thuộc gây ra. Ví dụ: hãy sử dụng Trạng thái phần mềm và Mạng State, cả hai đều là cha mẹ của Prints Quick. Đầu tiên, bạn sẽ nhận được xác suất trước rằng trạng thái phần mềm là chính xác:

```
tonerLowIndicatorOn.unobserve() val
softwareStateCorrectPrior =
    VariableElimination.probability(softwareState, 'true') println("Xác suất trước trạng
thái phần mềm là đúng = softwareStateCorrectPrior")
```

tính toán  
 $P(\text{Phần mềm})$   
 $\text{Trạng thái} = \text{đúng}$

Điều này in như sau:

Xác suất trước trạng thái phần mềm là đúng = 0,8

Tiếp theo, bạn sẽ quan sát thấy rằng mạng đã hoạt động và truy vấn lại trạng thái phần mềm:

```
networkState.observe('up') val
softwareStateCorrectGivenNetworkUp =
    VariableElimination.probability(softwareState, 'true') println("Xác suất trạng thái
phần mềm là chính xác khi mạng lên = softwareStateCorrectGivenNetworkUp")
```

tính toán	P(Phần mềm
Nhà nước	= chính xác
	Mạng
	Nhà nước = lên)

Điều này in như sau:

Xác suất trạng thái phần mềm là chính xác với mạng up = 0,8

Xác suất không thay đổi, mặc dù có một đường dẫn rõ ràng từ Network State đến Trạng thái phần mềm thông qua Bản in nhanh chóng! Điều này cho thấy rằng nhìn chung, hai nguyên nhân của cùng một hiệu lực là độc lập. Điều này đúng về mặt trực giác: mạng đang hoạt động không liên quan đến việc trạng thái phần mềm có đúng hay không.

Bây giờ, nếu bạn biết rằng máy in không in nhanh, điều đó khác. Nếu bạn thấy máy in in chậm, mô hình của chúng tôi đưa ra hai cách giải thích khả thi: mạng vẫn đề hoặc một vấn đề phần mềm. Nếu bạn quan sát thấy mạng vẫn hoạt động, đó phải là sự cố phần mềm. Bạn có thể thấy điều này với đoạn mã sau:

tính toán	networkState.unobserve()
P(Trạng thái phần mềm	printQuickly.observe(false) val
= đúng	softwareStateCorrectGivenPrintsSlowly =
In nhanh	VariableElimination.probability(softwareState, 'true') println("Xác suất trạng thái
= sai)	phần mềm là chính xác cho bản in"
	+ "từ từ = " + phần mềmStateCorrectGivenPrintsSlowly)

tính toán	networkState.observe('up') val
P(Trạng thái phần mềm	softwareStateCorrectGivenPrintsSlowlyAndNetworkUp =
= đúng	VariableElimination.probability(softwareState, 'true') println("Xác suất trạng thái
In nhanh =	phần mềm là chính xác khi in " + "từ từ và kết nối mạng = "
sai, Mạng	softwareStateCorrectGivenPrintsSlowlyAndNetworkUp)
Nhà nước = lên)	

Chạy mã này in như sau:

Trạng thái phần mềm xác suất là chính xác khi in chậm =  
0,6197991391678623

Xác suất trạng thái phần mềm là chính xác khi in chậm và kết nối mạng =  
0,39024390243902435

Biết rằng mạng đang hoạt động làm giảm đáng kể khả năng phần mềm bị lỗi.

Chính xác. Vì vậy, Trạng thái phần mềm và Trạng thái mạng là độc lập, nhưng chúng không có điều kiện độc lập với nhau khi cung cấp In nhanh. Đây là một ví dụ về sự phụ thuộc gây ra.

Để tóm tắt:

- Khi lập luận từ tác động X đến nguyên nhân gián tiếp Y, X không độc lập với Y, nhưng trở nên độc lập có điều kiện, với Z, nếu Z chặn đường đi từ X đến Y
- Điều tương tự cũng xảy ra khi lập luận từ nguyên nhân đến kết quả gián tiếp hoặc giữa hai hậu quả của cùng một nguyên nhân.
- Đối với hai nguyên nhân X và Y cùng tác dụng Z thì ngược lại. X và Y là độc lập, nhưng không độc lập có điều kiện, cho trước Z, là kết quả của sự phụ thuộc cảm ứng.

Điều đó kết thúc ví dụ chẩn đoán hệ thống máy tính. Bạn đã thấy một mạng khá quan trọng với một số mẫu lý luận thú vị. Trong phần tiếp theo, bạn sẽ vượt ra ngoài các mạng Bayesian truyền thống.

#### **5.4 Sử dụng lập trình xác suất để mở rộng mạng Bayesian: dự đoán thành công của sản phẩm**

Phần này chỉ ra cách mở rộng mô hình mạng Bayes

cơ bản để dự đoán thành công của chiến dịch tiếp thị và vị trí sản phẩm. Mục đích của ví dụ này gồm hai phần: thứ nhất, để thể hiện sức mạnh của việc sử dụng các ngôn ngữ lập trình để mở rộng mạng Bayes, và thứ hai, để minh họa cách mạng Bayes có thể được sử dụng không chỉ để suy ra nguyên nhân của các sự kiện được quan sát mà còn để dự đoán các sự kiện trong tương lai. Như với ví dụ chẩn đoán hệ thống máy tính, trước tiên tôi sẽ trình bày cách thiết kế mô hình và thể hiện nó trong Figaro, sau đó là cách lập luận với mô hình.

##### **5.4.1 Thiết kế mô hình dự đoán thành công của sản phẩm**

Hãy tưởng tượng rằng bạn có một sản phẩm mới và bạn muốn làm cho sản phẩm đó thành công nhất có thể. Bạn có thể đạt được điều này theo nhiều cách khác nhau. Bạn có thể đầu tư vào bao bì của sản phẩm và những thứ khác có thể khiến nó hấp dẫn khách hàng. Bạn có thể cố gắng định giá nó theo cách mà mọi người sẽ có nhiều khả năng mua nó hơn. Hoặc bạn có thể quảng cáo sản phẩm bằng cách tặng các phiên bản miễn phí với hy vọng mọi người sẽ chia sẻ chúng với bạn bè của họ. Trước khi bạn chọn một chiến lược, bạn muốn biết tầm quan trọng tương đối của từng yếu tố.

Phần này mô tả khuôn khổ của một mô hình mà bạn có thể sử dụng cho ứng dụng này. Tôi nói khuôn khổ, bởi vì đây chỉ là khung của mô hình mà bạn sẽ xây dựng nếu bạn làm điều này thực sự, nhưng nó đủ để tạo nên điểm nhấn. Mô hình được trình bày ở đây là một mạng Bayesian đơn giản chỉ có bốn nút, nhưng các loại nút và CPD rất phong phú và thú vị.

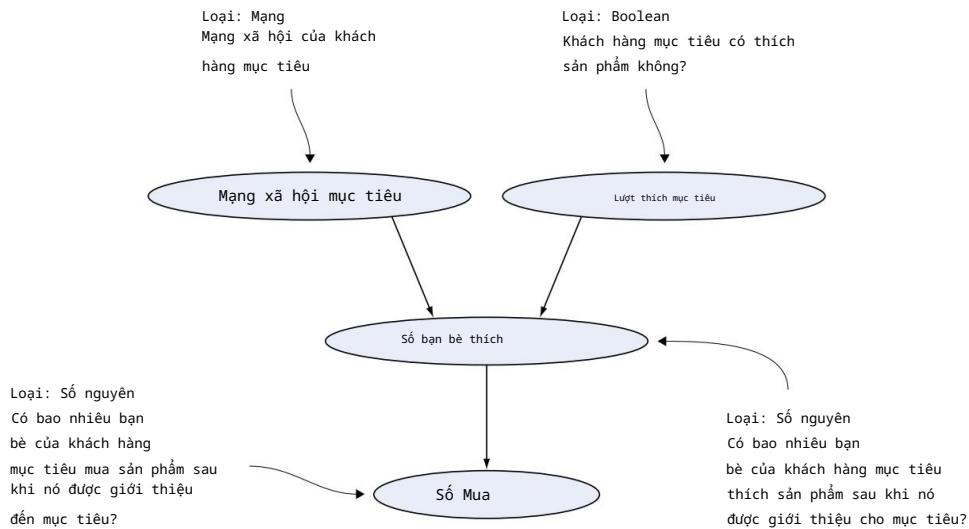
Mô hình có bốn biến, như thể hiện trong hình 5.12:

- Mạng xã hội mục tiêu là một biến có loại là mạng xã hội. Đây là một cách mà ngôn ngữ lập trình cho phép bạn vượt xa các mạng Bayesian thông thường, nơi các biến chỉ là Boolean, kiểu liệt kê, số nguyên hoặc số thực.

CPD của biến này tạo mạng ngẫu nhiên dựa trên mức độ phổ biến của mục tiêu. Đối với bản thân mức độ phổ biến, bởi vì nó là một biến kiểm soát, bạn mô hình hóa nó như một hằng số đã biết chứ không phải là một biến. Nhưng nếu muốn, bạn có thể đưa ra sự không chắc chắn về mức độ phổ biến và biến nó thành một biến số.

- Lượt thích mục tiêu là một biến Boolean cho biết liệu mục tiêu có thích sản phẩm hay không, đây là một hàm của chất lượng sản phẩm. Một lần nữa, chất lượng sản phẩm là một hằng số đã biết, nhưng bạn có thể biến nó thành một biến phụ thuộc vào về đầu tư.
- Số Bạn bè Thích là một biến số nguyên, nhưng CPD của nó đi qua Mạng Xã hội Mục tiêu để xác định số lượng bạn bè được xem sản phẩm và những người thích sản phẩm đó. Đây là một CPD phong phú hơn nhiều so với truyền thống có sẵn trong các mạng Bayesian.
- Số Mua là một biến số nguyên. Mô hình của nó được xác định đơn giản bằng cách xét rằng mỗi người thích sản phẩm sẽ mua nó với xác suất phụ thuộc vào khả năng chi trả của nó, một biến kiểm soát có giá trị là một hằng số đã biết.  
Do đó, số người mua sản phẩm là một nhị thức.

Đây là mã Figaro cho mô hình này. Tôi sẽ mô tả cách mã hoạt động ở mức cao và sau đó trình bày chi tiết về mô hình.



Hình 5.12 Mạng dự đoán thành công của sản phẩm

### Lịch kê 5.3 Mô hình dự đoán thành công của sản phẩm trong Figaro

```

lớp Mạng (mức độ phỏ biến: Gấp đôi) {
    val numNodes = Poisson(mức độ phỏ biến)
}

Tạo một lớp người mẫu
mắt cái đã biết điều khiển
thông số như đối số

mô hình lớp (mục tiêu Phổ biến: Gấp đôi, chất lượng sản phẩm: Gấp đôi,
khả năng chi trả: Gấp đôi) {

def generateLikes(numFriends: Int,
                  productQuality: Double): Element[Int] = {

    người trợ giúp def(friendsVisited: Int, totalLikes: Int,
                      lượt thích chưa được xử lý: Int): Element[Int] = {
        if (Số lượt thích chưa được xử lý == 0) Hằng số (tổng số lượt thích) khác {

            val unvisitedFraction =
                1.0 - (friendsVisited.toDouble - 1) / (numFriends - 1)
            val newVisited = Binomial(2, UnvisitedFraction) val newLikes =


                Nhị thức (mới được truy cập, Hằng số (chất lượng sản phẩm))
                Chuỗi (mới được truy cập, mới được thích,
                      (đã truy cập: Int, thích: Int) =>
                      helper(bạn bè đã ghé thăm + chưa ghé thăm, tổng số Lượt
                            thích + lượt thích, lượt thích
                            chưa xử lý + lượt thích - 1))

        }
    }

    người trợ giúp (1, 1, 1)
}

val targetSocialNetwork = new Network(mục tiêu Phổ biến)

Liệu mục tiêu thích sản phẩm là một Boolean yêu tố dựa trên sản phẩm chất lượng.

val targetLikes = Flip(productQuality)

val numberFriendsLike =
    Chuỗi(lượt thích mục tiêu, targetSocialNetwork.numNodes, (l: Boolean, n: Int) => if (l) generateLikes(n,
                                              productQuality) other Constant(0))

số valMua =
    Nhị thức (số Bạn bè Thích, Hằng số (khả năng chi trả))
}

```

Xác định một lớp Mạng với một thuộc tính duy nhất được xác định bởi một Phần tử Poisson (xem văn bản)

Định nghĩa đệ quy quy trình tạo số lượng người thích sản phẩm (xem văn bản)

Nếu mục tiêu thích sản phẩm, hãy tính số lượng bạn bè bằng cách tạoLikes. Nếu không, có thể sẽ không nói với bạn bè về điều đó, vì vậy con số là 0.

Số lượng bạn bè mua sản phẩm là một nhị thức (xem văn bản).

Ba chi tiết của mã cần giải thích thêm: phần tử Poisson được sử dụng trong lớp Mạng, quy trình tạoLikes và định nghĩa của numberBuy. Hãy đầu tiên nói về phần tử Poisson và logic numberBuy và sau đó đến phần generateLikes, đây là phần thú vị nhất của mô hình.

- Một phần tử Poisson là một phần tử số nguyên sử dụng cái được gọi là phân bố Poisson. Phân phối Poisson thường được sử dụng để mô hình hóa số lượng sự xuất hiện của một sự kiện trong một khoảng thời gian, chẳng hạn như số lượng mạng

thất bại trong một tháng hoặc số quả phạt góc trong một trận bóng đá. Với một chút sáng tạo, phân phối Poisson có thể được sử dụng để mô hình hóa mọi tình huống nơi bạn muốn biết số thứ trong một vùng. Ở đây, bạn sử dụng nó để mô hình hóa số lượng người trong mạng xã hội của ai đó, điều này khác từ cách sử dụng thông thường nhưng vẫn là một lựa chọn hợp lý.

Phần tử Poisson lấy đối số là số lần xuất hiện trung bình mà bạn mong đợi trong khoảng thời gian đó, nhưng cho phép số đó là nhiều hơn hoặc ít hơn mức trung bình. Trong mô hình này, đối số là sự phổ biến của mục tiêu; mức độ phổ biến phải là ước tính về số người trung bình mà bạn mong muốn có mặt trong mạng xã hội của mục tiêu.

- Đây là logic của số người mua sản phẩm. Mỗi người người thích sản phẩm sẽ mua nó với xác suất bằng với giá trị của tham số khả năng chi trả. Vì vậy, tổng số người mua được đưa ra bởi một nhị thức, trong đó số lần thử là số bạn bè thích sản phẩm, và xác suất mua phụ thuộc vào khả năng chi trả của sản phẩm. Vì bản thân số người thích sản phẩm đã là một yếu tố, bạn cần sử dụng nhị thức hợp chất lấy các phần tử làm đối số của nó. Yếu tố nhị thức hợp chất yêu cầu rằng xác suất thành công của một dùng thử cũng là một yếu tố, đó là lý do tại sao khả năng chi trả được gói gọn trong một Hằng số. Một phần tử Constant nhận một giá trị Scala thông thường và tạo ra phần tử Figaro luôn có giá trị đó.

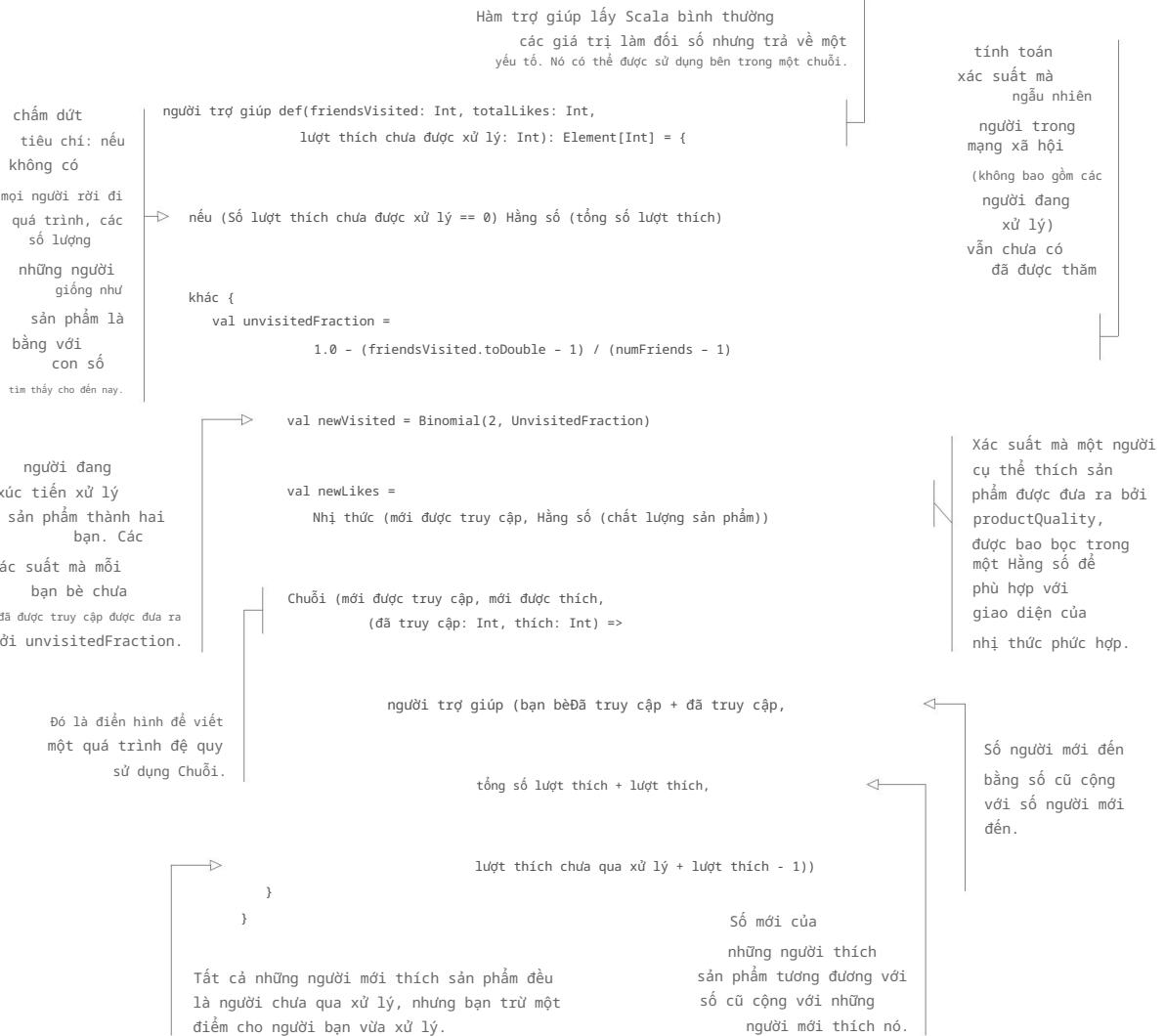
- Mục đích của hàm generateLikes là xác định số lượng những người thích sản phẩm sau khi đưa nó cho một mục tiêu có mạng xã hội chứa số lượng người nhất định. Chức năng này giả định rằng bản thân mục tiêu thích sản phẩm; nếu không, hàm sẽ không được gọi. Chức năng này mô phỏng một quá trình ngẫu nhiên của những người quảng cáo sản phẩm cho khách hàng của họ. Bạn bè nếu họ thích sản phẩm. Hàm generateLikes có hai đối số: (1) số người trong mạng xã hội của mục tiêu, là một Số nguyên và (2) chất lượng của sản phẩm, là Giá trị gấp đôi trong khoảng từ 0 đến 1.

Logic chính xác của hàm generateLikes không quan trọng, bởi vì điểm chính là bạn có thể sử dụng một hàm đệ quy thú vị như thế này như một CPDD. Nhưng tôi sẽ giải thích logic, để bạn có thể xem một ví dụ. Hầu hết các công việc của generateLikes được thực hiện bởi một hàm trợ giúp. Chức năng này theo dõi ba giá trị:

- friendsVisited nắm giữ số người trong mạng xã hội của mục tiêu những người đã được thông báo về sản phẩm. Điều này bắt đầu từ 1, bởi vì ban đầu mục tiêu đã được thông báo về sản phẩm.
- TotalLikes đại diện cho số lượng người, trong số những người đã từng đã truy cập cho đến nay, những người thích sản phẩm. Điều này cũng bắt đầu từ 1, bởi vì bạn giả sử rằng mục tiêu thích sản phẩm để tạoLikes được gọi.
- lượt thích chưa qua xử lý đại diện cho số người thích sản phẩm những người mà bạn chưa mô phỏng quảng cáo sản phẩm cho bạn bè của họ.

Tôi sẽ giải thích logic của hàm trợ giúp thông qua đoạn mã sau.

#### Lịch kê 5.4 Hàm trợ giúp để duyệt qua mạng xã hội

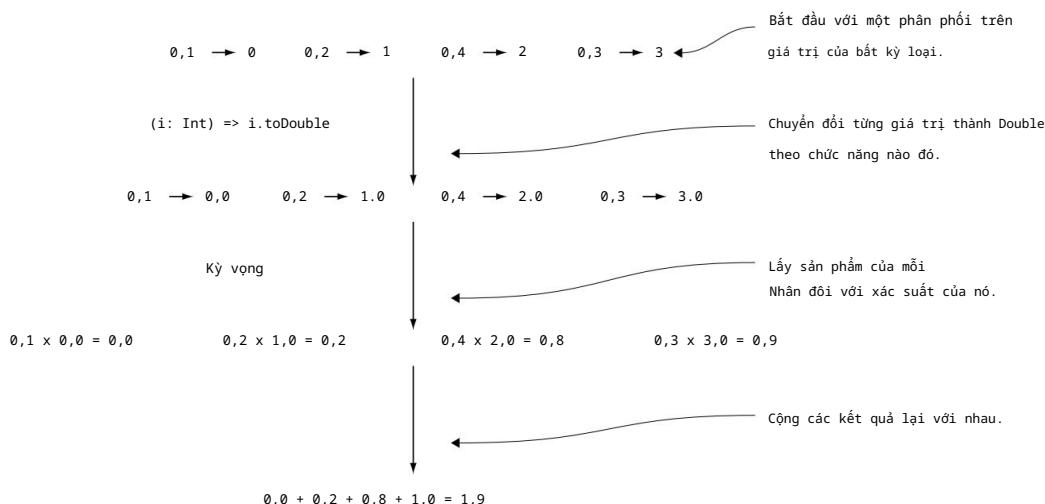


Ví dụ này sử dụng nhiều kỹ năng lập trình và Scala hơn, nhưng mô hình thiết yếu các kỹ thuật tương tự như mạng Bayesian. Điểm chính là để tưởng tượng một quá trình bằng cách mà một thế giới có thể được tạo ra. Trong ví dụ này, bạn đã thấy một tương đối đơn giản quá trình tuyên truyền sản phẩm qua mạng xã hội; một quy trình phong phú hơn có thể dễ dàng được mã hóa.

### 5.4.2 Lập luận với mô hình dự đoán thành công của sản phẩm

Bởi vì bạn đã thiết kế mô hình để dự đoán thành công, cách sử dụng thông thường sẽ là để đặt giá trị cho các hằng số kiểm soát và dự đoán số lượng người mua sản phẩm. Bởi vì số lượng người là một biến số nguyên có thể có phạm vi rộng phạm vi, bạn không thực sự quan tâm đến việc dự đoán xác suất của một giá trị cụ thể. Thay vào đó, bạn muốn biết giá trị trung bình mà bạn có thể mong đợi. Điều này được gọi là kỳ vọng của giá trị.

Trong lý thuyết xác suất, kỳ vọng là một khái niệm chung. Kỳ vọng mất một hàm được xác định trên một biến và trả về giá trị trung bình của hàm đó. Một ví dụ được thể hiện trong hình 5.13. Bạn bắt đầu với phân phối xác suất trên các giá trị của bất kỳ loại nào; trong ví dụ này, các giá trị là Số nguyên. Sau đó, bạn áp dụng một chức năng cho mỗi value để tạo ra một giá trị kép. Trong ví dụ này, hàm chuyển đổi từng Số nguyên thành biểu diễn kép của số nguyên. Tiếp theo, bạn lấy bình quân gia quyền của các giá trị Double này, trong đó mỗi giá trị Double được tính trọng số theo xác suất của nó. Cái này có nghĩa là bạn nhân từng giá trị Double với xác suất của nó và cộng các kết quả.

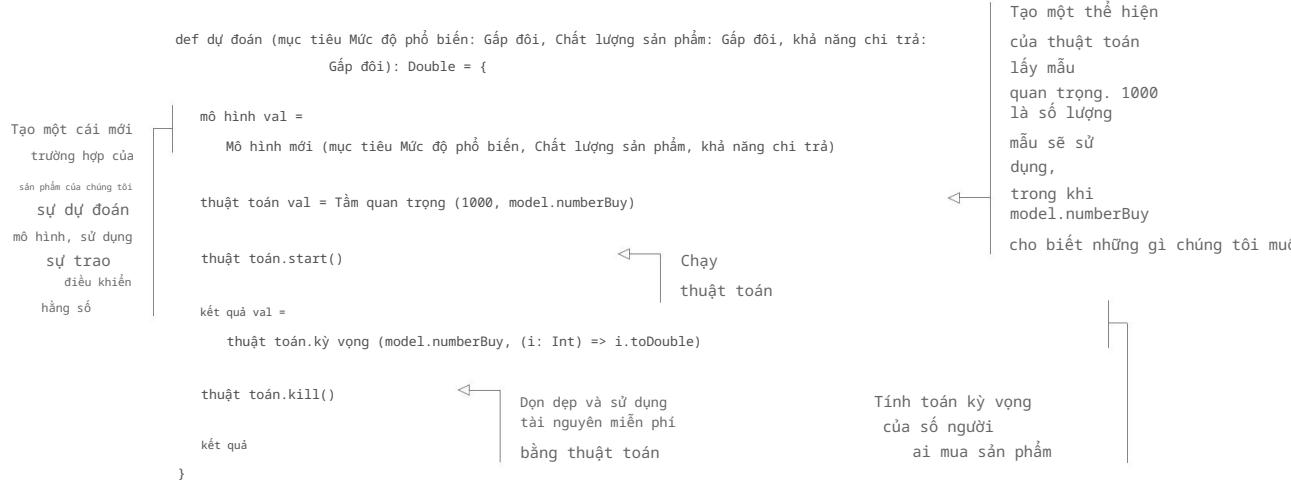


Hình 5.13 Tính toán kỳ vọng của phân phối trên một phần tử có giá trị Số nguyên. Các giá trị của phần tử đầu tiên được chuyển đổi thành Nhân đôi. Sau đó, bạn lấy giá trị trung bình của các Nhân đôi này, trong đó mỗi Nhân đôi được tính trọng số theo xác suất của nó.

Trong ví dụ về dự đoán thành công của sản phẩm, bạn muốn tính toán kỳ vọng của số người mua sản phẩm, là một biến số nguyên. Bạn có thể sử dụng dòng sau trong Figaro để làm điều này:

```
thuật toán.kỳ vọng (model.numberBuy, (i: Int) => i.toDouble)
```

Ở đây, thuật toán là phần xử lý của thuật toán suy luận mà bạn đang sử dụng. Ứng dụng này sử dụng một thuật toán lấy mẫu quan trọng, đây là một thuật toán đặc biệt tốt để dự đoán kết quả của một quá trình tổng hợp phức tạp. Bởi vì bạn cần xử lý thuật toán, bạn cần sử dụng mã phức tạp hơn một chút so với trước đây để chạy suy luận, như được giải thích trong đoạn mã sau. Toàn bộ quá trình lấy trong các hằng số kiểm soát và tính toán số lượng người dự kiến mua sản phẩm được thực hiện bởi một chức năng gọi là dự đoán:



Nếu bạn đang cố gắng hiểu tác động của các biện pháp kiểm soát khác nhau đối với số lượng người mua sản phẩm, bạn sẽ muốn chạy chức năng dự đoán này nhiều lần với các đầu vào khác nhau. Bạn có thể tạo ra một kết quả như thế này:

Mức độ phổ biến	Chất lượng sản phẩm	Giá cả phải chăng	Số lượng người mua
100	2.016999999999985	0,5	3.7759999999999962
100	0,5	0,9	29.2149999999997
100	0,9	0,5	53.13799999999996
10	0,5	0,9	0.7869999999999979
10	0,9	0,5	1.4769999999999976
10	0,5	0,9	3.341999999999985
10	0,9	0,9	6.066999999999985

Bạn có thể kết luận một vài điều từ bảng này. Số lượng người mua dường như gần như tỷ lệ thuận với khả năng chi trả của sản phẩm. Nhưng có một sự phụ thuộc không cân xứng vào chất lượng sản phẩm: đối với mọi trường hợp mức độ phổ biến và khả năng chi trả, số lượng người mua khi chất lượng sản phẩm bằng 0,9 ít nhất gấp vài lần cao như khi chất lượng là 0,5. Khi mức độ phổ biến là 100, nó cao gấp khoảng 15 lần. Dường như có sự tương tác giữa mức độ phổ biến và chất lượng sản phẩm, trong đó

sự phỏ biến đặt ra giới hạn về số lượng người sẽ đạt được khi chất lượng cao. Khi chất lượng thấp, mức độ phỏ biến không quan trọng bằng.

Trong ví dụ này, bạn đã thấy Figaro được sử dụng làm ngôn ngữ mô phỏng. dự đoán cái gì sẽ xảy ra trong tương lai thường là những gì mô phỏng làm và đây là trường hợp sử dụng được mô tả ở đây. Bạn có thể dễ dàng sử dụng mô hình để lập luận ngược. Vì Ví dụ, sau khi bạn phát sản phẩm cho một số người và quan sát xem có bao nhiêu người khác những người mà họ đã ảnh hưởng để mua sản phẩm, bạn có thể ước tính chất lượng của sản phẩm. Đây có thể là một cách sử dụng thay thế có giá trị của mô hình.

## 5.5 Sử dụng mạng Markov

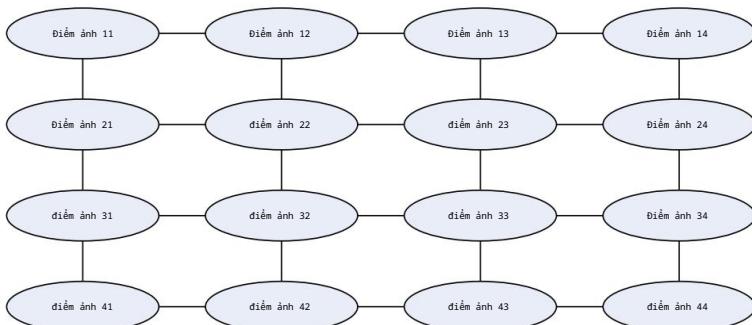
Các phần trước đã đề cập đến các mạng Bayesian, mã hóa phụ thuộc trực tiếp. Đến lúc chuyển sự chú ý của bạn sang các phụ thuộc vô hướng. Đối tác với các mạng Bayesian cho các phụ thuộc vô hướng là các công trình mạng Markov. Tôi sẽ giải thích các nguyên tắc đằng sau mạng Markov bằng ứng dụng khôi phục hình ảnh diễn hình. Sau đó, tôi sẽ chỉ cho bạn cách biểu diễn và lập luận với mô hình khôi phục ảnh trong Figaro.

### 5.5.1 Mạng Markov được xác định

Mạng Markov là một đại diện của một mô hình xác suất bao gồm ba điều:

- Một tập hợp các biến—Mỗi biến có một miền, là tập hợp các giá trị có thể của biến.
- Một đồ thị vô hướng trong đó các nút là các biến—Các cạnh giữa các nút là vô hướng, nghĩa là chúng không có mũi tên từ biến này sang biến khác. Cái này đồ thị được phép có chu trình.
- Một tập hợp các điện tử—Những điện tử này cung cấp các thông số số của người mẫu. Tôi sẽ giải thích chi tiết tiềm năng trong giây lát.

Hình 5.14 hiển thị mạng Markov cho ứng dụng khôi phục ảnh. Có một biến có thể cho mọi pixel trong hình ảnh. Hình này hiển thị một mảng pixel 4x4, nhưng thật dễ dàng



Hình 5.14 Mạng Markov cho ảnh pixel

dể xem làm thế nào nó có thể được khái quát hóa cho bất kỳ hình ảnh kích thước nào. Về nguyên tắc, giá trị của một pixel có thể là bất kỳ màu nào, nhưng để ví dụ, giả sử đó là một Boolean đại diện pixel sáng hay tối. Có một cạnh giữa bất kỳ cặp pixel nào kè nhau theo chiều ngang hoặc chiều dọc. Theo trực giác, các cạnh này mã hóa thực tế rằng, tất cả những thứ khác đều bằng nhau, hai pixel liền kề có nhiều khả năng có cùng giá trị hơn các giá trị khác nhau.

Vòng loại “tất cả những thứ khác đều bình đẳng” này rất quan trọng để hiểu ý nghĩa của Mô hình này. Nếu bạn bỏ qua cạnh giữa hai pixel trong giây lát và xem xét phân phối xác suất riêng lẻ trên từng pixel, có khả năng là trên thực tế, chúng khác nhau. Ví dụ, dựa trên mọi thứ khác mà bạn biết, bạn có thể tin rằng pixel 11 sáng với xác suất 90% và pixel 12 sáng với xác suất 10%. Trong trường hợp này, rất có thể pixel 11 và pixel 12 khác nhau. Nhưng ranh giới giữa pixel 11 và pixel 12 khiếu chúng có nhiều khả năng trở thành giống như họ sẽ có được. Nó bổ sung thêm một phần kiến thức rằng họ đang có khả năng là như nhau. Kiến thức này đối trọng với kiến thức khác mà chúng có thể khác nhau, nhưng nó có thể không thay đổi hoàn toàn kết luận tổng thể. Kiến thức cụ thể được biểu thị bằng cạnh giữa pixel 11 và pixel 12 là được biểu diễn bằng điện thế trên cạnh đó. Vậy giờ hãy xem chính xác tiềm năng như thế nào dưới c đì nh nghĩa.

### TIỀM NĂNG

Các tham số số của mạng Markov được xác định như thế nào? Trong mạng Bayesian, mỗi biến có một CPD. Trong mạng Markov, nó không đơn giản như vậy. Các biến không sở hữu các tham số số của chúng. Thay vào đó, các chức năng được gọi là tiềm năng được xác định trên tập hợp các biến. Khi có sự phụ thuộc đối xứng, một số trạng thái chung của các biến phụ thuộc vào nhau có nhiều khả năng xảy ra hơn những trạng thái khác, tất cả những trạng thái khác là bình đẳng. Tiềm năng chỉ định một trọng số cho mỗi trạng thái chung như vậy. Các quốc gia chung với cao trọng số có nhiều khả năng hơn các trạng thái khớp có trọng số thấp, tất cả những thứ khác đều bằng nhau. Xác suất tương đối của hai trạng thái khớp bằng tỷ lệ giữa trọng số của chúng, một lần nữa, tất cả những thứ khác đều bình đẳng.

Về mặt toán học, một tiềm năng chỉ đơn giản là một hàm từ các giá trị của các biến thành thực con số. Chỉ các số thực dương hoặc số 0 mới được phép làm giá trị của một điện thế.

Bảng 5.1 cho thấy một ví dụ về điện thế đơn nguyên trên một pixel cho ứng dụng khôi phục hình ảnh và bảng 5.2 cho thấy điện thế nhị phân trên hai pixel.

Bảng 5.1 Điện thế đơn nguyên trên một pixel. Tiềm năng này mã hóa một thực tế rằng, tất cả những thứ khác đều bằng nhau, một pixel được thấp sáng với xác suất 0,4.

điểm ảnh 31	Giá trị tiềm năng
F	0,6
t	0,4

Bảng 5.2 Một điện thế nhị phân trên hai pixel liền kề. Tiềm năng này mã hóa một thực tế rằng, nếu tất cả những thứ khác đều bằng nhau, hai pixel có khả năng có cùng giá trị cao gấp 9 lần so với các giá trị khác nhau.

diễn ảnh 31	diễn ảnh 32	Giá trị tiềm năng
F	F	0,9
F	t	0,1
t	F	0,1
t	t	0,9

Làm thế nào để các chức năng tiềm năng tương tác với cấu trúc đồ thị? Có hai quy tắc:

- Một hàm tiềm năng chỉ có thể đề cập đến các biến được kết nối trong biểu đồ.
- Nếu hai biến được kết nối trong biểu đồ, chúng phải được đề cập cùng nhau bởi một số chức năng tiềm năng.

Trong ví dụ khôi phục hình ảnh của chúng tôi, mọi biến sẽ có một bản sao của tiềm năng đơn nguyên trong bảng 5.2 và mọi cặp pixel liền kề, theo chiều ngang hoặc chiều dọc, sẽ có một bản sao của thế nhị phân trong bảng 5.2. Bạn có thể thấy rằng hai quy tắc được tôn trọng bởi sự phân công tiềm năng này.

#### CÁCH MẠNG MARKOV XÁC ĐỊNH PHÂN PHỐI XÁC SUẤT

Bạn đã thấy mạng Markov được định nghĩa như thế nào. Nó xác định phân phối xác suất như thế nào? Làm thế nào để gán một xác suất cho mọi thế giới khả dĩ sao cho các ràng buộc xác suất của tất cả các thế giới khả dĩ cộng lại bằng 1? Câu trả lời không hoàn toàn đơn giản như đối với Bayesian mạng nhưng cũng không quá phức tạp.

Giống như trong mạng Bayesian, một thế giới có thể có trong mạng Markov bao gồm một gán giá trị cho tất cả các biến, đảm bảo rằng giá trị của từng biến nằm trong miền của nó. Xác suất của một thế giới có thể như vậy là gì? Hãy xây dựng nó lên mảnh từng mảnh bằng cách sử dụng một ví dụ.

Để đơn giản hóa mọi thứ, hãy xem xét một mảng pixel  $2 \times 2$  như sau  
gán giá trị: pixel 11 = true, pixel 12 = true, pixel 21 = true, pixel 22 = false.  
Bạn sẽ xem xét tất cả các tiềm năng trong mô hình và các giá trị tiềm năng của chúng cho khả năng này thế giới. Đối với các điện thế đơn nguyên, bạn có các giá trị trong bảng 5.3. Pixel đúng sự thật

Bảng 5.3 Các giá trị tiềm năng cho các tiềm năng một đơn vị, ví dụ thế giới có thể

Biến đổi	Giá trị tiềm năng
Điểm ảnh 11	0,4
Điểm ảnh 12	0,4
Điểm ảnh 21	0,4
điểm ảnh 22	0,6

có giá trị tiệm nồng 0,4, trong khi một pixel sai có giá trị tiệm nồng 0,6.

Bảng 5.4 cho thấy các giá trị tiệm nồng từ bốn điểm ảnh nhị phân. Các trường hợp mà hai pixel có cùng giá trị có giá trị tiệm nồng 0,9, trong khi hai trường hợp còn lại có giá trị tiệm nồng 0,1. Điều này bao gồm tất cả các tiệm nồng trong mô hình.

Bảng 5.4 Giá trị tiệm nồng cho tiệm nồng nhị phân ví dụ thế giới có thể

Biểu 1	Biểu 2	Giá trị tiệm nồng
Điểm ảnh 11	Điểm ảnh 12	0,9
Điểm ảnh 21	điểm ảnh 22	0,1
Điểm ảnh 11	Điểm ảnh 21	0,9
Điểm ảnh 12	điểm ảnh 22	0,1

Tiếp theo, bạn nhân các giá trị tiệm nồng từ tất cả các tiệm nồng. Trong ví dụ của chúng tôi, bạn lấy  $0,4 \times 0,4 \times 0,4 \times 0,6 \times 0,9 \times 0,1 \times 0,9 \times 0,1 = 0,00031104$ . Tại sao bạn nhân lên?

Hãy suy nghĩ về nguyên tắc “tất cả những thứ khác đều bình đẳng”. Nếu hai thế giới có cùng xác suất ngoại trừ một tiệm nồng, thì xác suất của các thế giới tỷ lệ thuận với giá trị tiệm nồng của chúng theo tiệm nồng đó. Đây chính xác là hiệu ứng bạn nhận được khi bạn nhân xác suất với giá trị của tiệm nồng này. Tiếp tục lý do này, bạn nhân các giá trị tiệm nồng của tất cả các tiệm nồng để có được “xác suất” của một thế giới khả thi.

Tôi đặt “xác suất” trong ngoặc kép vì nó không thực sự là một xác suất. Khi bạn nhân các giá trị tiệm nồng theo cách này, bạn sẽ thấy rằng “xác suất” không có tổng bằng 1. Điều này dễ dàng được khắc phục. Để có được xác suất của bất kỳ thế giới khả dĩ nào, bạn chuẩn hóa “xác suất” được tính bằng cách nhân các giá trị tiệm nồng. Bạn gọi đây là những xác suất không bình thường. Tổng của các xác suất không chuẩn hóa này được gọi là hệ số chuẩn hóa và thường được ký hiệu bằng chữ Z. Vì vậy, bạn lấy xác suất không chuẩn hóa xác suất và chia chúng cho Z để có xác suất. Đừng lo lắng nếu quá trình này nghe có vẻ rườm rà với bạn; Figaro chăm sóc tất cả.

Một điểm đáng ngạc nhiên đến từ cuộc thảo luận này. Trong một mạng Bayesian, bạn có thể tính toán xác suất của một thế giới có thể bằng cách nhân các mục CPD có liên quan. TRONG mạng Markov, bạn không thể xác định xác suất của bất kỳ thế giới khả dĩ nào nếu không có xem xét tất cả các thế giới có thể. Bạn cần tính xác suất không chuẩn hóa của mọi thế giới có thể để tính hệ số chuẩn hóa. Vì lý do này, một số người thấy rằng việc biểu diễn mạng Markov khó hơn mạng Bayesian, bởi vì nó khó hơn để giải thích các con số như xác định một xác suất. Tôi nói rằng nếu bạn tiếp tục lưu ý đến nguyên tắc “tất cả những thứ khác đều bình đẳng”, bạn sẽ có thể xác định các tham số của một Mạng Markov với sự tự tin. Bạn có thể để việc tính toán hệ số chuẩn hóa cho Figaro. Tất nhiên, các tham số của cả mạng Bayes và mạng Markov có thể được học từ dữ liệu. Sử dụng bất kỳ cấu trúc nào có vẻ phù hợp hơn với bạn, dựa trên các loại mối quan hệ trong ứng dụng của bạn.

### 5.5.2 Biểu diễn và lập luận với mạng Markov

Có một cách mạng Markov chắc chắn đơn giản hơn mạng Bayesian: trong các mẫu lập luận. Mạng Markov không có khái niệm về sự phụ thuộc cảm ứng. Bạn có thể suy luận từ biến này sang biến khác theo bất kỳ con đường nào, miễn là đường dẫn không bị chặn bởi một biến đã được quan sát. Hai biến phụ thuộc nếu có một đường dẫn giữa chúng và chúng trở nên độc lập có điều kiện với một tập hợp của các biến nếu các biến đó chặn tất cả các đường dẫn giữa hai biến. Đó là tất cả ở đó là để nó.

Ngoài ra, vì tất cả các cạnh trong mạng Markov đều vô hướng, nên không có khái niệm về nhân quả hay quá khứ và tương lai. Bạn thường không nghĩ đến những nhiệm vụ như dự đoán kết quả trong tương lai hoặc suy ra nguyên nhân trong quá khứ của các quan sát hiện tại. Thay vào đó, bạn chỉ cần suy ra các giá trị của một số biến, đưa ra các biến khác.

#### ĐẠI DIỆN MÔ HÌNH PHỤC HỒI ẢNH TRONG FIGARO

Trong ứng dụng khôi phục hình ảnh, bạn sẽ cho rằng một số điểm ảnh được quan sát và phần còn lại không quan sát được. Bạn muốn khôi phục các pixel không được quan sát. Bạn sẽ sử dụng mô hình được mô tả trong phần trước, trong đó xác định cả giá trị tiềm năng cho mỗi pixel đang bật và giá trị tiềm năng cho các pixel liền kề có cùng giá trị. Đây là mã Figaro để đại diện cho mô hình. Hãy nhớ rằng trong phần 5.1.2, tôi cho biết có hai phương pháp để chỉ định các mối quan hệ đối xứng, một ràng buộc phương pháp và phương pháp điều kiện. Mã này sử dụng phương thức ràng buộc:

```
val pixel = Array.fill(10, 10)(Flip(0,4))

def setConstraint(i1: Int, j1: Int, i2: Int, j2: Int) {
    val pixel1 = pixel(i1)(j1) val pixel2 =
    pixel(i2)(j2) val pair = ^^(pixel1,
    pixel2) pair.addConstraint(bb => if (bb._1
    == bb._2) 0.9; khác 0,1)
}

vì {
    tôi <- 0 cho đến 10
    j <- 0 cho đến 10 }

{ if (i <= 8) setConstraint(i, j, i+1, j) if (j <= 8)
    setConstraint(i, j, i, j+1)
}
```

Đặt ràng buộc đơn nguyên  
trên mỗi biến

Đặt ràng buộc  
nhị phân trên một  
cặp biến cho tọa  
độ của chúng

Áp dụng ràng buộc  
nhị phân cho tất cả  
các cặp biến liền kề

Một vài lưu ý về mã này theo thứ tự:

- Trong định nghĩa pixel, `Array.fill(10, 10)(Flip(0.4))` tạo ra một ô  $10 \times 10$  mảng và lấp đầy mọi phần tử của mảng bằng một thê hiện khác của `Flip(0,4)`. Tất cả các pixel khác nhau được xác định bởi các phần tử Lật khác nhau, đó là quan trọng bởi vì tất cả chúng có thể có các giá trị khác nhau.
- Bạn có thể thắc mắc tại sao một phần tử `Flip` lại được sử dụng cho các điện thế đơn nguyên chứ không phải là một ràng buộc. Đối với một điện thế đơn nguyên, xác định nó theo cách thông thường

Cách Figaro hoặc sử dụng một ràng buộc có tác dụng tương tự. Trong trường hợp này, Flip sẽ trở thành đúng với xác suất 0,4 và sai với xác suất 0,6. Những xác suất này sẽ được nhân lên thành xác suất phi chuẩn hóa của một thế giới có thể xảy ra, giống như thế chúng đã được chỉ định thông qua một ràng buộc.

Trên thực tế, mọi phần tử Figaro phải được xác định theo cách thông thường, sử dụng một số loại hàm tạo phần tử, ngay cả khi bạn đang sử dụng Figaro để mã hóa Markov mạng. Nếu phần tử của bạn không có ràng buộc đơn nguyên, Figaro bình thường này nhà xây dựng nên trung lập và không ứng hộ bất kỳ thế giới có thể nào hơn thế giới khác. Bạn có thể sử dụng Flip(0,5) hoặc phần tử Đồng nhất để đạt được điều này.

- Trong định nghĩa của setConstraint, ^^ là hàm tạo cặp Figaro. Vì thế  $\wedge\wedge(\text{pixel1}, \text{pixel2})$  tạo một phần tử có giá trị là cặp giá trị của phần tử pixel1 và pixel2.
- Trong phần hiểu, 0 đến 10 là Scala cho các số nguyên từ 0 đến 10, loại trừ; nói cách khác, các số nguyên 0, 1, .., 9. Nếu bạn muốn nói bao gồm từ 0 đến 10, bạn sẽ sử dụng 0 đến 10.
- Điều này để dễ hiểu cũng cho thấy một ví dụ về vòng lặp chồng nhau. Trong các ngôn ngữ khác, điều này sẽ được viết bằng cách sử dụng một vòng lặp for bên trong một vòng lặp khác. TRONG Scala, bạn có thể đặt cả hai vòng giống nhau cho tiêu đề.

#### LÝ LUẬN VỚI MÔ HÌNH PHỤC HỒI HÌNH ẢNH

Bạn muốn sử dụng mô hình khôi phục hình ảnh để suy ra giá trị của các pixel không được quan sát, đưa ra các pixel được quan sát. Bạn cần ba điều: một cách để nhập và xử lý bằng chứng, một cách để tính toán các trạng thái pixel có khả năng xảy ra nhất và một cách để xem kết quả. Hãy xem xét từng cái một.

Bằng chứng quy trình: Nếu bạn có một mảng pixel  $10 \times 10$ , thì bạn có thể có dữ liệu là  $10 \times$  mảng 10 ký tự, trong đó mỗi ký tự là 0 cho tắt, 1 cho bật hoặc ? cho chưa biết.

Bạn có thể xử lý dữ liệu này bằng cách sử dụng hàm setEvidence đơn giản sau :

```
def setEvidence(data: String) = {
  cho { n <- 0 cho đến khi data.length } {
    giá trị tôi = n / 10
    giá trị j = n % 10
    dữ liệu (n) khớp {
      case '0' => pixels(i)(j).observe(false) case '1' => pixels(i)
        (j).observe(true) => ()
        trường hợp -
    }
  }
}
```

Sử dụng khung mẫu Scala, trong trường hợp này rất giống câu lệnh switch trong các ngôn ngữ khác. Chỉ ra một trường hợp mặc định. Vì vậy, không quan sát sẽ được thực hiện trên một '?'.

Tính toán trạng thái pixel có khả năng nhất: Ví dụ này giới thiệu một loại truy vấn mới mà bạn đã không xem xét trước đây. Trước đây, bạn muốn ước tính xác suất sau của các nguyên tố. Lần này, bạn quan tâm đến các giá trị có khả năng xảy ra nhất của các phần tử (các giá trị có xác suất cao nhất). Nhưng bạn không quan tâm đến các giá trị có khả năng nhất của các phần tử riêng lẻ, mà không quan tâm đến các phần tử khác; thay vào đó, bạn muốn

việc gán giá trị chung có khả năng nhất cho tất cả các biến. Bạn muốn biết những gì thế giới có khả năng nhất là.

Truy vấn này được biết đến trong Figaro như một **truy vấn giải thích** có thể xảy ra nhất (MPE) , bởi vì bạn muốn biết thế giới đó là lời giải thích hợp lý nhất cho dữ liệu. Các thuật toán cho các truy vấn MPE khác với các thuật toán tính toán xác suất bạn đã thấy cho đến nay, mặc dù chúng có liên quan với nhau. Trong ví dụ này, bạn sẽ sử dụng một phiên bản của tuyên truyền niềm tin được thiết kế để tính toán MPE. Thuật toán này được gọi là Tuyên truyền niềm tin MPE. Truyền bá niềm tin là một thuật toán lặp đi lặp lại và bạn có thể kiểm soát số lần lặp lại với một tham số. Trong trường hợp này, bạn sẽ sử dụng 10 lần lặp lại. Bạn có thể tạo một phiên bản của thuật toán MPEBeliefPropagation và yêu cầu nó chạy trong cách này:

```
thuật toán val = MPEBeliefPropagation(10)
thuật toán.start()
```

Xem kết quả: Đây đơn giản là vấn đề xem qua tất cả các pixel, nhận các giá trị có khả năng nhất của chúng và in chúng cho phù hợp. Bạn có thể có được nhiều khả năng nhất giá trị của một phần tử bằng cách sử dụng phương pháp mostLikelyValue của MPEBeliefPropagation. Đây là mã:

```
vì {
    tôi <- 0 cho đến 10
} {
    cho { j <- 0 cho đến 10 } {
        val mlv = thuật toán.mostLikelyValue(pixels(i)(j))
        if (mlv) print('1') other print('0')
    }
    println()
}
```

Để chạy mô hình, bạn cần cung cấp cho nó một số đầu vào. Thông thường, đây sẽ là đọc từ một tệp hoặc được cung cấp theo chương trình bởi một mô-đun khác. Để đơn giản hóa ví dụ này, bạn có thể xác định đầu vào trực tiếp trong chương trình như sau:

```
dữ liệu giá trị =
"""00?000?00
0?010?0010
110?010011
11?000111
11011000?1
1?0?100?10
00001?0?00
0010?0?100
01?01001?0
0??000110?""".filterNot(_.isWhitespace)

setEvidence(dữ liệu)
```

của Scala "" người xây dựng  
cho phép bạn tạo các chuỗi dài trải rộng  
trên nhiều dòng. Bạn lọc  
bỏ tất cả các ký  
tự khoảng trắng để tạo  
ra chuỗi 100 ký  
tự.

Khi chạy trên dữ liệu này, chương trình xuất ra như sau:

```
0000000000
0001000010
1100010011
1100000111
1101100011
1000100010
0000100000
0010000100
0100100100
0000001100
```

Đó là tất cả những gì có đối với mạng Markov. Đây là một chương dài, nhưng bạn đã học được rất nhiều. Bây giờ bạn đã biết tất cả các nguyên tắc chính của các mô hình xác suất và có thể viết các chương trình xác suất cho nhiều ứng dụng khác nhau.

## 5.6 Tóm tắt

- Mô hình xác suất là tất cả về mã hóa các mối quan hệ giữa các biến.  
Các mối quan hệ đối xứng tạo ra các phụ thuộc có hướng, trong khi các mối quan hệ bất đối xứng tạo ra các phụ thuộc vô hướng.
- Các quan hệ phụ thuộc có định hướng dẫn từ nguyên nhân đến kết quả. Có nhiều loại mối quan hệ nhân quả.
- Các mạng Bayes mã hóa các phụ thuộc có hướng bằng cách sử dụng một chu kỳ có hướng đồ thị.
- Hướng của các mũi tên trong mạng Bayes không nhất thiết phải là hướng suy luận. Mạng Bayesian có thể được sử dụng để suy luận theo mọi hướng trong mạng.
- Mạng Markov mã hóa các phụ thuộc vô hướng bằng cách sử dụng một đồ thị.
- Nếu bạn có thể xác định các loại mối quan hệ giữa các biến trong mô hình của mình và sử dụng chúng để viết chương trình của bạn, bạn sẽ không sai lầm đâu.

## 5.7 Bài tập

Các giải pháp cho các bài tập chọn lọc có sẵn trực tuyến tại [www.manning.com/books/realistic-probabilistic-programming](http://www.manning.com/books/realistic-probabilistic-programming).

- 1 Đối với mỗi cặp biến sau đây, hãy quyết định xem mối quan hệ phụ thuộc giữa chúng là có hướng hay vô hướng và nếu có hướng thì mũi tên sẽ đi theo hướng nào. a Bài xì phé của người chơi A và tiền cược của người chơi b Bài xì phé của người chơi 1 và quân bài xì phé của người chơi 2 c Tâm trạng của tôi và thời tiết hôm nay d Tâm trạng của tôi và việc tôi đã ăn sáng e Nhiệt độ trong phòng khách của tôi và cài đặt bộ điều nhiệt trong nhà

f Nhiệt độ trong phòng khách của tôi và chỉ số nhiệt kế trong phòng khách phòng ngủ

g Nhiệt độ trong phòng khách của tôi và nhiệt độ trong bếp h Chủ đề của một tin bài và nội dung của bài viết i Phần tóm tắt của một tin tức và nội dung của bài viết

2 Đối với mỗi bộ biến sau đây, hãy vẽ một mạng Bayes trên biến.

quản bài xì phé của Người chơi 1, quản bài xì phé của người chơi 2, tiền cược của người chơi 1 và của người chơi 2 đặt cược tiếp theo

b Tâm trạng của tôi khi thức dậy, tâm trạng của tôi lúc 10 giờ sáng, thời tiết hôm nay và liệu tôi đã ăn sáng chưa

c Nhiệt độ trong phòng khách của tôi, nhiệt độ trong bếp, cài đặt bộ điều nhiệt trong nhà, chỉ số nhiệt kế trong phòng khách và chỉ số nhiệt kế trong bếp

d Chủ đề của một tin tức, tóm tắt của bài báo, nội dung của bài viết và các bình luận về bài viết

3 Nhiệm vụ của bạn là thiết kế một mạng Bayes để mô hình hóa quá trình nấu súp.

Mục tiêu của mạng này là giúp bạn quyết định nên sử dụng nguyên liệu nào và lượng từng nguyên liệu, cũng như các biến số nấu ăn như lượng nhiệt và thời gian, nhằm tối ưu hóa các chất lượng thực phẩm khác nhau như độ cay và độ ngọt.

a Các biến trong mô hình của bạn là gì? b Vẽ

cấu trúc mạng Bayes trên các biến này. c Chọn các dạng hàm

Figaro cho mỗi biến của bạn. d Điền vào các dạng hàm của bạn trong

Figaro với các tham số số. e Sử dụng mô hình Figaro của bạn để trả lời các câu hỏi như bạn nên nấu súp trong bao lâu với một bộ nguyên liệu nhất định để đảm bảo độ sánh mịn tối ưu.

4 Một trận quần vợt bao gồm nhiều hiệp, mỗi hiệp bao gồm một số ván đấu. Người chơi đầu tiên thắng hai set sẽ thắng trận đấu. Người chơi đầu tiên thắng sáu ván sẽ thắng một set. (Hãy bỏ qua những người bẻ hòa, nhưng sau khi thực hiện bài tập 5, bạn sẽ có thể làm mẫu cho họ.) Người chơi luôn phiêu giao bóng cho một trò chơi tại một thời điểm. Viết một chương trình Figaro nhận hai đối số-xác suất mà mỗi người chơi thắng một trò chơi mà anh ta là người phục vụ-và dự đoán người chiến thắng trong trận đấu.

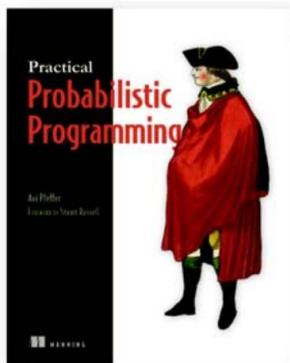
5 Nay giờ, hãy xây dựng mô hình quần vợt của bạn để mô hình hóa các điểm riêng lẻ. Trong một trò chơi, người chơi cố gắng để có được điểm. Người chơi đầu tiên giành được bốn điểm sẽ thắng trò chơi, trừ khi cả hai người chơi đều đạt được ba điểm. Trong trường hợp đó, người chơi đầu tiên đi trước đối thủ hai điểm sẽ thắng trò chơi. Viết một chương trình Figaro nhận hai đối số như trước, ngoại trừ việc bây giờ chúng là xác suất mà mỗi đối số

người chơi giành được một điểm trong đó anh ta là máy chủ. Một lần nữa, chương trình của bạn sẽ dự đoán người chiến thắng trong trận đấu.

- 6 Hôm nay bạn có thể xây dựng thêm mô hình quần vợt để mô hình hóa các điểm riêng lẻ dưới dạng các cuộc biểu tình. Người chơi có thể có các biến số như khả năng giao bóng, tốc độ và mức độ mắc lỗi. Mô hình của bạn có thể chi tiết như bạn muốn, bao gồm cả vị trí của các cầu thủ và quả bóng ở mỗi cú đánh trong cuộc biểu tình.
- 7 Nhà tôi có điều hòa không khí trung tâm được điều khiển bằng bộ điều nhiệt ở tầng dưới. Tầng trên cùng thường nóng hơn tầng trệt. Tạo một mạng Markov đại diện cho nhiệt độ trong nhà (hôm nay bỏ qua bộ điều nhiệt). Viết mô hình Figaro để biểu diễn mạng. Sử dụng mô hình để tính xác suất mà tầng trên cùng có nhiệt độ ít nhất là 80 độ F, với điều kiện là tầng trệt là 72 độ F.
- 8 Hôm nay, hãy thêm bộ điều nhiệt vào mô hình cũng như nhiệt độ bên ngoài và liệu có cửa sổ nào đang mở hay không. Mô hình này sẽ kết hợp các phụ thuộc có hướng và không có hướng, do đó, nó sẽ không phải là một mạng Markov thuần túy, nhưng nó đủ dễ dàng để thực hiện sự kết hợp này trong Figaro. Viết chương trình Figaro và sử dụng nó để quyết định xem tôi có nên mở cửa sổ ở tầng trên cùng hay không.
- 9 Xem xét ứng dụng bộ lọc thư rác từ chương 3. Trong ứng dụng đó, mọi email được coi là độc lập. Hôm nay, giả sử bạn có nhiều email từ cùng một người gửi. Tình trạng thư rác của họ sẽ có mối tương quan cao. **a** Tạo mạng Markov để nắm bắt các mối tương quan này. **b** Mạng Bayesian cho mỗi thư rác được thể hiện trong hình 3.8. Sao chép mạng này trong mạng Markov từ bài tập 9a. Một lần nữa, đây là mạng kết hợp có hướng và vô hướng.

Lưu ý: Mặc dù tôi đã nói rằng lớp của một đối tượng xác định các thuộc tính của nó gây ra đồng minh, đó là cách tiếp cận được sử dụng trong bộ lọc thư rác, nhưng trong một số trường hợp, nên đi theo hướng ngược lại. Đây là trường hợp khi tất cả các tính năng luôn được quan sát, như trường hợp của bộ lọc thư rác. Trong trường hợp này, có thể lãng phí khi lập mô hình phân phối xác suất trên các đặc điểm được quan sát một cách rõ ràng. Thay vào đó, một mô hình có thể được tạo trong đó các tính năng của mỗi email xác định loại của một e-mail. Các lớp của email sau đó được liên kết bởi mạng Markov từ bài tập 9a. Loại mô hình này, được gọi là trường ngẫu nhiên có điều kiện, được sử dụng rộng rãi trong các ứng dụng hiểu ngôn ngữ tự nhiên và thị giác máy tính.

Tôi sẽ không đi vào chi tiết các trường ngẫu nhiên có điều kiện ở đây, nhưng tôi sẽ lưu ý, đối với những người đã quen thuộc với chúng, rằng chúng có thể dễ dàng được biểu diễn trong Figaro. Bí quyết là đảm bảo rằng các tính năng email được quan sát không phải là các phần tử Figaro mà là các biến Scala giúp xác định phân phối trên phần tử biểu thị liệu email có phải là thư rác hay không. Tất nhiên, mọi tham số có thể học được của mô hình sẽ là các phần tử Figaro và chúng có thể tương tác với các biến Scala đại diện cho các tính năng để xác định xác suất thư rác.



Dữ liệu bạn tích lũy về khách hàng, sản phẩm và người dùng trang web có thể giúp bạn không chỉ diễn giải quá khứ mà còn dự đoán tương lai của bạn!

Lập trình xác suất sử dụng mã để rút ra các suy luận xác suất từ dữ liệu. Bằng cách áp dụng các thuật toán chuyên biệt, các chương trình của bạn chỉ định mức độ xác suất cho kết luận. Điều này có nghĩa là bạn có thể dự báo các sự kiện trong tương lai như xu hướng bán hàng, lỗi hệ thống máy tính, thử nghiệm kết quả, và nhiều mối quan tâm quan trọng khác.

#### **Lập trình xác suất thực tế** giới thiệu

lập trình viên đang làm việc sang lập trình xác suất. TRONG cuốn sách này, bạn sẽ ngay lập tức làm việc với các bài kiểm tra thực tế như xây dựng bộ lọc thư rác, chẩn đoán các vấn đề về dữ

liệu hệ thống máy tính và khôi phục hình ảnh kỹ thuật số. Bạn sẽ khám phá ra xác suất suy luận, trong đó các thuật toán giúp đưa ra dự đoán mở rộng về các vấn đề như xã hội sử dụng phương tiện truyền thông. Đồng thời, bạn sẽ học cách sử dụng lập trình kiểu chức năng cho văn bản phân tích, các mô hình hướng đối tượng để dự đoán các hiện tượng xã hội như sự lan rộng của tweet và các mô hình vũ trụ mở để đánh giá mức độ sử dụng mạng xã hội trong đời thực. Cuốn sách cũng có các chương về cách các mô hình xác suất có thể giúp ích trong việc ra quyết định và lập mô hình của các hệ thống động.

#### **Có gì bên trong**

- Giới thiệu về mô hình xác suất
- Viết chương trình xác suất trong Figaro
- Xây dựng mạng Bayes
- Dự đoán vòng đời sản phẩm
- Thuật toán ra quyết định

Cuốn sách này giả định không có tiếp xúc trước với lập trình xác suất. Kiến thức về Scala là hữu ích.

# mục lục

---

## ký hiệu

+ toán tử 10

## sô ho c

Mô hình tuyến tính 2-D 68

Không gian 2 chiều 73

tăng độ chính xác

107 hàm precision(), gói dự báo 35, 48 ACF  
(chức năng tự tương quan) biểu đồ 53 hàm  
acf() 35, 53 kích hoạt  
mạng thần kinh 86 chức năng  
kích hoạt 69, 75, 78-83, 88 cấu hình  
kích hoạt 79 ADF  
(Dickey-Tăng cường Fuller) test 54 hàm  
adf.test(), gói tseries 36, 54 thambi mý 9-10

thuật toán 102, 172 alpha

thuật ngữ 83 phân

tích dữ liệu, trong nghiên cứu tình huống phân loại bài đăng  
trên Reddit 124-128

hàm AND 69-70

Các mô hình dự báo ARIMA (độ tuổi trung bình động  
tích hợp tự hồi quy) 52, 54-60 hàm

arima(), gói thống kê 36, 57

Mô hình ARMA 54-59 mỗi

quan hệ bất đối xứng 136, 139

lợi thế của cách tiếp cận điều kiện 142

tử cụ thể/chi tiết đến trừu tượng/tóm tắt 137  
ràng buộc và điều kiện 140 một

phản đến toàn bộ

136 cụ thể đến chung 137

các loại 135

Kiểm định Dickey-Fuller tăng cường. Xem  
hàm ADF auto.arima(), gói dự báo 36, tự tương  
quan 59 53

đồ thị hàm tự tương quan. Xem dự báo tự  
động ACF 51-52 phương pháp tự  
động 78 công cụ suy luận  
tự động 100 tự động hóa, trong bài  
đăng Reddit nghiên cứu trường hợp phân loại 128-  
130 đường trung

bình động tích hợp tự hồi quy.  
Xem trung

bình ARIMA, trọng số 164

## b

lan truyền ngược 78-83

túi từ tiếp cận 103-105 biểu đồ  
thanh

kiểm tra phân phối cho một biến 15-18 kiểm  
tra

mối quan hệ giữa hai biến 24-29

Mạng Bayesian 133 và

xác định phân phối xác suất 146-147 và  
mã hóa

các phụ thuộc có hướng 134 các bước cơ bản  
trong thiết kế 150 được xác  
định 144-146

thiết kế các cạnh

có hướng 150-155 144

hướng mũi tên trong cấu trúc

mạng 173 150-151

lý luận 147-149 và

chảy dọc theo con đường 147

giữa các nguyên nhân khác nhau của cùng một  
hiệu ứng 157-159

Mạng Bayesian, lý luận (tiếp theo)  
giữa các tác động khác nhau của cùng một  
nguyên nhân

157 con đường bị chặn và 147,  
con đường không bị chặn 149  
và 149 biến 150

hàm bds.test(), gói tseries 36 Bengio,

Yoshua 87 Máy

Boltzmann bị hạn chế Bernoulli.

Xem thuật

ngữ sai lệch

BRBM 77 bigram

104 phân phối hai chiều 11

túi từ được mã hóa nhị phân 103 số

nhi phân 104 thế

nhi phân 168 Nhị thức,

phản tử hỗn hợp 162 tham số băng thông

nhi phân 12 biểu đồ hai

phía 88 Hàm

Box.test(), gói thống kê 36, 58 BRBM

(Bernoulli bị hạn chế Boltzmann Máy) 87-89

xô 106-107

## C

---

mối quan hệ nhân quả 136, 151 và các  
phụ thuộc có hướng 135 định hướng  
137-138 Hàm cbind() 50 Thủ

CC 105 Thủ CD 105

Chuỗi, phần

tử Figaro và

thiết kế cấu trúc mạng

153 phụ thuộc có hướng và 138 biến con

135 độ chính xác phân loại 125 conda

cài đặt praw 112 cō

diều kiện phân phối xác suất.

Xem trường ngữ nhiên có

diều kiện CPD 175 cách tiếp cận điều kiện 159 độc  
lập có điều kiện, phụ thuộc vô

hướng và ma trận nhầm lẫn 141 123-

127, 130 Cách tiếp cận ràng buộc phản tử không đổi  
162, phụ

thuộc vô hướng và 140

mũi tên hội tụ, mạng Bayesian 148-149 lệnh coord\_flip  
28

CPD (phân phối xác suất có điều kiện) và sử  
dụng hàm đệ quy 162 ví dụ về thiết

ké mạng Bayesian 151-153 trên biến, mạng Bayesian

145 dữ liệu chéo 33

Cún, Yann Le 87

## D.

---

thành phần giảm chấn 51

mảng dữ liệu

71 thu thập dữ liệu 110

chuẩn bị và phân tích dữ liệu, tệp sổ tay IPython 110

data\_processing() function 117 danh

mục khoa học dữ liệu 118

phân loại cây quyết định 106-108

quyết địnhTreeData.json 110

học sâu 65-66 biểu đồ

mật độ 12-15 phụ thuộc

hướng thích

hợp, bất đồng về 138 khác biệt giữa trực tiếp và  
trực tiếp 143 gián tiếp 142 gây ra 159 mối quan

hệ giữa các

biến được dịch

thành 133 hàm diff() 35, 54 chính tả (đúng)  
khác nhau

102 không gian chiều 73

đồ thị tuần hoàn có hướng, mạng

Bayes 144-145 chu kỳ có

hướng 144 phụ thuộc có hướng 134-139 hình dạng phân phối

10 phân loại tài

liệu 108 ma trận thuật ngữ tài liệu

103-104, biểu đồ chấm 17

mô hình hàm mũ kép 45 mạng hạ

lưu 155

d-tách, tiêu chí trong mạng Bayesian 147

Thẻ DT 105

## E

---

easy\_install 109

cạnh 135

Chức năng EM (tối đa hóa kỳ vọng) 80

end(), gói thống kê 35, 38 mô hình dựa

trên năng lượng 89 entropy

107 dữ liệu

kiểm tra lỗi

kiểm tra phân phối cho biểu đồ thanh biến đơn  
15

biểu đồ mật độ 12

biểu đồ 11 kiểm

tra mối quan hệ giữa hai biến

biểu đồ thanh

24 biểu đồ hexbin

23 biểu đồ

đường 19 biểu đồ

phân tán 20 phạm vi

dữ liệu lệnh tóm tắt 6

dữ liệu kiểm tra lỗi, lệnh tóm tắt (tiếp theo) giá trị không hợp lệ 6 giá trị bị thiếu 5 ngoại lệ 6 tổng quan 3 đơn vị 7 sử dụng trực quan hóa 8 hàm `ets()` 35, 45, 51-52 bằng chứng quá trình 171 truy vấn với 155-156 Thẻ EX 105 hàm `exp()` 50 kỳ vọng và lý thuyết xác suất 164 của phân phối, tính toán 164 tối đa hóa kỳ vọng. Xem EM khám phá phân phối kiểm tra dữ liệu cho các biểu đồ thanh một biến số 15-18 biểu đồ mật độ 12-15 biểu đồ 11-12 kiểm tra mối quan hệ giữa hai biểu đồ thanh biến số 24-29 biểu đồ hexbin 23-24 biểu đồ đường 19-20 biểu đồ phân tán 20-23 trong nghiên cứu điển hình phân loại bài đăng Reddit 118-120 phạm vi dữ liệu lệnh tóm tắt 6-7 giá trị không hợp lệ 6 các giá trị bị thiếu 5-6 ngoại lệ 6 tổng quan 3-5 đơn vị 7-8 sử dụng trực quan hóa 8-10 mô hình dự báo hàm `mô` 45-52 Holt và Holt-Winters làm trơn theo cấp số nhân 48-50 làm trơn theo cấp số nhân đơn giản 46-48

**F**

`f1 score` 92 facetting graph 26 thừa số, lệnh tóm tắt 4 Đồi tương FeedForwardNetwork 83-84 Figaro và cơ chế suy luận 154 như một ngôn ngữ mô phỏng 166 phụ thuộc có hướng 138 phụ thuộc không có hướng 139-142 nhiều cách khác nhau để xác định CPD trong 151

biểu đồ thanh đầy 25 Phân tử lật 170 tệp forceGraph.html 110, 128 gói dự báo 46 hàm dự báo(), gói dự báo 35, 47, 50, 58 khuôn khổ dự báo 34, 51-52, riêng biệt cho các phụ thuộc có hướng và không có hướng 144 hàm `frequency()`, gói thống kê 35, 38 Đồi tương FullConnection 84 Thẻ FW 105

**g**

Mô hình hỗn hợp Gaussian 80 chức năng lọc từ chung 115 mô hình tổng quát 135 quy trình tổng quát 135 lớp geom 21 `ggplot2` 9-10 GLM (mô hình tuyến tính tổng quát) 80 Google 2012 paper by 87 và khai thác văn bản 98 Kính Google 65 Google Maps 101 giá trị thang độ xám 92

**h**

nơ-ron được tạo thủ công 67 hapaxes 118 hàm trợ giúp 162-163 biểu đồ hexbin 23-24 lớp ẩn 77, 83 nút ẩn 77, 88-89, 92 đơn vị ẩn 88, 92 biến ẩn 88-89 không gian nhiều chiều hơn 73

Hinton, Geoffrey 87-88 biểu đồ 118-119 kiểm tra phân phối cho biến đơn 11-12 được xác định 12 Holt hàm `mô` làm mịn 45, 48-50 `holt()`, dự báo gói 46 Làm mịn hàm `mô` Holt-Winters 45, 48-50 hàm `HoltWinters()` 35, 45 Huang, Jeubin 66 hàm `hw()`, gói dự báo 46 cầu hình hyperbol 79 siêu phẳng 74

IBM Watson 100  
nhập mã nltk 109  
Độc lập thẻ  
IN 105, chặn 156 chỉ báo (đo lường), biến 153 phân phối xác suất riêng lẻ 167 phụ thuộc gây ra 140, 148, 157 thu được thông tin 107 đầu vào ức chế 67-68 lớp đầu vào 75, 83  
biến đầu vào 74 biến  
tương tác 106-107  
biến trung gian 142-143 giá trị không hợp lệ 6

Tệp máy tính xách tay IPython 110, 115 lần lặp 81, 85

**J**

Thẻ JJ 105  
Thẻ JJR 105  
thẻ JJS 105

**K**

thuật toán k-means 80

**L**

hàm lag(), gói thống kê 35 độ trễ của chuỗi thời gian 52  
thuật toán ngôn ngữ 102 nơ ron ngoài cùng bên trái 78 từ vựng 105-106 khả năng 89  
biểu đồ đường 19-20 mô hình tuyến tính 68, 75 hàm liên kết 80 hàm hoàng thổ 21 khả năng log 89  
biểu đồ mật độ tỷ lệ logarit 14 khi nào sử dụng 15 đầu vào logic 67  
chức năng logistic 89 hồ sơ logistic 79 hồi quy logistic 90 dữ liệu theo chiều dọc 33 từ viết thường 105 chức năng lowess 21  
Thẻ LS 105

**M**

hàm ma(), gói dự báo 35, 39  
Mã hóa mạng Markov 133, 166  
và phụ thuộc vô hướng 134 được xác định 166-169

phân phối xác suất được xác định bởi suy luận 168-169 với 171-173 đại diện cho 170-171 các cạnh vô hướng trong 144, 170 so với mạng Bayesian 170 lệnh tối đa 4 McCulloch, Warren 67 Mô hình MCP 67 Nơ-ron MCP 68 Thẻ MD 105 có nghĩa là lỗi tuyệt đối 48

lỗi phần trăm tuyệt đối trung bình 48 lỗi tỷ lệ tuyệt đối trung bình 48 lệnh trung bình 4 lỗi trung bình 48 lỗi phần trăm trung bình 48 lệnh trung bình 4 lệnh tối thiểu 4 Minsky, Marvin 75 thiểu giá trị, kiểm tra dữ liệu bằng tóm tắt lệnh 5-6

Chức năng kích hoạt MLP (perceptron đa lớp) 75-86 79  
lan truyền ngược và 78-83 trong scikit-learning 83-85  
đã học 86  
thiết kế mô hình, ví dụ về mạng Bayesian và 159, 162-163 khung mô hình hóa 133 hệ thống giám sát, cho phương tiện truyền thông xã hội 108 hàm monthplot() 35, 44 phương pháp mostlikelyValue 172  
Truy vấn MPE (giải thích hợp lý nhất) 172  
Thuật toán MPEBeliefPropagation 172 phân phối đa phương thức 11

**N**

không gian n chiều 74 Bộ phân loại Naïve Bayes 106, 124 Mô hình Naïve Bayes 110, 125-126 NaiveBayesData.json 110 kỹ thuật nhận dạng thực thể có tên 98 phạm vi dữ liệu hẹp 7 Xử lý ngôn ngữ tự nhiên. Xem Bộ công cụ ngôn ngữ tự nhiên NLP. Xem hàm NLTK ndiffs() gói dự báo 36, 54 lớp phủ định 76

- cáu hình hàm mũ âm 79 dấu âm 83  
 cấu trúc mạng,  
 ví dụ về mạng thần kinh Bayesian 154
- đa lớp 75-86 chức  
 năng kích hoạt 79 lan  
 truyền ngược và 78-83 trong  
 scikit-learning 83-85  
 tri giác đa lớp đã học 86 tổng quan  
 66-67 tri giác  
 68-74 diễn giải hình  
 học của hai đầu vào 73-74 huấn luyện  
 69-73
- RBM (Máy Boltzmann bị hạn chế)  
 BRBM (Máy Boltzmann hạn chế Bernoulli) 88-  
 89 ví dụ minh họa  
 89 tổng quan 87-88 nơ-  
 ron 78 Ng,  
 Andrew 65
- NLP (Xử lý ngôn  
 ngữ tự nhiên) 98 NLTK (Bộ công cụ ngôn  
 ngữ tự nhiên) 109-110 Tập văn bản NLTK 115 Bộ  
 phân loại cây quyết  
 định NLTK 107 Gói NLTK 110 Mã thông  
 báo từ NLTK 117 lệnh  
 nltk.download() 109 nltk.org  
 109 Thẻ NN 105 Thẻ NNP 106 Thẻ  
 NNPS 106 Thẻ  
 NNS 106  
 chuẩn hóa
- tổ chức dữ liệu để phân tích 3  
 xác suất không chuẩn hóa và 169 hàm  
 nudge\_dataset 90  
 mảng NumPy 71
- 
- Ô**
- sắp xếp dữ liệu để phân tích 3  
 ngoại lệ 6  
 lôi đầu ra 82-83 lớp  
 đầu ra 75, 77, 82-83 nút  
 đầu ra 77 cấu  
 hình đầu ra 79 lắp  
 quá mức 88, 108
- 
- P**
- hàm pacf() 35, 53  
 Papert, Seymour 75  
 thông số 167
- biến cha 135 Phần  
 Speech Tagging. Xem Tự động tương quan một  
 phần gắn thẻ POS 53 đạo hàm  
 riêng 82 Thẻ PDT 106  
 thuật toán  
 học perceptron 70-71 perceptron 68-74
- diễn giải hình học của hai đầu vào 73-74 đa  
 lớp  
 75-86 chức năng  
 kích hoạt 79  
 lan truyền ngược và 78-83  
 trong scikit-learning  
 83-85 đã học 86  
 đào tạo 69-73  
 lệnh pip 109 pip  
 cài đặt praw 112  
 Pitts, Walter  
 67 hàm plot() 35, 59  
 đầu ra đồ thị 74  
 Phân phối Poisson 161  
 Gắn thẻ POS (Một phần của Gắn thẻ bằng giọng nói)  
 105-106 lớp dương  
 76 số thực dương, giá trị của một tiềm năng 167  
 giá trị dương 86  
 (các) thế giới có thể, mạng Bayes và 146 hàm  
 tiềm năng, tương tác với cấu trúc đồ thị 168
- Gói PRAW 110, 112-113 hàm  
 prawGetData() 115 hàm dự đoán  
 165 chuẩn bị dữ liệu,  
 trong trường hợp phân loại bài đăng Reddit  
 nghiên cứu 115-118, 120-  
 124 truy vấn xác suất trước 154  
 thành phần mô hình  
 xác suất của 138  
 diễn hình và các biến 142  
 lập trình xác suất, được sử dụng để mở rộng  
 mạng Bayes 159 suy luận  
 xác suất, hướng suy luận so với hướng phụ thuộc 135  
 ràng buộc phân phối xác suất và  
 141
- thu được tổng thể 142  
 không chuẩn hóa 169
- Thẻ PRP 106
- cắt tia cây quyết định 108, 127  
 punkt 109
- PyBrain 83-84, 86, 93
- lệnh python -m SimpleHTTPServer 8000  
 110
- Gói Python 110

**H**

hàm qqline() 58 hàm  
qqnorm() 58 tự định lượng  
65 hàm quantile() 4

**R**

thé RB 106  
RBM (Máy Boltzmann hạn chế) 87  
BRBM (Bernoulli hạn chế Boltzmann  
Máy) 88-89 ví dụ  
minh họa 89 tổng quan 87-88

Đường ống RBM/LR 91  
Thé RBR 106  
Thé RBS 106  
mẫu lý luận 144, 170  
Phân loại bài đăng trên Reddit nghiên cứu diễn hình  
108-130 phân tích dữ liệu  
124-128 khai phá dữ liệu 118-  
120 chuẩn bị dữ liệu 115-118, 120-124 truy  
xuat dữ liệu 112-115 tổng  
quan về quy trình khoa học dữ liệu 111  
Tổng quan về Bộ công cụ Ngôn ngữ Tự nhiên  
109-110 108  
thuyết trình và tự động hóa 128-130 nghiên  
cứu mục tiêu 111  
Thư viện API Reddit Python 112  
Reddit SQLite file 115  
trình mã thông báo biểu thức chính quy  
121 mối quan  
hệ giữa hai nguyên nhân có cùng tác động 140  
một phần-toàn bộ  
137 kiểm tra trực quan  
biểu đồ thanh 24-29  
biểu đồ hexbin 23-24  
biểu đồ đường 19-  
20 biểu đồ phân tán 20-  
23 toàn phần 137  
Máy Boltzmann bị hạn chế. Xem kết quả RBM, xem  
172 truy xuất dữ liệu,  
trong nghiên cứu tình huống phân loại bài đăng trên  
Reddit 112-115  
Hàm tạo RichCPD 152  
nơ ron ngoài cùng bên  
phải 78 hàm rollmean(), gói sở thú 39 lỗi  
binh phuong trung binh goc 48  
Rosenblatt, Frank 70  
Tâm thám thé  
RP 106, được xác định 26

**S**

biểu đồ phân tán 20-  
23 tài liệu scikit-learning 73, 88-89 hàm  
seasonplot(), gói dự báo 35, 44 mă thông báo câu 106  
phân tích tinh cảm 108 hàm  
ses(), gói dự báo 46 bộ  
tiềm năng, mạng Markov và 166 hàm  
setEvidence 171 hình dạng phân phối 10 làm mịn  
hàm mũ đơn giản 46-48 số hạng  
đơn giản 104 mô hình hàm mũ  
đơn 45 đơn vị ẩn đơn 92 siêu phẳng đơn 76-  
77 perceptron đơn 72 giá  
trị trọng số đơn 82 số hạng xuất  
hiện đơn lẻ 118

Hàm SMA(), gói TTR 39 đường cong làm  
mịn 21 quả cầu tuyết  
xuất phát 120 hệ thống giám  
sát mạng xã hội 108 phương thức sortModules()  
84 lỗi chính tả 102

Thuật giả kim SQL 112  
Tệp SQLite 112  
Gói SQLite3 110 cấu hình  
căn bậc hai 79 biểu đồ  
thanh xếp chồng 24 hàm  
start() 35, 38 lớp chỉ số  
21 góc 105-106  
hàm stl(), gói thống  
kê 35, 41-42 dừng lọc từ 105 từ dừng 109,  
116 subreddits mảng 115  
tóm tắt (), kiểm tra dữ  
liệu để tìm lỗi phạm vi  
dữ liệu 6-7 giá trị không hợp lệ 6

các giá trị bị thiếu  
5-6 ngoại  
lệ 6 tổng quan  
3-5 đơn vị  
7-8 Sunburst.html 110  
Thé SYM 106  
phụ thuộc đối xứng, tiềm năng và 167 mối quan hệ  
đối xứng 139

**T**

mảng mục tiêu 71  
thuật ngữ, một lần xuất hiện 118  
phân loại văn bản 103

khai thác và phân tích văn bản 96-130  
 tổng quan 96-97  
 các ứng dụng trong thế giới thực của 98-102  
 Phân loại bài đăng trên Reddit nghiên cứu điển hình 108-130 phân tích dữ liệu 124-128 khám phá dữ liệu 118-120 chuẩn bị dữ liệu 115-118, 120-124 truy xuất dữ liệu 112-115  
 tổng quan về quy trình khoa học dữ liệu 111  
 Bộ công cụ ngôn ngữ tự nhiên 109-110  
 trình bày và tự động hóa 128-130 mục tiêu nghiên cứu 111  
 kỹ thuật 103-108  
 phương pháp tiếp cận túi từ 103-105 phân loại cây quyết định 106  
 108 bắt nguồn và từ vựng 105-106  
 TF (Tần số kỳ hạn) 104  
 TF-IDF (Tài liệu Nghịch đảo Tần số Thuật ngữ Tần suất) 104  
 sai lệch ngưỡng 68  
 chuỗi thời gian 33-60  
 Các mô hình dự báo ARIMA 52-60  
 Các mô hình ARMA 54-60  
 tự tương quan 53 dự  
 báo ARIMA tự động 59-60 dự báo tự động 51-52 tạo các đối tượng chuỗi  
 thời gian 36-38 thành phần giảm chấn 51 phân biệt 53 đảm bảo tính ổn định 55-56 đánh giá mức độ phù hợp của mô hình 58 mô  
 hình dự báo hàm mũ 45-52 mô hình phù hợp 57- 58 chức năng để  
 phân tích 35  
 Holt và Holt-Winters làm mịn hàm mũ 48-50 xác định các  
 mô hình hợp lý 56-57 thành phần bất thường 40 độ trễ 52 đưa ra dự báo  
 58 đo lường độ chính xác dự báo 48 thành phần theo mùa  
 40 phân rã theo mùa 40-45  
 làm mịn hàm mũ đơn giản 46-48  
 làm mịn với trung bình động đơn giản 38-39 cố định và không cố định 53 thành phần xu hướng 40 đối tượng chuỗi thời gian 36-38 tokenization 104  
 tokenizer 117  
  
 topicID cột 113 phương thức train() 85  
 mạng thần kinh được đào tạo 84,  
 86 bất quá  
 104 mô hình hàm mũ ba lớp 45

hàm ts(), gói thống kê 35, 37 hai nguyên nhân của cùng một hiệu ứng đã biết, mối quan hệ đối xứng 140 mạng ẩn hai lớp 77 perceptron hai lớp 77

**bạn**

Thẻ UH 106  
 ràng buộc đơn nguyên 171 thẻ đơn nguyên 167 chu kỳ vô hướng 144 phụ thuộc vô hướng 134, 139-144 đồ thị vô hướng, mạng Markov và 166 unigram 104 phân phối đơn thức 10  
 đơn vị, kiểm tra dữ liệu bằng lệnh tóm tắt 7-8

**V**

giá trị trung bình và kỳ vọng 164 gấp đôi 164  
 kỳ vọng của 164 biến Boolean 160  
 kiểm tra phân phối trực quan biểu đồ thanh 15-18 biểu đồ mật độ 12-15 biểu đồ 11-12  
 tổng quan 10-11  
 độc lập có điều kiện và 147 kiểm soát 160 phụ  
 thuộc 134 tắt  
 định 153  
 lớp thừa số và lệnh tóm tắt 4 độc lập 134 số nguyên 160  
 trạng thái chung của các biến 167 giá trị chung 141  
 mẫu mũi tên thông thường và hội tụ 148 trực quan hóa cho một 18-19 trực quan hóa cho hai 29  
 lệnh phương sai 4  
 Thẻ VB 106  
 thẻ VBD 106  
 Thẻ VBG 106  
 thẻ VBN 106  
 Thẻ VBP 106  
 Thẻ VBZ 106  
 móc hiển thị 89  
 nút hiển thị 89  
 đơn vị hiển thị 88, 92

## trục quan hóa

kiểm tra phân phối cho biểu đồ thanh một  
biến 15-18 biểu  
đồ mật độ 12-15 biểu  
đồ 11-12 tổng quan  
10-11 kiểm tra  
mối quan hệ giữa biểu đồ thanh hai biến 24-29  
biểu đồ hexbin 23-  
24 biểu đồ đường 19-  
20 biểu đồ phân  
tán 20-23 tổng quan  
8-10

**W**

giá trị w\_0 69

Thẻ WDT trọng

số 106, gán cho các giá trị liên kết khác nhau  
141 tổng trọng số 69,  
tổng trọng số 82 67

hàm window(), gói thống kê 35, 38

Công cụ Wolfram Alpha tổng  
quan về lọc  
100 từ 115

dùng 105 từ,  
viết thường 105 từ mã  
thông báo 106, 117  
thẻ WP 106  
thẻ WP\$ 106  
Thẻ WRB 106

**X**


---

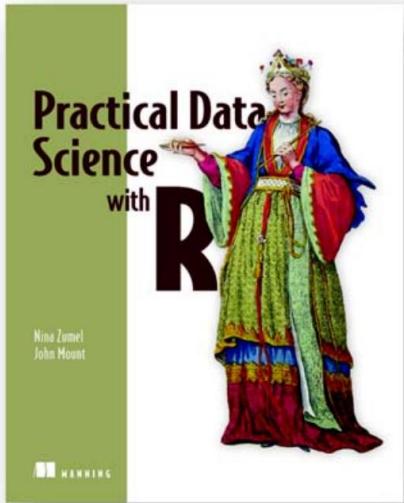
Hàm XOR 76-77, 80-81, 84, 86

**Z**


---

không, giá trị của một tiềm năng 167

Tiết kiệm 50% cho những cuốn sách đã chọn này—eBook, pBook và MEAP. Chỉ cần nhập nguồn cấp dữ liệu50 trong Hộp Mã Khuyến mại khi bạn thanh toán. Chỉ có tại [manning.com](http://manning.com).



Khoa học dữ liệu thực tế với R

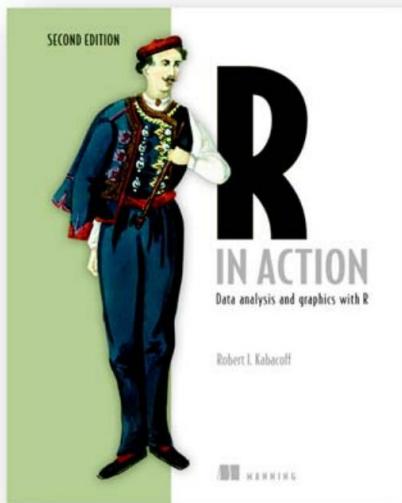
của Nina Zumel và John Mount

ISBN: 9781617291562 416

trang

\$49,99

Tháng 3 năm 2014



R trong hành động, Phiên bản thứ hai

Phân tích dữ liệu và đồ họa với R

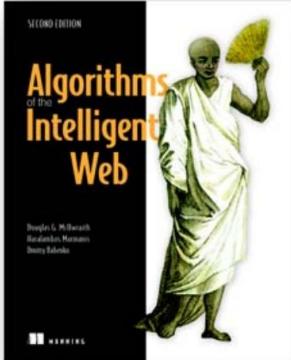
của Robert I. Kabacoff

ISBN: 9781617291388 608

trang

\$59,99

Tháng 5 năm 2015



[Các thuật toán của Web thông minh,  
Phiên bản thứ hai](#)

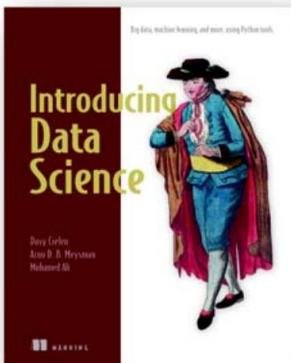
của Douglas G. McIlwraith, Haralambos Marmanis, và  
Dmitry Babenko

ISBN: 9781617292583

325 trang

\$44,99

Tháng 6 năm 2016



[Giới thiệu Khoa học dữ liệu  
Dữ liệu lớn, máy học và hơn thế nữa, sử  
dụng các công cụ Python](#)

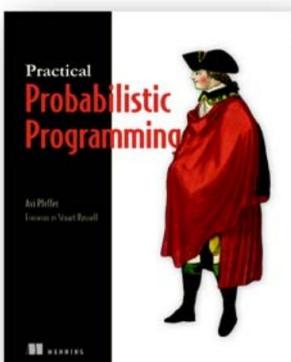
bởi Davy CieLEN, Arno DB Meysman,  
và Mohamed Ali

ISBN: 9781633430037

325 trang

\$44,99

Tháng 5 năm 2016



[Lập trình xác suất thực tế  
của Avi Pfeffer](#)

ISBN: 9781617292330

456 trang

\$59,99

Tháng 3 năm 2016