

Tài liệu đọc

Phân Tích Dữ Liệu

Khóa 7: Phân tích dữ liệu với lập trình R

Phần 1: Tài liệu đọc bổ trợ

<u>Bài đọc 1</u>	Thế giới lập trình
	1.1 Tìm hiểu và làm quen với R <ul style="list-style-type: none">- Câu hỏi thảo luận 1.2 Cuộc cạnh tranh giữa R và Python 1.3 Tìm hiểu tình huống: sử dụng Kaggle để đặt câu hỏi về R 1.4 Lập trình như nhà phân tích dữ liệu <ul style="list-style-type: none">- Ngôn ngữ lập trình và học lập trình- So sánh các loại bảng tính- Sử dụng R Console 1.5 Học lập trình với RStudio <ul style="list-style-type: none">- Khởi động với RStudio- Những tình huống sử dụng RStudio- Kết nối với cộng đồng R
<u>Bài đọc 2</u>	Lập trình với RStudio
	2.1 Khái niệm lập trình cơ bản <ul style="list-style-type: none">- Các khái niệm cơ bản trong R 2.2 Kiểu dữ liệu trong R 2.3 Khám phá lập trình trong R <ul style="list-style-type: none">- Toán tử và phép tính- Toán tử logic và câu lệnh điều kiện- Đảm bảo mã nguồn dễ đọc

	2.4 Tìm hiểu gói Tidyverse <ul style="list-style-type: none"> - Gói (package) trong R - Phân tích dữ liệu với Tidyverse - Làm việc với đường ống và mã lồng nhau - Tài nguyên hỗ trợ R
Bài đọc 3	Làm việc với dữ liệu trong R
	3.1 Khảo sát dữ liệu và R <ul style="list-style-type: none"> - Khung dữ liệu R: tạo khung dữ liệu và làm việc với khung dữ liệu - Tìm hiểu về tibbles 3.2 Quan sát dữ liệu <ul style="list-style-type: none"> - Làm sạch, tổ chức và biến đổi dữ liệu với R - Làm việc với dữ liệu thiên lệch
Bài đọc 4	Trực quan hóa, tính thẩm mỹ, và chú thích
	4.1 Tạo trực quan hóa dữ liệu trong R <ul style="list-style-type: none"> - Cơ bản về trực quan hóa trong R và tidyverse - Trực quan hóa dữ liệu với ggplot2() 4.2 Khảo sát tính thẩm mỹ trong phân tích <ul style="list-style-type: none"> - Tính thẩm mỹ: các thuộc tính và khía cạnh - Làm việc nhiều hơn với ggplot: các yếu tố của ggplot, các bộ lọc và đồ thị, và phép làm mịn 4.3 Chú thích và sao lưu phép trực quan hóa <ul style="list-style-type: none"> - Thêm các chú thích trong R - Sao lưu phép trực quan hóa của bạn
Bài đọc 5	Tài liệu và báo cáo
	5.1 Phát triển tài liệu và báo cáo trong RStudio <ul style="list-style-type: none"> - Tổng quan về R Markdown: R Markdown notebook, sử dụng R Markdown trong RStudio, và các tài nguyên về R Markdown - Đối chiếu R Markdown notebook với Jupyter notebook (optional) 5.2 Tạo tài liệu R Markdown

	<ul style="list-style-type: none">- Câu hỏi thảo luận <p>5.3 Hiểu biết đoạn mã và thao tác xuất tài liệu</p> <ul style="list-style-type: none">- Thêm đoạn mã vào R Markdown notebook- Kết xuất tài liệu R Markdown notebook
--	---

Phần 2: Hướng dẫn trả lời câu hỏi - Quiz

Phần 1

TÀI LIỆU ĐỌC BỔ TRỢ

Bài đọc 1: Thế giới lập trình

R là ngôn ngữ lập trình có thể hỗ trợ người học trong quy trình phân tích dữ liệu. Trong phần này của khóa học, chúng ta sẽ tìm hiểu R và RStudio, môi trường để làm việc với R. Chúng ta sẽ khám phá những lợi ích của việc sử dụng R và RStudio, cũng như bắt đầu việc phân tích cùng những thành phần của RStudio.

Mục tiêu học tập:

- Phân biệt môi trường lập trình R và môi trường lập trình RStudio.
- Mô tả môi trường lập trình RStudio, bao gồm các thành phần và lợi ích của công cụ.
- Mô tả ngôn ngữ lập trình R và môi trường lập trình của R.
- Mô tả các ngôn ngữ lập trình và cách sử dụng phù hợp, có ví dụ đi kèm.
- Tải xuống và cài đặt R vào máy tính.
- Mở R và thực thi một câu lệnh.
- Phân biệt giữa R Console và các môi trường lập trình R.
- Thực thi các thao tác trong R sử dụng toán tử toán học như +, -, *, và /.
- Tải xuống và sử dụng RStudio Desktop.
- Minh họa cách hoàn thành các tác vụ cơ bản trong R.

Lập trình máy tính (programming) là việc đưa ra các chỉ thị cho máy tính để thực hiện một hành động hoặc tập hợp các hành động theo cú pháp của một ngôn ngữ lập trình cụ thể. Việc chọn lựa ngôn ngữ lập trình tùy thuộc vào dự án bạn đang thực hiện hoặc vấn đề muốn giải quyết.

Ngôn ngữ lập trình (programming language) là hệ thống các từ và ký hiệu dùng để viết các chỉ thị mà máy tính theo đó thực hiện. *Cú pháp* (syntax) là bộ quy tắc riêng về cách sử dụng các từ và ký hiệu để chúng có ý nghĩa với máy tính. Ngôn ngữ lập trình là cầu nối cho phép con người và máy tính giao tiếp.

1. Tìm hiểu và làm quen với R

R là một ngôn ngữ lập trình được sử dụng cho phân tích thống kê, trực quan hóa và các tác vụ phân tích dữ liệu khác. Là một nhà phân tích dữ liệu, bạn sẽ sử dụng R để hoàn thành nhiều nhiệm vụ liên quan đến quá trình phân tích dữ liệu. Hiểu cách thức hoạt động và lý do bạn sử dụng nó là rất quan trọng để phát triển thành thạo kỹ năng phân tích dữ liệu.

Câu hỏi thảo luận 1

Viết 2-3 câu (tổng cộng 20-60 từ) để trả lời các câu hỏi sau:

- Nền tảng chuyên môn của bạn là gì?
- Điều gì khiến bạn đăng ký tham gia khóa học này?

Câu hỏi thảo luận 2

Tiếp theo, viết 3-5 câu (tổng cộng 60-100 từ) để chia sẻ suy nghĩ của bạn về nền tảng trong lập trình:

- Bạn đã từng làm việc với ngôn ngữ lập trình nào?
- Bạn đã sử dụng lập trình cho các dự án cá nhân hay chuyên nghiệp nào?
- Bạn thích điều gì nhất về lập trình?
- Nếu bạn là người mới học lập trình, bạn có cảm nghĩ gì về việc học lập trình bằng R: bạn thấy hào hứng, có chút lo lắng, hay cả hai?

Câu hỏi thảo luận 3

Viết 2-3 câu (40-60 từ) trả lời cho mỗi câu hỏi dưới đây:

- Điều gì khiến bạn quyết định tìm hiểu về R?
- Bạn hào hứng tìm hiểu những phần nào của R? Những phần nào có vẻ khó?

2. Cuộc cạnh tranh giữa R và Python

Mọi người thường tự hỏi họ nên học ngôn ngữ lập trình nào đầu tiên. Khóa học này tập trung vào R, do R là một điểm khởi đầu tuyệt vời để phân tích dữ liệu nền tảng và nó có các gói hữu ích để có thể áp dụng cho các dự án phân

tích dữ liệu. Tuy Python không có trong chương trình học, chúng tôi khuyến khích bạn khám phá Python sau khi hoàn thành khóa học.

Bảng sau đây so sánh R và Python. Nó tóm tắt các đặc điểm chung, cũng như ưu điểm và thử thách riêng của mỗi ngôn ngữ. Trên thực tế, hai ngôn ngữ này thường được sử dụng cùng nhau hơn là loại trừ nhau.

Ngôn ngữ	R	Python
Đặc điểm chung	<ul style="list-style-type: none"> Dữ liệu được lưu trữ trong khung dữ liệu (data frames) Có cộng đồng phát triển và hỗ trợ mã Mã nguồn mở Công thức và chức năng có sẵn 	
Ưu điểm riêng	<ul style="list-style-type: none"> Thao tác dữ liệu, trực quan hóa dữ liệu và gói thống kê Phương pháp tiếp cận "Scalpel" cho dữ liệu: tìm gói để thực hiện những gì bạn muốn với dữ liệu 	<ul style="list-style-type: none"> Cú pháp dễ dàng cho nhu cầu học máy Tích hợp với nền tảng đám mây như Google Cloud, Amazon Web Services và Azure
Thử thách riêng	<ul style="list-style-type: none"> Quy ước đặt tên không nhất quán → người mới bắt đầu khó chọn hàm Thao tác biến hơi phức tạp đối với người mới bắt đầu 	<ul style="list-style-type: none"> Người mới bắt đầu phải quyết định nhiều thứ như đầu vào / đầu ra dữ liệu, cấu trúc, biến, gói, và đối tượng Phương pháp tiếp cận "Swiss army knife" cho dữ liệu: tìm ra cách để thực hiện những gì bạn muốn với dữ liệu

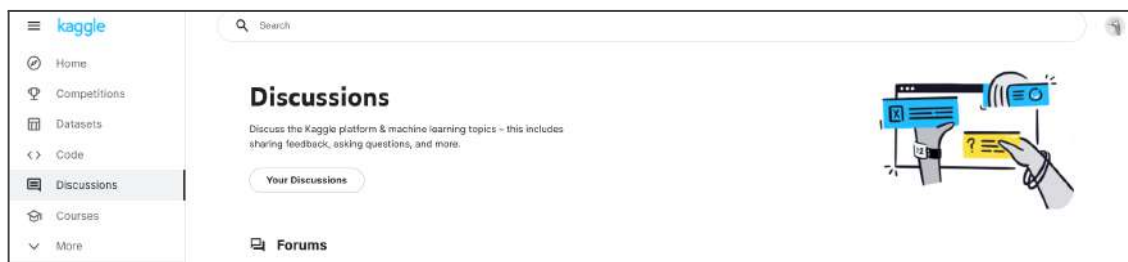
3. Tìm hiểu tình huống: Sử dụng Kaggle để đặt câu hỏi về R

Một trong những tài nguyên hữu ích nhất để học R là cộng đồng trực tuyến lớn gồm nhiều nhà phân tích dữ liệu. Lập trình viên R trên khắp thế giới hỗ trợ

n nhau trên diễn đàn trực tuyến bằng việc thảo luận và giải quyết các câu hỏi cú pháp và vấn đề về dữ liệu.

Trong tình huống này, bạn sẽ đăng trên diễn đàn nhờ ai đó trợ giúp trả lời một câu hỏi về R, có thể là về chiến lược học tập R hay cú pháp R. Hãy nhớ sử dụng chức năng tìm kiếm để kiểm tra xem câu hỏi của bạn đã được trả lời chưa, điều này giúp tránh được câu hỏi trùng lặp và do đó tiết kiệm thời gian cho mọi người.

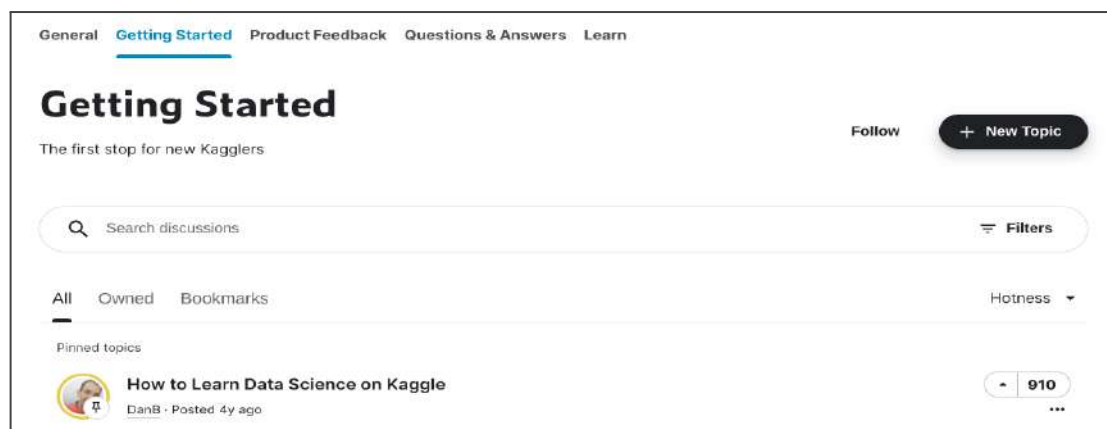
1. Đầu tiên, đăng nhập tài khoản Kaggle và điều hướng đến tab Thảo luận (Discussions) trên menu.



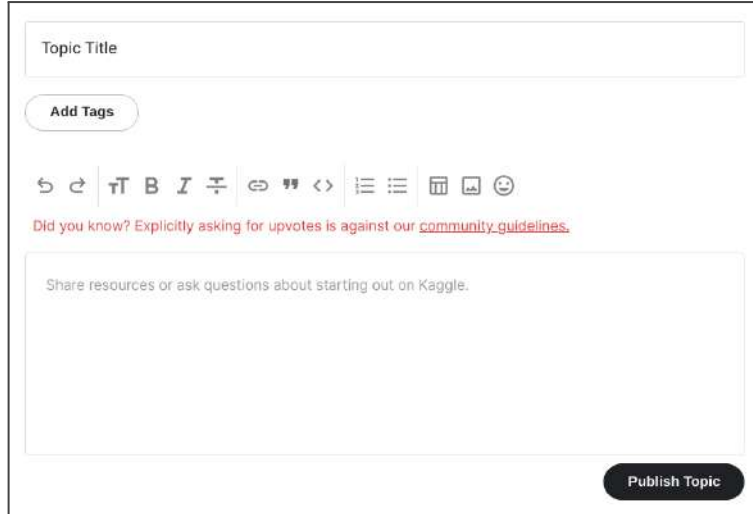
2. Chọn danh mục con **Bắt đầu (Getting Started)**.



3. Sau đó, nhấp nút **Chủ đề mới (New Topic)** để tạo chủ đề mới trên diễn đàn.



4. Viết **Tiêu đề chủ đề (Topic Title)** và nội dung câu hỏi, sau đó nhấn nút **Xuất bản chủ đề (Publish Topic)**.



Như vậy bạn đã đăng thành công một câu hỏi lên diễn đàn Kaggle. Kaggle và các diễn đàn phân tích dữ liệu khác có thể là nguồn tài nguyên hữu dụng khi bạn tìm hiểu về R và bất kỳ chủ đề phân tích dữ liệu nào khác.

4. Lập trình như nhà phân tích dữ liệu

Ngôn ngữ lập trình và học lập trình

Hãy cùng điểm qua một số ngành nghề tiềm năng và các ngôn ngữ lập trình phổ biến nhất được sử dụng trong những nghề nghiệp này.

Nhà phân tích dữ liệu: thu thập, biến đổi và sắp xếp dữ liệu để đưa ra kết luận hay dự đoán, và thúc đẩy việc ra quyết định sáng suốt.

- Ngôn ngữ phổ biến là R và Python.
- R cung cấp các tính năng thống kê thuận tiện để phân tích dữ liệu và hữu ích để tạo trực quan hóa dữ liệu nâng cao.
- Python là ngôn ngữ dùng chung cho nhiều tác vụ phân tích dữ liệu khác nhau.

Nhà thiết kế web: chịu trách nhiệm về kiểu dáng và bố cục của các trang web chứa văn bản, đồ họa và video.

- Ngôn ngữ phổ biến là HTML5 và CSS
- HTML5 cung cấp cấu trúc cho các trang web và được sử dụng để kết nối với các nền tảng lưu trữ.
- CSS được sử dụng để thiết kế trang web và kiểm soát các yếu tố đồ họa (màu sắc, bố cục và phông chữ) và trình bày trang trên nhiều thiết bị (màn hình lớn, màn hình di động và máy in).

Khi bắt đầu học một ngôn ngữ lập trình mới cần ghi nhớ một số điểm chính sau:

- Xác định một dự án thực tế và sử dụng ngôn ngữ để giúp bạn hoàn thành nó. Điều này làm cho quá trình học tập thực tế hơn và hấp dẫn hơn.
- Hãy ghi nhớ các khái niệm và quy tắc lập trình trước đây. Nhiều trong số này có thể chuyển giao giữa các ngôn ngữ lập trình. Vì vậy, sau khi bạn đã học một ngôn ngữ, việc học ngôn ngữ lập trình thứ hai hoặc thứ ba sẽ dễ dàng hơn.
- Tạo và lưu giữ các ghi chú hay và các trang tính ở bất kỳ định dạng nào (viết tay hoặc đánh máy) phù hợp nhất với bạn.
- Tạo một hệ thống tài liệu thông tin trực tuyến mà bạn có thể dễ dàng truy cập khi làm việc trong các môi trường lập trình khác nhau.

So sánh bảng tính (spreadsheet), SQL, và R

Mặc dù ngôn ngữ lập trình R có thể mới đối với bạn, nhưng nó có rất nhiều điểm tương đồng với các công cụ khác mà bạn đã khám phá trong chương trình này. Bây giờ, bạn sẽ so sánh bảng tính, SQL và R để hiểu rõ hơn về cách sử dụng từng loại công cụ này trong tương lai.

Là một nhà phân tích dữ liệu, rất có thể bạn sẽ làm việc với SQL, R và bảng tính vào một thời điểm nào đó. Mỗi công cụ đều có những điểm mạnh và điểm yếu riêng, nhưng chúng đều giúp quá trình phân tích dữ liệu trở nên trơn tru và hiệu quả hơn. Có hai điểm chung mà cả ba công cụ đều có:

- Sử dụng bộ lọc: Trong R, bạn có thể sử dụng chức năng bộ lọc. Điều này thực hiện tác vụ tương tự như một truy vấn SQL cơ bản SELECT WHERE. Trong bảng tính, bạn có thể tạo bộ lọc bằng cách sử dụng các tùy chọn menu.
- Sử dụng các hàm: Trong bảng tính, bạn sử dụng các hàm trong công thức. Trong SQL, bạn đưa chúng vào các truy vấn. Trong R, bạn sẽ sử dụng các hàm trong mã nguồn phân tích.

Bảng dưới đây trình bày các câu hỏi chính để khám phá các khía cạnh để so sánh ba công cụ này với nhau. Bạn có thể sử dụng nó như một hướng dẫn chung khi bắt đầu thao tác với R.

Câu hỏi chính	Bảng tính	SQL	R
Đây là gì?	Chương trình sử dụng cột và dòng để tổ chức dữ liệu, phân tích và thao tác dữ liệu với công thức, hàm và các chức năng hỗ trợ sẵn	Ngôn ngữ lập trình cơ sở dữ liệu để giao tiếp với cơ sở dữ liệu cho việc phân tích dữ liệu	Ngôn ngữ lập trình chung cho việc phân tích thống kê, trực quan hóa dữ liệu và những phân tích dữ liệu khác
Ưu thế chính là gì?	Gồm các công cụ và tính năng trực quan hóa	Cho phép người dùng thao tác và nhận diện dữ liệu cần cho phân tích	Cung cấp ngôn ngữ truy cập được để tổ chức, hiệu chỉnh, và làm sạch khung dữ liệu, và tạo trực quan hóa sâu sắc về dữ liệu
Loại tập dữ liệu nào là tốt nhất?	Tập dữ liệu nhỏ	Tập dữ liệu lớn	Tập dữ liệu lớn

Nguồn dữ liệu nào?	Nhập thủ công hoặc nhập từ nguồn ngoại vi	Truy cập từ một cơ sở dữ liệu ngoại vi	Được tải bằng R từ máy tính, hoặc từ nguồn ngoại vi
Dữ liệu phân tích thường được lưu ở đâu?	Trong tập tin bảng tính trên máy tính của bạn	Trong các bảng trên cơ sở dữ liệu đã truy cập	Trong tập tin R trên máy tính của bạn
Sử dụng công thức và hàm?	Có	Có	Có
Tạo trực quan hóa?	Có	Có, bằng công cụ hỗ trợ như hệ quản trị cơ sở dữ liệu (RDBMS) và công cụ trí tuệ kinh doanh (BI)	Có

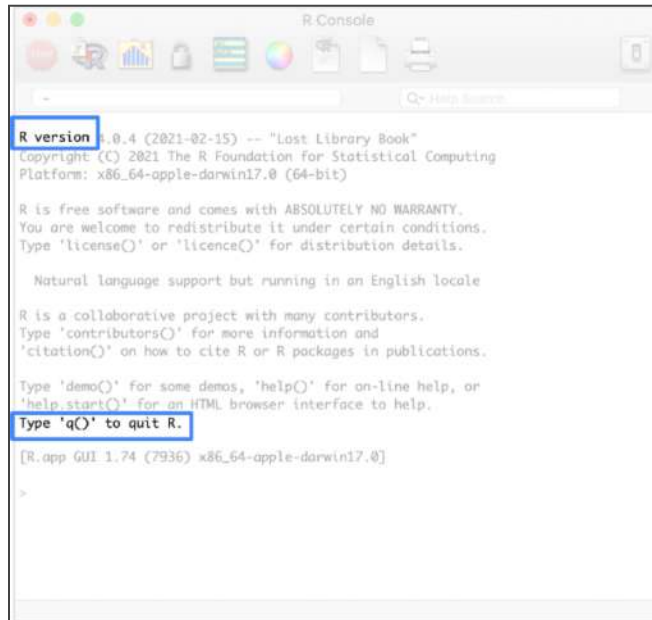
Sử dụng R Console

R Console là cửa sổ chương trình trong R để sử dụng ngôn ngữ lập trình R. Đây là giao diện cho phép bạn xem, viết, chỉnh sửa và thực thi mã R.

RStudio là môi trường phát triển tương tác (IDE) để lập trình R, sử dụng *R Console* và các công cụ khác để giúp viết và thực thi mã R dễ dàng hơn. Bạn có thể cài đặt *RStudio* trực tiếp trên máy tính (*RStudio Desktop*) hoặc sử dụng trên nền tảng đám mây (*RStudio Cloud*).

Trong *RStudio*, *R Console* thường được gọi là bảng điều khiển. Nó cho phép bạn thực hiện bất kỳ tác vụ nào trong *R Console*. Bạn sẽ thấy bảng hiển thị một thông báo mặc định. Thông báo bắt đầu bằng phiên bản R và số phiên bản, và kết thúc bằng Nhập 'q ()' để thoát R. Phía trên thông báo, bạn sẽ tìm thấy một

menu với các biểu tượng đại diện cho các chức năng của bảng console và giao diện người dùng đồ họa (được biết đến trong chương trình như RGui).



```
R version 4.0.4 (2021-02-15) -- "Last Library Book"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin17.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[R.app GUI 1.74 (7936) x86_64-apple-darwin17.0]
>
```

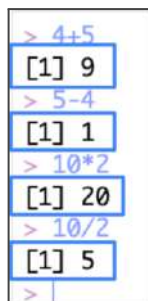
Đây là dòng lời nhắc và bất kỳ thứ gì bạn nhập sau nó sẽ được đọc dưới dạng mã R thực thi khi bạn nhấn Enter (Windows) hoặc Return (Mac).



```
[R.app GUI 1.74 (7936) x86_64-apple-darwin17.0]
> |
```

Hãy nhớ rằng mọi thứ bạn viết trong R Console sẽ biến mất sau khi bạn kết thúc phiên làm việc của mình (hoặc đóng bảng điều khiển). Nếu bạn muốn lưu đoạn mã đã thực thi, tốt hơn hết bạn nên lưu nó vào tập tin văn bản hoặc tập tin .rmd (bạn sẽ tìm hiểu thêm trong các bài học sắp tới).

Bây giờ hãy viết một phép toán. Bắt đầu với phép cộng đơn giản bằng cách sử dụng toán tử cộng (+). Nhập 4, sau đó là dấu cộng +, và cuối cùng là số 5. Như vậy, văn bản bạn vừa nhập giống như 4 + 5. Nhấn Enter (Windows) hoặc Return (Mac). R Console sẽ trả về câu trả lời cho câu hỏi này, đó là 9.



```
> 4+5
[1] 9
> 5-4
[1] 1
> 10*2
[1] 20
> 10/2
[1] 5
>
```

Câu hỏi thảo luận 1

Bạn đã tải xuống và cài đặt các tập tin cho ngôn ngữ lập trình R. Hãy viết 2-3 câu (40-60 từ) để trả lời cho mỗi câu hỏi sau.

- Ưu điểm của việc cài đặt R thay vì sử dụng nó trên nền tảng trực tuyến là gì?
- Học R sẽ giúp bạn xây dựng kỹ năng phân tích dữ liệu như thế nào?

Câu hỏi thảo luận 2

Bạn đã sử dụng R Console để viết một số chức năng cơ bản. Hãy viết 2-3 câu (40-60 từ) để trả lời cho mỗi câu hỏi sau.

- R Console cho bạn biết điều gì về lập trình trong giao diện R?
- Việc sử dụng R Console và việc viết mã R trong tập tin văn bản có gì khác biệt?

5. Học lập trình với RStudio

Bạn có thể tiếp cận RStudio thông qua RStudio Desktop hoặc RStudio Cloud.

RStudio Desktop là phiên bản mã nguồn mở theo giấy phép công cộng, được cài đặt trên máy tính. Bản dùng thử miễn phí *RStudio Pro* có tất cả các tính năng của phiên bản nguồn mở và giấy phép thương mại.

RStudio Cloud cho phép người dùng làm việc trực tuyến từ trình duyệt, và do đó không phụ thuộc vào hệ điều hành. Tuy nhiên, bạn cần đường truyền ổn định để có thể sử dụng *RStudio Cloud* thông suốt.

Khởi động với RStudio

Gói (package) bao gồm các đơn vị mã R có thể tái tạo, tài liệu mô tả, các bài kiểm tra mã nguồn và những tập dữ liệu mẫu. Cộng đồng R tạo ra các gói để quản lý các chức năng R mà họ viết và sử dụng lại.

Tidyverse là gói R với một triết lý thiết kế chung để thao tác, khám phá và trực quan hóa dữ liệu, và do đó là công cụ cần thiết đối với phân tích dữ liệu. Đây là gói cốt lõi ở trong R, và do đó bạn cần tìm hiểu làm sao có được gói *tidyverse* để sử dụng trong mỗi lần tương tác với R.

- Đầu tiên bạn cài đặt gói tidyverse từ console.

```

Console Terminal x Jobs x
R 4.0.3 · /cloud/project/

> install.packages("tidyverse")

* installing *binary* package 'tidyr' ...
* DONE (tidyr)
* installing *binary* package 'broom' ...
* DONE (broom)
* installing *binary* package 'modelr' ...
* DONE (modelr)
* installing *binary* package 'tidyverse' ...
* DONE (tidyverse)

The downloaded source packages are in
  '/tmp/RtmpEASzb7/downloaded_packages'

> |

```

- Sau đó, mỗi khi sử dụng, bạn gõ library(tidyverse) để nạp gói này vào chương trình hiện hành.

```

> library(tidyverse)
— Attaching packages — tidyverse 1.3.1 —
✓ ggplot2 3.3.5      ✓ purrr 0.3.4
✓ tibble 3.1.2       ✓ dplyr 1.0.7
✓ tidyr 1.1.3        ✓ stringr 1.4.0
✓ readr 1.4.0        ✓ forcats 0.5.1
— Conflicts — tidyverse_conflicts() —
x dplyr::filter() masks stats::filter()
x dplyr::lag() masks stats::lag()

```

Câu hỏi thảo luận

Bạn đã truy cập RStudio như một IDE để lập trình R. Hãy viết 2-3 câu (40-60 từ) để trả lời cho mỗi câu hỏi sau.

- Trải nghiệm sử dụng RStudio khác với các môi trường khác như chương trình R chuẩn như thế nào? (Nếu bạn không cài đặt R vào thiết bị của mình, so sánh các tính năng như thế nào?)
- Lợi thế của việc sử dụng RStudio Cloud là gì? Ngược lại, lợi thế của việc sử dụng RStudio Desktop là gì?
- Bạn có gặp những trở ngại nào hay không?

Những tình huống sử dụng RStudio

Là một nhà phân tích dữ liệu, bạn sẽ có rất nhiều công cụ để làm việc trong mỗi giai đoạn phân tích của mình. Đôi khi, bạn có thể đạt được các mục tiêu của mình bằng cách làm việc trong chương trình bảng tính hoặc sử dụng SQL với cơ sở dữ liệu. Bây giờ bạn sẽ xem qua một số ví dụ về thời điểm làm việc trong R và vì thế RStudio có thể là lựa chọn tốt hơn của bạn.

Một trong những nhiệm vụ cốt lõi của bạn với tư cách là nhà phân tích sẽ là chuyển đổi dữ liệu thô thành thông tin chi tiết chính xác, hữu ích và thú vị. Điều đó có thể khó thực hiện khi dữ liệu thô phức tạp. R và RStudio được thiết kế để xử lý các tập dữ liệu lớn mà các bảng tính có thể không xử lý được. RStudio cũng giúp bạn dễ dàng tái tạo công việc của mình trên các bộ dữ liệu khác nhau, đơn giản là chỉ cần tải một tập dữ liệu mới và chạy lại tập lệnh của bạn.

Bạn cũng có thể tạo hình ảnh trực quan chi tiết hơn bằng RStudio. Khi dữ liệu được trải rộng trên nhiều danh mục hoặc nhóm, việc quản lý phân tích, trực quan hóa xu hướng và xây dựng đồ họa có thể là một thách thức. Càng nhiều nhóm dữ liệu mà bạn cần làm việc, các nhiệm vụ đó càng trở nên khó khăn hơn.

Ví dụ, bạn đang phân tích dữ liệu bán hàng cho mọi thành phố trên toàn bộ quốc gia, có rất nhiều dữ liệu từ nhiều nhóm khác nhau – trong trường hợp này, mỗi thành phố có một nhóm dữ liệu riêng. Và đây là một số cách RStudio có thể trợ giúp bạn trong tình huống dữ liệu này:

- Sử dụng RStudio để thực hiện một bước phân tích cụ thể và thực hiện nó cho từng nhóm. Trong ví dụ này, bạn có thể tính toán dữ liệu bán hàng trung bình hàng năm cho mọi thành phố.
- RStudio cũng cho phép hiển thị dữ liệu linh hoạt. Bạn có thể hình dung sự khác biệt giữa các thành phố một cách hiệu quả bằng cách sử dụng các tính năng biểu đồ của RStudio như các hàm về facet.
- Bạn cũng có thể sử dụng RStudio để tự động tạo kết quả thống kê tóm tắt.

Câu hỏi thảo luận

RStudio cung cấp cho bạn một không gian làm việc duy nhất, nơi bạn có thể sử dụng R cho tất cả các giai đoạn của quá trình phân tích dữ liệu. Hãy viết một văn bản gồm hai hoặc nhiều đoạn văn (150–200 từ) mô tả những suy nghĩ ban đầu của bạn về RStudio.

- Bạn nghĩ RStudio có thể giúp bạn như thế nào trong vai trò nhà phân tích dữ liệu trong tương lai?
- Bạn yêu thích những tính năng nào của RStudio?
- Nếu bạn chưa quen với R và RStudio, bạn nghĩ tính năng nào sẽ hữu ích nhất cho bạn với tư cách là một người học?

Kết nối với cộng đồng R

R là một công cụ mạnh mẽ trong bộ công cụ phân tích dữ liệu của bạn. R có thể hơi khó học lúc đầu, nhưng may mắn là nó có một cộng đồng mạnh mẽ gồm những người dùng R, những người quan tâm đến việc làm việc cùng nhau và giúp đỡ lẫn nhau.

Dưới đây là một vài nơi để bạn có thể bắt đầu kết nối, trực tuyến và gặp trực tiếp, với các nhà phân tích khác trong cộng đồng R.

- Cộng đồng trực tuyến cho phép bạn kết nối với những người dùng R khác bất kể bạn sống ở đâu. Đó là các diễn đàn, kênh thảo luận, thẻ truyền thông xã hội (hashtag) trên nền tảng truyền thông xã hội.
- Nhiều tổ chức tổ chức cả buổi gặp mặt trực tiếp và trực tuyến cho người dùng R. Bạn cũng cần luôn thận trọng và an toàn khi trực tiếp tham dự các buổi gặp mặt.

Những tài nguyên này là một điểm khởi đầu tốt nếu bạn muốn bắt đầu kết nối với cộng đồng các nhà phân tích dữ liệu lớn hơn, vì vậy hãy tận dụng chúng!

Bài đọc 2: Lập trình với RStudio

Sử dụng R có thể giúp người học hoàn tất việc phân tích của họ một cách hiệu quả và năng suất. Trong phần này của khóa học, họ sẽ khám phá các khái niệm cơ bản liên quan đến R. Họ sẽ học về hàm và biến dùng cho các phép tính và lập trình. Ngoài ra, người học sẽ khám phá các gói R, đây là những tập hợp hàm R, mã nguồn, và dữ liệu mẫu mà họ sẽ sử dụng trong RStudio.

Mục tiêu học tập:

- Mô tả nội dung và các thành phần của gói tidyverse cho R.
- Mô tả khái niệm về gói trong ngôn ngữ lập trình R.
- Mô tả cách sử dụng toán tử để hoàn thành các phép tính trong ngôn ngữ lập trình R.
- Mô tả các khái niệm cơ bản liên quan đến lập trình R, bao gồm hàm, biến, kiểu dữ liệu, đường ống, và vector.
- Cài đặt và tải gói tidyverse.
- Sử dụng hàm `BrowseVignettes("packagename")` để đọc qua chi tiết các lần tải xuống của một gói đã tải.
- Định vị tài nguyên hỗ trợ bằng cách sử dụng R.

1. Khái niệm lập trình cơ bản

Chúng ta có một số khái niệm cơ bản trong R, bao gồm: hàm, biến, chú thích, kiểu dữ liệu, và đường ống. Tất cả chúng làm việc cùng nhau để làm nền tảng cho việc sử dụng R.

Các khái niệm cơ bản trong R

Hàm (function)

Phần mã có thể sử dụng lại để thực hiện các tác vụ cụ thể.

Hàm bắt đầu bằng tên hàm (ví dụ, `print` hay `paste`) và thường được theo sau bởi một hoặc nhiều đối số trong dấu ngoặc đơn. Đối số (argument) là thông tin mà một hàm R cần để chạy.

Ví dụ:

```
> print("Coding in R")
[1] "Coding in R"
>
```

```
print(x, ...)
```

print prints its argument and returns it *invisibly* (via `invisible(x)`). It is a generic function which means that new printing methods can be easily added for new classes.

Press F1 for additional help

print	{base}
print.AsIs	{base}
print.by	{base}
print.condition	{base}
print.connection	{base}
print.data.frame	{base}
print.Date	{base}
print.default	{base}

Biến (variable)

Biến là một đại diện của một giá trị trong R có thể được lưu trữ để sử dụng sau này trong quá trình lập trình. Biến cũng có thể được gọi là đối tượng. Tên biến thường là cụm từ ngắn có phân biệt hoa thường, bắt đầu bằng một chữ cái và cũng có thể chứa số và dấu gạch dưới.


Bạn có thể quan sát nội dung của các biến tại tab Global environment.


Environment

History


Connections


Tutorial





Import Dataset ▾

 137 MiB ▾



List ▾

R ▾

Global Environment ▾

Values

first_variable	"This is my variable"
vec 1	num [1:5] 13 48.5 71 101.5 2

Chú thích (comment)

Chú thích luôn bắt đầu bằng dấu # (hashtag), theo sau là những gì bạn cần mô tả hoặc giải thích về đoạn mã nguồn tương ứng. Bạn nên sử dụng chú thích nhiều nhất có thể để tập lệnh R dễ đọc hơn và mọi người đều có thể hiểu được mã nguồn.

Ví dụ:

```
> #Here is an example of a variable
> first_variable <- "This is my variable"
> first_variable
[1] "This is my variable"
```

Cấu trúc dữ liệu (data structures)

Cấu trúc dữ liệu là định dạng để tổ chức và lưu trữ dữ liệu. Các cấu trúc dữ liệu phổ biến nhất trong bao gồm: vectơ, khung dữ liệu (data frames), mảng và ma trận.

Chúng ta sẽ tìm hiểu kỹ hơn ở phần kế tiếp.

Đường ống (pipe)

Đường ống thể hiện một chuỗi gồm nhiều phép toán, được sử dụng để áp dụng đầu ra của một hàm vào một hàm khác. Sử dụng đường ống giúp mã nguồn dễ đọc và dễ hiểu hơn.

Ví dụ, đường ống này dùng để lọc và sắp xếp dữ liệu.

```
ToothGrowth %>%  
filter(dose == 0.5) %>%  
arrange(len)
```

2. Kiểu dữ liệu trong R

Vectơ (vector)

Một nhóm các phần tử dữ liệu cùng kiểu được lưu trữ thành chuỗi trong R.

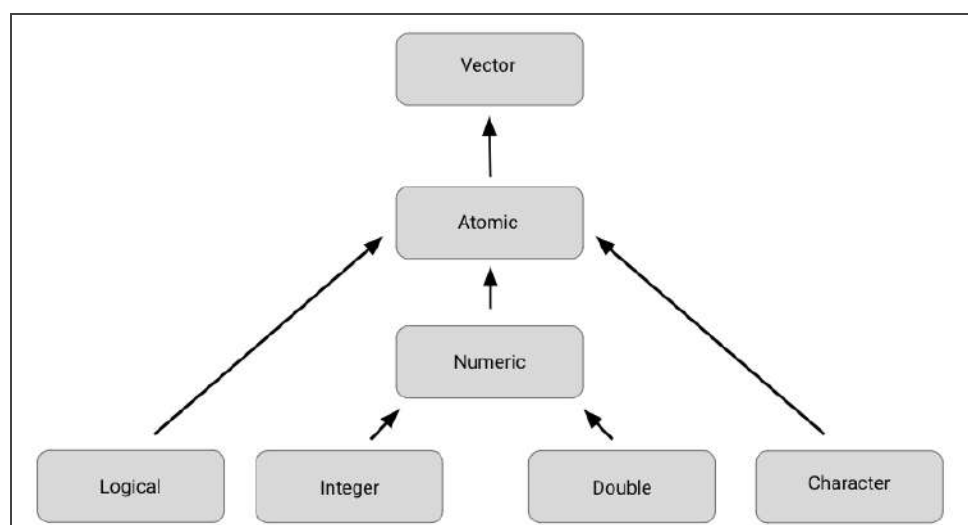
Ví dụ:

```
> vec_1 = c(13, 48.5, 71, 101.5, 2)  
> vec_1  
[1] 13.0 48.5 71.0 101.5 2.0
```

Có sáu loại vectơ nguyên tử chính: lôgic, số nguyên, kép, ký tự (chứa chuỗi), phức và thô. Hai loại cuối cùng không phổ biến trong phân tích dữ liệu, vì vậy chúng ta sẽ tập trung vào bốn loại đầu tiên như hiển thị trong bảng dưới đây.

Loại	Mô tả	Ví dụ
Logical	True/False	TRUE
Integer	Giá trị nguyên dương và âm	3
Double	Giá trị thực	101.175
Character	Giá trị chuỗi / ký tự	"Coding"

Các mối quan hệ giữa bốn loại vectơ chính được thể hiện bằng biểu đồ dưới đây:



Tiếp theo, chúng ta hãy xem xét một số thao tác cơ bản có thể thực hiện trên vectơ.

- Tạo vectơ bằng hàm `combine c()`.

```
c(2.5, 48.5, 101.5)
```

```
# Vectơ gồm số thực
```

<code>c("Sara", "Lisa", "Anna")</code>	# Vectơ gồm chuỗi
<code>c(TRUE, FALSE, TRUE)</code>	# Vectơ gồm giá trị Boolean

- Kiểm tra loại vectơ bằng hàm **typeof()** hoặc kiểm tra loại cụ thể bằng hàm **is**.
- Kiểm tra số phần tử trong vectơ bằng hàm **length()**.

```
x <- c(33.5,
57.75, 120.05)

typeof(x)

#> [1] "double"

is.double(x)

#> [1] TRUE

length(x)

#> [1] 3
```

- Tất cả loại vectơ đều có thể được đặt tên, điều này rất hữu ích để mã nguồn dễ đọc và mô tả các đối tượng rõ ràng. Bạn có thể đặt tên cho các phần tử của vectơ bằng hàm **names()**. Trong ví dụ bên dưới, ta gán biến x cho một vectơ mới có ba phần tử và sau đó gán một tên khác cho từng phần tử của vectơ; khi chạy mã kiểm tra nội dung biến x, R cho thấy rằng phần tử đầu tiên của vectơ được đặt tên là a, b thứ hai và c thứ ba.

```
> x <- c(1, 3, 5)
> names(x) <- c("a", "b", "c")
> x
a b c
1 3 5
```

Danh sách (list)

Danh sách khác với vectơ nguyên tử vì các phần tử của chúng có thể thuộc bất kỳ loại nào — như ngày tháng, khung dữ liệu, vectơ, ma trận, v.v. Danh sách thậm chí có thể chứa các danh sách khác.

Bạn có thể tạo danh sách bằng hàm **list()** và xem cấu trúc danh sách bằng hàm **str()**.

```
str(list("a", 1L, 1.5, TRUE))
#> List of 4
#> $ : chr "a"
#> $ : int 1
#> $ : num 1.5
#> $ : logi TRUE
```

```
list(list(list(1, 3, 5)))
#> List of 1
#> $ :List of 1
#> ..$ :List of 3
#> .. ..$ : num 1
#> .. ..$ : num 3
#> .. ..$ : num 5
```

Kiểu dữ liệu thời gian trong R

Bạn cần nạp cả hai gói, *tidyverse* và *lubridate* (thuộc gói *tidyverse*), để có thể làm việc với ngày và giờ trong R. R hỗ trợ ba loại dữ liệu để cập đến một thời điểm trong thời gian:

- Ngày tháng ("2016-08-16").
- Thời gian trong ngày ("20:11:59 UTC").
- Ngày tháng-thời gian ("2018-03-31 18:15:48 UTC").

Bạn có thể lấy ngày hiện tại bằng hàm **today()**, lấy ngày-giờ hiện tại bằng hàm **now()**, và sử dụng hàm **as_date()** để chuyển đổi ngày-giờ thành ngày tháng.

```
today()
#> [1] "2021-01-20"
```

```
now()
#> [1] "2021-01-20 16:25:05 UTC"
```

```
as_date(now())
#> [1] "2021-01-20"
```

Bạn có ba cách để tạo định dạng ngày-giờ trong R:

- Từ một chuỗi.
- Từ một ngày tháng đơn lẻ.
- Từ một đối tượng ngày / giờ hiện có.
- R tạo ngày ở định dạng yyyy-mm-dd chuẩn theo mặc định.

```
ymd("2021-01-20")
#> [1] "2021-01-20"
mdy_hm("01/20/2021 08:01")
#> [1] "2021-01-20 08:01:00 UTC"
ymd(20210120)
#> [1] "2021-01-20"
```

Khung dữ liệu (data frames)

Khung dữ liệu là tập hợp các cột – tương tự như bảng tính hoặc bảng SQL – giúp tóm tắt và sắp xếp dữ liệu ở định dạng dễ đọc và sử dụng. Mỗi cột có một tên ở trên cùng đại diện cho một biến và bao gồm một quan sát trên mỗi hàng.

Ví dụ:

```
> data.frame(x = c(1, 2, 3) , y = c(1.5, 5.5, 7.5))
  x  y
1 1 1.5
2 2 5.5
3 3 7.5
```


	carat	cut	color	clarity	depth	table	price	x	y	z
1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
5	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75
6	0.24	Very Good	J	VVS2	62.8	57.0	336	3.94	3.96	2.48
7	0.24	Very Good	I	VVS1	62.3	57.0	336	3.95	3.98	2.47
8	0.26	Very Good	H	SI1	61.9	55.0	337	4.07	4.11	2.53
9	0.22	Fair	E	VS2	65.1	61.0	337	3.87	3.78	2.49
10	0.23	Very Good	H	VS1	59.4	61.0	338	4.00	4.05	2.39
11	0.30	Good	J	SI1	64.0	55.0	339	4.25	4.28	2.73
12	0.23	Ideal	J	VS1	62.8	56.0	340	3.93	3.90	2.46
13	0.22	Premium	F	SI1	60.4	61.0	342	3.88	3.84	2.33
14	0.31	Ideal	J	SI2	62.2	54.0	344	4.35	4.37	2.71
15	0.20	Premium	E	SI2	60.2	62.0	345	3.79	3.75	2.27
16	0.32	Premium	E	I1	60.9	58.0	345	4.38	4.42	2.68
17	0.30	Ideal	I	SI2	62.0	54.0	348	4.31	4.34	2.68
18	0.30	Good	J	SI1	63.4	54.0	351	4.23	4.29	2.70
19	0.30	Good	J	SI1	63.8	56.0	351	4.23	4.26	2.71
20	0.30	Very Good	J	SI1	62.7	59.0	351	4.21	4.27	2.66
21	0.30	Good	I	SI2	63.3	56.0	351	4.26	4.30	2.71

Showing 1 to 22 of 53,940 entries, 10 total columns

Tập tin trong R

Bạn có thể thao tác với tập tin và thư mục trong R như trong bất kỳ công cụ nào khác. Sau đây là một số thao tác cơ bản thường thấy:

- Sử dụng hàm **dir.create()** để tạo một thư mục mới để chứa các tập tin của bạn.

```
dir.create("destination_folder")
```

- Sử dụng hàm **file.create()** để tạo một tập tin trống. Đặt tên và loại tập tin trong dấu ngoặc đơn của hàm. Các loại tập tin thường gặp là .txt, .docx hoặc .csv. Nếu tập tin được tạo thành công, R sẽ trả về giá trị TRUE (nếu không, trả về FALSE).

```
file.create("new_csv_file.csv")
```

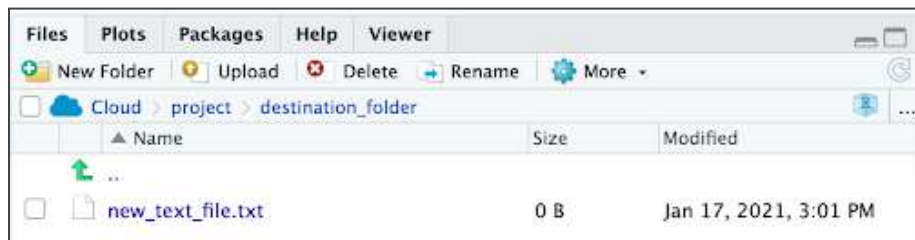
- Việc sao chép tập tin có thể được thực hiện bằng hàm **file.copy()**. Trong dấu ngoặc đơn, bạn chỉ định tên tập tin được sao chép, tiếp theo nhập dấu phẩy, và theo sau là tên của thư mục đích mà bạn muốn sao chép tập tin vào.

```
file.copy("new_text_file.csv", "destination_folder")
```

- Bạn có thể xóa các tập tin bằng cách sử dụng hàm **unlink()**. Nhập tên tập tin trong dấu ngoặc đơn của hàm.

```
unlink("new_text_file.csv")
```

Bạn có thể quan sát các tập tin trong R thông qua tab Tập tin trong RStudio.



Ma trận (matrix)

Ma trận là một tập hợp hai chiều của các phần tử dữ liệu. Điều này có nghĩa là nó có cả hàng và cột. Trong khi đó, vectơ là một chuỗi các phần tử dữ liệu một chiều. Tuy nhiên, giống như vectơ, ma trận chỉ có thể chứa một kiểu dữ liệu duy nhất, tức là bạn không thể có cả số logic và số trong một ma trận.

Để tạo ma trận trong R, bạn có thể sử dụng hàm **matrix()**. Hàm này có hai đối số chính: một vectơ chứa các giá trị bạn muốn đặt trong ma trận, và ít nhất một chiều của ma trận. Bạn có thể chọn chỉ định số hàng hoặc số cột bằng cách sử dụng mã `nrow =` hoặc `ncol =`.

Ví dụ:

```
> matrix(c(3:8), nrow = 2)
      [,1] [,2] [,3]
[1,]    3    5    7
[2,]    4    6    8
```

```
> matrix(c(3:8), ncol = 2)
      [,1] [,2]
[1,]    3    6
[2,]    4    7
[3,]    5    8
```

3. Khám phá lập trình trong R

Toán tử và phép tính

Toán tử (operator) là một ký hiệu đặt tên cho kiểu thao tác hoặc phép tính sẽ được thực hiện trong một công thức.

Toán tử để thực hiện các phép tính bao gồm:

- Toán tử gán (assignment operator) gán giá trị cho các biến và vectơ. Ví dụ, `x <- c(33.5, 57.75, 120.05)`.
- Toán tử số học (arithmetic operator) để hoàn thành các phép tính toán học quen thuộc: + (cộng), - (trừ), * (nhân), và / (chia).

Toán tử logic và câu lệnh điều kiện

Toán tử logic (logical operator)

Toán tử logic trả về kiểu dữ liệu logic như TRUE hay FALSE. Có ba loại toán tử logic chính: AND (ký hiệu & hoặc &&), OR (ký hiệu | hoặc ||) và NOT (ký hiệu !). Hoạt động của những toán tử này hoàn toàn giống như toán tử logic trong toán học.

Ví dụ, bạn đang làm việc với tập dữ liệu chất lượng không khí được tải trước trong RStudio. Nó chứa dữ liệu về các phép đo chất lượng không khí hàng ngày ở New York từ tháng 5 đến tháng 9 năm 1973. Khung dữ liệu có sáu cột: Ozone (đo ôzôn), Solar.R (đo mặt trời), Wind (đo gió), Temp (nhiệt độ tính bằng Fahrenheit), Tháng và Ngày của các phép đo này.

	Ozone	Solar.R	Wind	Temp	Month	Day
1	41	190	7.4	67	5	1
2	36	118	8.0	72	5	2
3	12	149	12.6	74	5	3
4	18	313	11.5	62	5	4

- Bạn muốn chỉ định các hàng biểu diễn những ngày cực kỳ nắng gió, được xác định bằng tiêu chí: số đo Solar.R trên 150 VÀ số đo Wind trên 10. Kết quả trả về là những dòng thỏa cả hai điều kiện.

Solar.R > 150 & Wind > 10

	Ozone	Solar.R	Wind	Temp	Month	Day
1	18	313	11.5	62	5	4

- Tiếp theo, bạn muốn chỉ định các hàng ở đó cực kỳ nắng hoặc cực nhiều gió, được xác định bằng tiêu chí: số đo Solar.R trên 150 HOẶC số đo Wind trên 10. Kết quả trả về những dòng thỏa một trong hai điều kiện.

Solar.R > 150 | Wind > 10

	Ozone	Solar.R	Wind	Temp	Month	Day
1	41	190	7.4	67	5	1
2	12	149	12.6	74	5	3
3	18	313	11.5	62	5	4

- Cuối cùng, bạn muốn tập trung vào các tình huống không quá nắng VÀ không quá nhiều gió, dựa trên các tiêu chí phía trên về cực kỳ nắng và cực kỳ nhiều gió. Nói cách khác, phát biểu sau đây không được đúng: số đo Solar.R lớn hơn 150 hoặc số đo Wind lớn hơn 10.

!(Solar.R > 150 | Wind > 10)

	Ozone	Solar.R	Wind	Temp	Month	Day
1	36	118	8.0	72	5	2

Phát biểu điều kiện (conditional statement)

Phát biểu điều kiện là câu khai báo rằng nếu một điều kiện nhất định đúng, thì một sự kiện nhất định phải diễn ra. Ví dụ, "Nếu nhiệt độ trên mức đóng băng, thì tôi sẽ ra ngoài đi dạo.", nếu điều kiện đầu tiên là đúng (nhiệt độ trên mức đóng băng), thì điều kiện thứ hai sẽ xảy ra (tôi sẽ đi dạo). Các câu lệnh điều kiện trong mã R có một logic tương tự.

Câu lệnh **if** trong R đặt một điều kiện. Nếu điều kiện cho giá trị là TRUE thì mã R liên kết với câu lệnh if sẽ được thực thi. Điều kiện được chỉ định bên trong dấu ngoặc đơn của câu lệnh if. Mã cần thực thi nếu điều kiện là TRUE được đặt trong dấu ngoặc nhọn { expr }.

Cú pháp	Ví dụ
<code>if (condition) {expr}</code>	<code>if (x > 0) {print("x is a positive number")}</code>

Câu lệnh **else** được sử dụng kết hợp với câu lệnh **if**.

Cú pháp	Ví dụ
<code>if (condition) {expr 1} else {expr 2}</code>	<code>if (x > 0) {print("x is a positive number")} else { print(" x is a non-positive number")}</code>

Trong một số trường hợp, bạn có thể muốn tùy chỉnh thêm câu lệnh điều kiện của mình bằng cách thêm câu lệnh **else if**. Câu lệnh **else if** nằm giữa câu lệnh **if** và câu lệnh **else**.

Cú pháp	Ví dụ
<code>if (condition1) {expr1} else if (condition2) {expr2} else {expr3}</code>	<code>if (x > 0) {print("x is a positive number")} else if (x < 0){print("x is negative number")} else {print("x is zero")}</code>

Câu hỏi thảo luận

Bây giờ bạn đã viết các truy vấn bằng SQL và sử dụng mã để lập trình trong R, bạn có thể nhận thấy một số điểm tương đồng giữa hai điều này. Hãy viết một thảo luận gồm hai hoặc nhiều đoạn văn (tổng cộng 100-150 từ) về bất kỳ điểm tương đồng nào mà bạn có thể gặp phải.

Đảm bảo mã nguồn dễ đọc

Khi viết mã R (hoặc bất kỳ ngôn ngữ lập trình nào khác), điều quan trọng là sử dụng một phong cách rõ ràng và nhất quán, không có lỗi. Điều này giúp làm cho mã của bạn dễ đọc và dễ hiểu hơn. Bạn cần nắm một số quy tắc áp dụng để làm theo khi viết mã R và nắm một số mẹo để xác định và sửa lỗi trong mã R.

Sử dụng phong cách viết mã rõ ràng và nhất quán thường làm cho mã của bạn dễ đọc hơn cho người khác. Không có kiểu mã hóa chính thức đối với tất cả người dùng R nhưng tồn tại những quy tắc bất thành văn về mã nguồn trong cộng đồng R.

Có hai lý do chính để sử dụng một phong cách mã hóa nhất quán:

- Nếu bạn đang làm việc với cộng tác viên hoặc đồng đội, việc sử dụng phong cách nhất quán là điều quan trọng để mọi người có thể dễ dàng đọc, chia sẻ, chỉnh sửa và làm việc trên mã của nhau.
- Nếu bạn đang làm việc một mình, việc sử dụng một phong cách nhất quán là rất quan trọng vì nó giúp việc xem xét mã nguồn trở nên dễ dàng và nhanh chóng hơn nhiều, sau đó là hiệu chỉnh hoặc sửa lỗi.
- Sửa lỗi mã R bắt đầu bằng việc chẩn đoán chính xác sự cố. Bước đầu tiên để chẩn đoán sự cố trong mã của bạn là hiểu những gì bạn mong đợi sẽ xảy ra. Sau đó, bạn có thể xác định điều gì đã thực sự xảy ra và nó khác với mong đợi của bạn như thế nào.

4. Tìm hiểu gói tidyverse

Gói (package) trong R

Gói trong R bao gồm các chức năng R có thể tái sử dụng và tài liệu về các chức năng, bao gồm cả cách sử dụng chúng. Chúng cũng chứa các bộ dữ liệu mẫu và các bài kiểm tra để kiểm tra mã nguồn.

Theo mặc định, R bao gồm một tập hợp các gói được gọi là Base R có sẵn để sử dụng trong RStudio. Cũng có các gói khuyến nghị tải về nhưng không được cài đặt sẵn. Trước khi sử dụng các hàm từ một trong các gói này, bạn phải tải nó bằng lệnh thư viện.

Để kiểm tra danh sách gói đã cài đặt, bạn chạy lệnh **install.packages()**. Cột Priority cho biết những gì cần thiết để sử dụng các chức năng từ gói. Nếu bạn bắt gặp từ "base" trong cột này thì gói đó đã được cài đặt và tải, và do đó bạn có thể sử dụng tất cả các chức năng của gói đó ngay khi mở RStudio. Nếu bạn tìm thấy từ "recommended", thì gói đã được cài đặt nhưng chưa được tải, và do đó bạn cần phải tải nó trước khi sử dụng trong phiên làm việc.

```
> installed.packages()
```

	Package	LibPath	Version	Priority
base	"base"	"/opt/R/4.2.1/lib/R/library"	"4.2.1"	"base"
boot	"boot"	"/opt/R/4.2.1/lib/R/library"	"1.3-28"	"recommended"
class	"class"	"/opt/R/4.2.1/lib/R/library"	"7.3-20"	"recommended"
cluster	"cluster"	"/opt/R/4.2.1/lib/R/library"	"2.1.3"	"recommended"
codetools	"codetools"	"/opt/R/4.2.1/lib/R/library"	"0.2-18"	"recommended"
compiler	"compiler"	"/opt/R/4.2.1/lib/R/library"	"4.2.1"	"base"
datasets	"datasets"	"/opt/R/4.2.1/lib/R/library"	"4.2.1"	"base"
foreign	"foreign"	"/opt/R/4.2.1/lib/R/library"	"0.8-82"	"recommended"
graphics	"graphics"	"/opt/R/4.2.1/lib/R/library"	"4.2.1"	"base"
grDevices	"grDevices"	"/opt/R/4.2.1/lib/R/library"	"4.2.1"	"base"
grid	"grid"	"/opt/R/4.2.1/lib/R/library"	"4.2.1"	"base"

Với đa dạng gói hỗ trợ từ cộng đồng R, thật khó để biết gói nào sẽ hữu ích nhất cho bạn. Sau đây là một số tài nguyên thông tin để bạn tham chiếu:

- *Tidyverse*: là một tập hợp các gói R được thiết kế đặc biệt để làm việc với dữ liệu. Đây là thư viện tiêu chuẩn cho hầu hết các nhà phân tích dữ liệu, nhưng bạn cũng có thể tải xuống các gói riêng lẻ.
- *Danh sách nhanh các gói R hữu ích (Quick list of useful R packages)*: đây là danh sách các gói hữu ích của Bộ phận Hỗ trợ RStudio với hướng dẫn cài đặt và mô tả chức năng.
- *CRAN Task Views*: đây là chỉ mục của các gói CRAN được sắp xếp theo tác vụ. Bạn có thể tìm kiếm loại nhiệm vụ bạn cần thực hiện và nó sẽ kéo lên một trang với các gói liên quan đến nhiệm vụ đó để bạn khám phá.

Phân tích dữ liệu với tidyverse

Tidyverse là tập hợp các gói trong R với một triết lý thiết kế chung cho thao tác, khám phá và trực quan hóa dữ liệu. Sử dụng gói này có thể giúp bạn làm việc trong toàn bộ quá trình phân tích dữ liệu. Các gói trong tidyverse hoạt động với nhau một cách tự nhiên.

Bạn cần phải tải tidyverse và nạp (load) gói này mỗi khi sử dụng. Kết quả trả về cho thấy có tám gói được tải: ggplot2, tibble, slimr, readr, purrr, dplyr, stringr và forcats. Các gói này là cốt lõi của tidyverse vì bạn sẽ sử dụng chúng trong hầu hết mọi phân tích. Ngoài ra, đi kèm là một danh sách các *xung đột*

(conflicts), là những gì xảy ra khi các gói có các hàm trùng tên với các hàm khác.

```
install.packages("tidyverse")
```

```
library("tidyverse")
```

Attaching packages			
✓	ggplot2	3.3.6	✓ purrr 0.3.4
✓	tibble	3.1.7	✓ dplyr 1.0.9
✓	tidyr	1.2.0	✓ stringr 1.4.0
✓	readr	2.1.2	✓ forcats 0.5.1
Conflicts			
✗	dplyr::filter()	masks	stats::filter()
✗	dplyr::lag()	masks	stats::lag()

Bốn trong số tám gói đề cập ở trên có vai trò thiết yếu với quy trình làm việc của nhà phân tích dữ liệu: ggplot2, dplyr, slimr và readr. Rất có thể bạn sẽ sử dụng những gói này thường xuyên hơn những cái khác.

- *ggplot2* được sử dụng để trực quan hóa dữ liệu, cụ thể là các biểu đồ. Với ggplot2, bạn có thể tạo nhiều loại dữ liệu khác nhau bằng cách áp dụng các thuộc tính trực quan khác nhau cho các biến dữ liệu.
- *tidyr* gói được sử dụng để dọn dẹp dữ liệu để làm cho dữ liệu gọn gàng hơn.
- *readr* được sử dụng để nhập dữ liệu. Hàm phổ biến nhất từ readr là `read_csv` để thao tác này sẽ nạp tập tin CSV vào R.
- *dplyr* cung cấp một bộ chức năng nhất quán giúp bạn hoàn thành một số tác vụ thao tác dữ liệu thông thường. Ví dụ: hàm `select` chọn các biến dựa trên tên của chúng và hàm `filter` tìm các trường hợp các điều kiện nhất định là đúng.

Làm việc với đường ống và mã lồng nhau

Xin nhắc lại nhanh, một đường ống là một công cụ trong R để thể hiện một chuỗi gồm nhiều phép toán. Nói cách khác, nó lấy đầu ra của một câu lệnh và

biến nó thành đầu vào của câu lệnh tiếp theo. Thay vì gõ các hàm có bên trong các hàm khác, bạn có thể sử dụng toán tử ống để thực hiện công việc tương tự. Trong lập trình, chúng ta gọi điều này là lồng nhau.

Lồng nhau (nested) mô tả mã thực hiện một chức năng cụ thể và được chứa trong mã thực hiện một chức năng rộng hơn. Bạn có thể xem đường ống như một cách để triển khai lồng nhau.

Ví dụ, các đoạn mã dưới đây thực hiện cùng tác vụ nhưng theo ba kịch bản khác nhau: không lồng nhau, lồng nhau, và dùng đường ống.

Không lồng nhau	<pre>filtered_tg <- filter(ToothGrowth, dose = 0.5) arrange(filtered_tg, len)</pre>
Lồng nhau	<pre>arrange(filter(ToothGrowth, dose = 0.5), len)</pre>
Đường ống	<pre>filtered_toothgrowth <- ToothGrowth %>% filter(ToothGrowth, dose = 0.5) %>% arrange(len)</pre>

Tài nguyên hỗ trợ R

Cộng đồng R có đầy đủ những người dùng tận tâm giúp nhau tìm ra giải pháp cho các vấn đề và cách sử dụng R. Ngoài ra còn có rất nhiều blog tuyệt vời, nơi bạn có thể tìm thấy các hướng dẫn và các tài nguyên khác. Dưới đây là một vài lựa chọn trong số đó.

- RStudio
- RStudio Blog
- Stack Overflow
- R-Bloggers
- R-Bloggers' tutorials for learning R

Bài đọc 3: Làm việc với dữ liệu trong R

Ngôn ngữ lập trình R được thiết kế để làm việc với dữ liệu ở tất cả giai đoạn của quy trình phân tích dữ liệu. Trong phần này của khóa học, người học khảo sát cách R có thể giúp họ cấu trúc, tổ chức, và làm sạch dữ liệu bằng cách sử dụng các hàm và tiến trình khác. Họ sẽ tìm hiểu về khung dữ liệu (data frames) và cách làm việc với chúng trong R. Họ cũng sẽ xem xét lại vấn đề dữ liệu thiên lệch và cách R hỗ trợ người dùng trong vấn đề này.

Mục tiêu học tập:

- Thảo luận cách sử dụng hàm R để giải quyết các vấn đề về thiên lệch và mối quan hệ giữa các biến dữ liệu.
- Mô tả các hàm R có thể được sử dụng để làm sạch và tổ chức dữ liệu.
- Mô tả các hàm dùng để làm việc với khung dữ liệu, bao gồm `read_csv()`, `data()`, và `datapasta()`.
- Thảo luận về sự khác biệt giữa tibbles và tribbles.
- So sánh và đối chiếu việc làm sạch dữ liệu bằng các công cụ khác nhau.
- Tạo ra và làm việc với dữ liệu trong R.

1. Khảo sát dữ liệu và R

Khung dữ liệu trong R

Bạn đã xem định nghĩa về khung dữ liệu trong R ở phần trước. Tiếp theo là một số lưu ý về định dạng dữ liệu trong R:

- Cột phải được đặt tên, sử dụng tên cột trống có thể gây ra vấn đề với kết quả sau này.
- Dữ liệu lưu trữ trong khung dữ liệu có thể ở nhiều kiểu khác nhau, chẳng hạn như số, hệ số hoặc ký tự.
- Mỗi cột phải chứa cùng một số mục dữ liệu, ngay cả khi một số mục dữ liệu bị thiếu.

Khung dữ liệu thường là điểm khởi đầu để phân tích dữ liệu trong R. Vì vậy,

điều quan trọng là phải hiểu các đặc điểm của khung dữ liệu và cách tạo chúng.

Ví dụ sau minh họa việc thao tác dữ liệu với khung dữ liệu trong R:

- Sử dụng tập dữ liệu **diamonds** có sẵn trong gói ggplot2 của tidyverse.
- Đầu tiên, nạp tập dữ liệu vào R bằng hàm **data()** và xem tập dữ liệu đã tải bằng hàm **view()**.

```
data("diamonds")
```

```
view(diamonds)
```

- Có 10 cột và 100 hàng trong khung dữ liệu này. Bạn có thể không muốn xem tất cả vì chúng quá nhiều, và bạn sẽ sử dụng hàm **head()** để chỉ hiển thị 6 hàng đầu tiên. Việc vô tình in khung dữ liệu đầy đủ vào bảng điều khiển có thể gây khó chịu và có thể mất nhiều thời gian để xử lý.

```
> head(diamonds)
# A tibble: 6 × 10
  carat cut      color clarity depth table price     x     y     z
<dbl> <ord>    <ord> <ord>    <dbl> <dbl> <int> <dbl> <dbl> <dbl>
1  0.23 Ideal    E     SI2     61.5   55   326   3.95   3.98   2.43
2  0.21 Premium E     SI1     59.8   61   326   3.89   3.84   2.31
3  0.23 Good    E     VS1     56.9   65   327   4.05   4.07   2.31
4  0.29 Premium I     VS2     62.4   58   334   4.2    4.23   2.63
5  0.31 Good    J     SI2     63.3   58   335   4.34   4.35   2.75
6  0.24 Very Good J     VVS2     62.8   57   336   3.94   3.96   2.48
```

- Bạn cũng có thể lấy cấu trúc của khung dữ liệu bằng cách sử dụng các hàm như **str()** và **colnames()** để lấy một số thông tin cấp cao như tên cột và loại dữ liệu có trong các cột đó.

```
> str(diamonds)
tibble [53,940 × 10] (S3: tbl_df/tbl/data.frame)
 $ carat : num [1:53940] 0.23 0.21 0.23 0.29 0.31 0.24 0.24 0.26 0.22 0.23 ...
 $ cut   : Ord.factor w/ 5 levels "Fair"<"Good"<...: 5 4 2 4 2 3 3 3 1 3 ...
 $ color : Ord.factor w/ 7 levels "D"<"E"<"F"<"G"<...: 2 2 2 6 7 7 6 5 2 5 ...
 $ clarity: Ord.factor w/ 8 levels "I1"<"SI2"<"SI1"<...: 2 3 5 4 2 6 7 3 4 5 ...
 $ depth : num [1:53940] 61.5 59.8 56.9 62.4 63.3 62.8 62.3 61.9 65.1 59.4 ...
 $ table : num [1:53940] 55 61 65 58 58 57 57 55 61 61 ...
 $ price : int [1:53940] 326 326 327 334 335 336 336 337 337 338 ...
 $ x     : num [1:53940] 3.95 3.89 4.05 4.2 4.34 3.94 3.95 4.07 3.87 4 ...
 $ y     : num [1:53940] 3.98 3.84 4.07 4.23 4.35 3.96 3.98 4.11 3.78 4.05 ...
 $ z     : num [1:53940] 2.43 2.31 2.31 2.63 2.75 2.48 2.47 2.53 2.49 2.39 ...
```

```
> colnames(diamonds)
[1] "carat" "cut" "color" "clarity" "depth" "table" "price" "x" "y" "z"
```

- Dùng hàm **mutate()** để thêm một cột và tính toán dữ liệu mới của cột. Ở đây, ta thêm một cột có tên `carat_2` và muốn tính toán cột mới này dựa vào cột `carat` cũ. Phần còn lại của khung dữ liệu sẽ vẫn như cũ.

```
> mutate(diamonds, carat_2 = carat*100)
# A tibble: 53,940 x 11
  carat cut      color clarity depth table price    x    y    z carat_2
<dbl> <ord>      <ord> <ord> <dbl> <dbl> <int> <dbl> <dbl> <dbl> <dbl>
1  0.23 Ideal    E      SI2     61.5   55   326   3.95  3.98  2.43   23
2  0.21 Premium E      SI1     59.8   61   326   3.89  3.84  2.31   21
3  0.23 Good    E      VS1     56.9   65   327   4.05  4.07  2.31   23
4  0.29 Premium I      VS2     62.4   58   334   4.2   4.23  2.63   29
5  0.31 Good    J      SI2     63.3   58   335   4.34  4.35  2.75   31
6  0.24 Very Good J      VVS2     62.8   57   336   3.94  3.96  2.48   24
7  0.24 Very Good I      VVS1     62.3   57   336   3.95  3.98  2.47   24
8  0.26 Very Good H      SI1     61.9   55   337   4.07  4.11  2.53   26
9  0.22 Fair    E      VS2     65.1   61   337   3.87  3.78  2.49   22
10 0.23 Very Good H      VS1     59.4   61   338   4     4.05  2.39   23
# ... with 53,930 more rows
```

Câu hỏi thảo luận

R là ngôn ngữ lập trình thường được sử dụng để phân tích thống kê, hình ảnh hóa và phân tích dữ liệu khác. R có một chút khác biệt so với các công cụ phân tích dữ liệu khác mà bạn đã khám phá cho đến nay. Hãy chia sẻ suy nghĩ của bạn về cách R quản lý tập dữ liệu so với SQL hoặc bảng tính? Ưu điểm và nhược điểm của từng công cụ này là gì?

Vui lòng trả lời bằng văn bản gồm hai đoạn văn trở lên (tổng cộng 100-150 từ).

Tìm hiểu về tibbles

Tibbles có một chút khác biệt so với khung dữ liệu chuẩn. Khung dữ liệu là một tập hợp các cột, giống như một bảng tính hoặc một bảng SQL. *Tibbles* giống như các khung dữ liệu được sắp xếp hợp lý được tự động đặt để chỉ kéo lên 10 hàng đầu tiên của tập dữ liệu và chỉ bao nhiêu cột có thể vừa với màn hình. Điều này thực sự hữu ích khi bạn đang làm việc với tập hợp dữ liệu lớn. Không giống như các khung dữ liệu, *Tibbles* không bao giờ thay đổi tên của các biến hoặc kiểu dữ liệu của các đầu vào của bạn. Nhìn chung, bạn có thể thực hiện nhiều thay đổi hơn đối với khung dữ liệu, nhưng *tibbles* sẽ dễ sử dụng hơn.

Tibbles là một phần của tidyverse. Vì vậy, nếu bạn đã cài đặt tidyverse, bạn sẽ có những thứ cần thiết để bắt đầu làm việc với tibbles.

Ví dụ sau minh họa việc thao tác dữ liệu với tibbles trong R.

- Tập dữ liệu diamonds có 10 cột và hàng nghìn hàng. Hình ảnh này hiển thị một phần của khung dữ liệu.

	carat	cut	color	clarity	depth	table	price	x	y	z
1	0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
2	0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
3	0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
4	0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
5	0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75
6	0.24	Very Good	J	VVS2	62.8	57.0	336	3.94	3.96	2.48
7	0.24	Very Good	I	VVS1	62.3	57.0	336	3.95	3.98	2.47
8	0.26	Very Good	H	SI1	61.9	55.0	337	4.07	4.11	2.53
9	0.22	Fair	E	VS2	65.1	61.0	337	3.87	3.78	2.49
10	0.23	Very Good	H	VS1	59.4	61.0	338	4.00	4.05	2.39
11	0.30	Good	J	SI1	64.0	55.0	339	4.25	4.28	2.73
12	0.23	Ideal	J	VS1	62.8	56.0	340	3.93	3.90	2.46
13	0.22	Premium	F	SI1	60.4	61.0	342	3.88	3.84	2.33
14	0.31	Ideal	J	SI2	62.2	54.0	344	4.35	4.37	2.71
15	0.20	Premium	E	SI2	60.2	62.0	345	3.79	3.75	2.27
16	0.32	Premium	E	I1	60.9	58.0	345	4.38	4.42	2.68
17	0.30	Ideal	I	SI2	62.0	54.0	348	4.31	4.34	2.68
18	0.30	Good	J	SI1	63.4	54.0	351	4.23	4.29	2.70
19	0.30	Good	J	SI1	63.8	56.0	351	4.23	4.26	2.71
20	0.30	Very Good	J	SI1	62.7	59.0	351	4.21	4.27	2.66
21	0.30	Good	I	SI2	63.3	56.0	351	4.26	4.30	2.71

Showing 1 to 22 of 53,940 entries, 10 total columns

- Bạn có thể tạo một tibble từ dữ liệu hiện có bằng hàm `as_tibble()` và chỉ định nguồn dữ liệu trong dấu ngoặc đơn của hàm. Trong trường hợp này, bạn sẽ đặt từ “diamonds”. Trong khi công cụ khung dữ liệu tích hợp của RStudio trả về hàng nghìn hàng trong tập dữ liệu, thì tibbles chỉ trả về 10 hàng đầu tiên trong một bảng được sắp xếp gọn gàng. Điều đó làm cho nó dễ dàng hơn để xem và in.

```
# A tibble: 53,940 x 10
  carat cut    color clarity depth table price    x    y    z
  <dbl> <ord> <ord> <ord>    <dbl> <dbl> <int> <dbl> <dbl> <dbl>
1 0.23 Ideal E     SI2     61.5    55   326  3.95  3.98  2.43
2 0.21 Prem... E     SI1     59.8    61   326  3.89  3.84  2.31
3 0.23 Good E     VS1     56.9    65   327  4.05  4.07  2.31
4 0.290 Prem... I     VS2     62.4    58   334  4.2   4.23  2.63
5 0.31 Good J     SI2     63.3    58   335  4.34  4.35  2.75
6 0.24 Very... J     VVS2    62.8    57   336  3.94  3.96  2.48
7 0.24 Very... I     VVS1    62.3    57   336  3.95  3.98  2.47
8 0.26 Very... H     SI1     61.9    55   337  4.07  4.11  2.53
9 0.22 Fair E     VS2     65.1    61   337  3.87  3.78  2.49
10 0.23 Very... H     VS1     59.4    61   338  4     4.05  2.39
# ... with 53,930 more rows
```

2. Quan sát dữ liệu

Làm sạch, tổ chức, và biến đổi dữ liệu với R

Thao tác trên dữ liệu có thể được thực hiện thông qua các toán tử R cơ bản: Số học (arithmetic), quan hệ (relational), logic (logical), và gán (assignment). Những toán tử này đều đã được trình bày ở những phần trước.

Các hàm thông dụng cho mục đích làm việc với dữ liệu trong R bao gồm:

- Tổ chức dữ liệu: `arrange()`, `group_by()`, `filter()`.
- Biến đổi dữ liệu: `separate()`, `unite()`, `mutate()`.

Bạn nên xem video minh họa về thao tác hàm trực tiếp trên R để có được hiểu biết tốt nhất.

Làm việc với dữ liệu thiên lệch

Để hiểu rõ những gì cần thực hiện khi làm việc với dữ liệu lệch, chúng ta hãy tìm hiểu tình huống sau. Kịch bản này được chia sẻ bởi một nhà phân tích định lượng, tạm gọi là John, với nhiệm vụ thu thập dữ liệu từ mọi người trên khắp thế giới. John giải thích cách anh ta phát hiện ra sự thiên vị trong dữ liệu của mình và cách sử dụng R để giải quyết vấn đề.

- John làm việc trong một nhóm thu thập dữ liệu phục vụ cho việc khảo sát.
- Một trong những nhiệm vụ mà nhóm của John cần thực hiện là so sánh song song. Họ hiển thị cho người dùng cùng lúc hai quảng cáo cạnh nhau, rồi hỏi người dùng liệu họ thích quảng cáo nào hơn.
- Sau nhiều lần lặp lại, nhóm phân tích nhận thấy sự thiên vị nhất quán có lợi cho mục đầu tiên. Ngoài ra, cũng có một sự sụt giảm có thể đo lường được trong sở thích đối với một mặt hàng nếu nó bị đổi vị trí của nó sang vị trí thứ hai.
- Vì vậy, nhóm phân tích quyết định thêm yếu tố ngẫu nhiên vào vị trí của quảng cáo bằng cách sử dụng R. Bằng cách này, họ đảm bảo rằng các mục xuất hiện ở vị trí thứ nhất và thứ hai với tần suất tương tự.

Hàm **sample()** đưa yếu tố ngẫu nhiên vào mã nguồn R, cho phép bạn lấy một mẫu phần tử ngẫu nhiên từ một tập dữ liệu. Việc thêm đoạn mã này sẽ xáo trộn các hàng trong tập dữ liệu một cách ngẫu nhiên. Đây chỉ là một trong nhiều hàm và phương thức trong R mà bạn có thể sử dụng để giải quyết sự sai lệch trong dữ liệu của mình. Tùy thuộc vào loại phân tích bạn đang tiến hành, bạn có thể cần kết hợp một số quy trình nâng cao trong lập trình của mình.

Câu hỏi thảo luận

Bạn hãy xem xét những điểm giống và khác nhau của R và bảng tính cho quy trình làm sạch dữ liệu. Viết câu trả lời gồm hai hoặc nhiều đoạn văn (tổng cộng 100-150 từ) thảo luận về những gì bạn nhận thấy trong khi làm sạch dữ liệu.

Bài đọc 4: Trực quan hóa – Tính mỹ thuật – Chú thích

R là công cụ rất thích hợp để tạo các trực quan hóa chi tiết. Trong phần này của khóa học, người học sẽ tìm hiểu cách sử dụng R để phát sinh trực quan hóa và khắc phục lỗi khi cần thiết. Họ cũng sẽ khảo sát các tính năng của R và RStudio để phép trực quan hóa và các chú thích có tính thẩm mỹ cao và sau đó sao lưu những kết quả này.

Mục tiêu học tập:

- Xác định các tính năng thẩm mỹ sẵn có trong R với tham chiếu đến kích thước, hình dạng, màu sắc, và biểu đồ.
- Giải thích một số vấn đề phổ thông liên quan đến trực quan hóa trong R.
- Sử dụng ggplot() để phát sinh các trực quan hóa cơ bản.
- Mô tả các lựa chọn để phát sinh trực quan hóa trong R.
- Thể hiện sự hiểu biết về chức năng của RStudio để lưu phép trực quan hóa.
- Tạo một biểu đồ trong ggplot2.
- Giải thích mục đích và logic cơ bản của gói ggplot2.

1. Tạo trực quan hóa dữ liệu trong R

Cơ bản về trực quan hóa trong R và tidyverse

Trực quan hóa (visualization) là một trong những phần quan trọng nhất của phân tích dữ liệu. Hình ảnh là hình thức truyền thông mạnh mẽ cho các bên liên quan biết một cách rõ ràng và hấp dẫn về việc dữ liệu của bạn có ý nghĩa như thế nào, đồng thời nêu bật những thông tin chi tiết chính. Hình ảnh đã giúp đưa câu chuyện về dữ liệu trở nên sống động và dễ hiểu hơn.

Các gói tương tác với R bao gồm ggplot2, plotly, lattice, RGL, dygraphs, leaflet, highcharter, patchwork, gganimate và ggridges. Trong đó, ggplot2 là công cụ được đánh giá là linh hoạt và phổ biến nhất.

Trực quan hóa dữ liệu với ggplot2

Gói ggplot2 hỗ trợ tạo các biểu diễn dữ liệu có thể tùy chỉnh và có chất lượng cao. Đây là hệ thống dựa trên ngữ pháp của đồ họa để mô tả và xây dựng hình ảnh trực quan hóa dữ liệu.

Ý tưởng thiết yếu đằng sau ngữ pháp đồ họa là bạn có thể xây dựng bất kỳ biểu đồ nào từ các thành phần cơ bản giống nhau, như các khối xây dựng. Các khối xây dựng này bao gồm:

- *Một tập dữ liệu (dataset)*
- *Tập hợp hình học (geom)* để cập đến đối tượng hình học được sử dụng để đại diện cho mỗi thành phần dữ liệu. Ví dụ, bạn có thể sử dụng điểm (point) để tạo biểu đồ phân tán, các thanh (bar) để tạo biểu đồ cột, các đường nét (line) để tạo sơ đồ đường, v.v.
- *Tập hợp thuộc tính thẩm mỹ* biểu diễn các thuộc tính thị giác của một đối tượng trong biểu đồ. Bạn có thể xem tính thẩm mỹ như một kết nối hoặc ánh xạ giữa một đối tượng hình học trong biểu đồ và một biến số trong dữ liệu. Ví dụ, trong biểu đồ phân tán, tính thẩm mỹ bao gồm những thứ như kích thước, hình dạng, màu sắc hoặc vị trí (trục x, trục y) của các điểm dữ liệu.

Để tạo một biểu đồ với ggplot2, bạn sẽ khởi động với việc chọn một tập dữ liệu, và tiếp theo là xác định cách tổ chức dữ liệu trên hệ tọa độ một cách trực quan – chọn tập hình học để đại diện cho điểm dữ liệu và tính thẩm mỹ để thể hiện các biến trong biểu đồ.

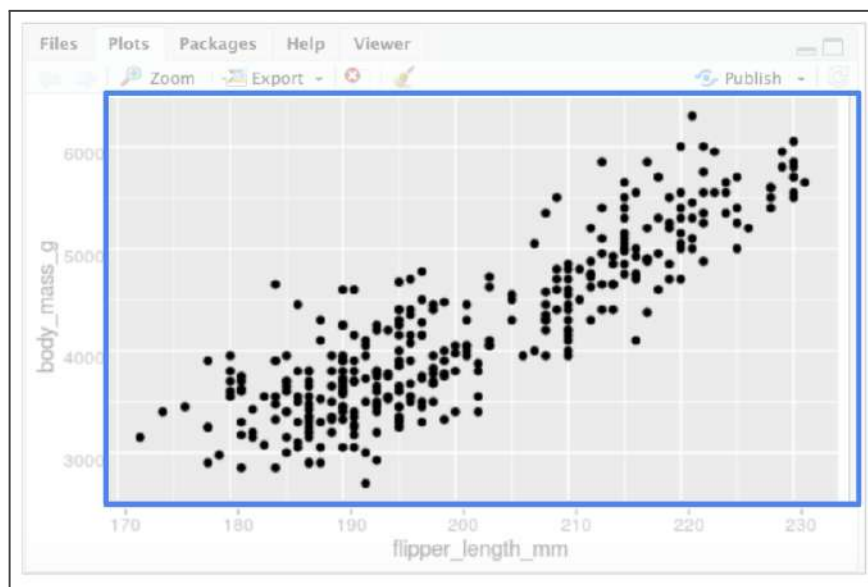
Ví dụ sau minh họa việc thao tác tạo biểu đồ phân tán với ggplot:

- Câu lệnh thực thi.

```
ggplot(data = penguins) + geom_point(mapping = aes(x = flipper_length_mm,  
y = body_mass_g)).
```

- `ggplot(data = penguins)`: Hàm **ggplot()** tạo ra một hệ tọa độ mà bạn có thể thêm vào các lớp (layer). Đối số của hàm là tập dữ liệu để sử dụng trong biểu đồ, trong trường hợp này, đó là “penguin”.

- Biểu tượng “+” dùng để thêm một lớp mới vào biểu đồ. Bạn sẽ hoàn thiện biểu đồ của mình bằng cách thêm một hoặc nhiều lớp vào ggplot().
- Hàm **geom_point()** sử dụng các điểm để tạo biểu đồ phân tán. Mỗi đối tượng hình học nhận một đối số ánh xạ, giúp xác định cách các biến trong tập dữ liệu được ánh xạ tới những thuộc tính trực quan. Đối số ánh xạ luôn được ghép nối với hàm **aes()**, các đối số x và y của hàm này chỉ định biến nào sẽ ánh xạ tới trục x và tới trục y của hệ tọa độ. Trong trường hợp này, bạn muốn ánh xạ biến “flipper_length_mm” sang trục x và biến “body_mass_g” sang trục y.



Câu hỏi thảo luận

Nếu bạn đã tham gia khóa học trước đó về chia sẻ dữ liệu thông qua kể chuyện, bạn có thể biết cách sử dụng Tableau để tạo hình ảnh trực quan dữ liệu hiệu quả. Bây giờ, bạn đã khám phá cách sử dụng mã R trong ggplot2 để tạo trực quan hóa dữ liệu.

Vui lòng viết một đến hai đoạn văn (tổng cộng 150-200 từ) mô tả suy nghĩ ban đầu của bạn về sự khác biệt giữa Tableau và ggplot2 khi nói đến trực quan hóa dữ liệu. Suy ngẫm về những câu hỏi sau:

- Điểm mạnh và hạn chế của Tableau khi nói đến trực quan hóa dữ liệu là gì?

- Bạn yêu thích những tính năng nào của Tableau?
- Nếu chưa quen với ggplot2, bạn nghĩ tính năng nào hữu ích nhất để hiển thị dữ liệu?
- Các công cụ trực quan hóa trong Tableau khác với công cụ trong ggplot2 như thế nào?

2. Khảo sát tính thẩm mỹ trong phân tích

Tính thẩm mỹ: các thuộc tính và khía cạnh

Có ba thuộc tính thẩm mỹ trong ggplot2:

- *Màu sắc*: tùy chỉnh màu của mọi điểm trên biểu đồ hoặc màu của từng nhóm dữ liệu.
- *Kích thước*: tùy chỉnh kích thước của các điểm theo nhóm dữ liệu.
- *Hình dạng*: tùy chỉnh hình dạng của các điểm theo nhóm dữ liệu.

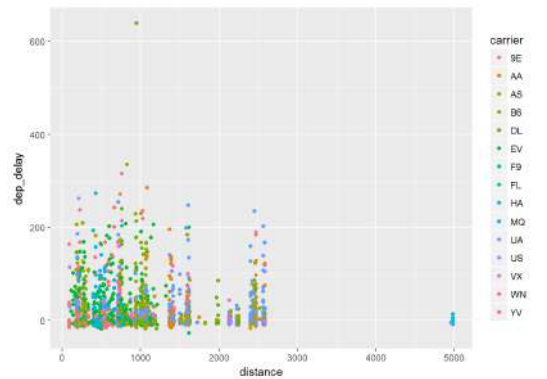
Bằng cách áp dụng các thuộc tính thẩm mỹ này vào công việc của bạn với ggplot2, bạn có thể tạo trực quan hóa dữ liệu trong R để truyền đạt rõ ràng các xu hướng trong dữ liệu của bạn.

Dưới đây là một ví dụ về cách các thuộc tính thẩm mỹ được hiển thị trong R:

- Biến data biểu diễn nội dung tập dữ liệu cần trực quan hóa.
- Hàm **geom_point()** sử dụng các điểm để tạo biểu đồ phân tán.
- Trục x và y lần lượt ánh xạ thuộc tính distance và dep_delay.

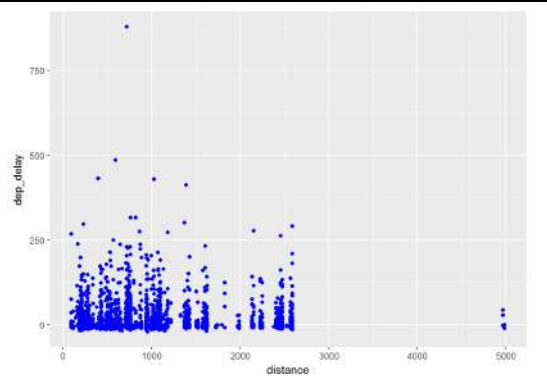
```
ggplot(data, aes(x=distance, y=
dep_delay, color=carrier)) +
geom_point()
```

color=carrier: thuộc tính thẩm mỹ về màu sắc của điểm trong biểu đồ được xác định dựa trên thuộc tính carrier (cột ở bên phải biểu đồ)



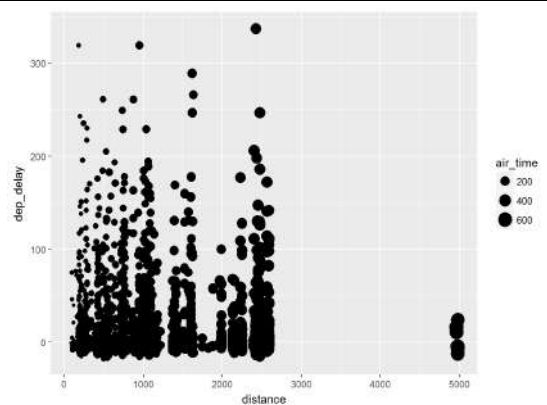
```
ggplot(data, aes(x=distance, y=
dep_delay)) +
geom_point(color=blue)
```

Đổi số color=blue của hàm geom_point() cho biết các điểm dữ liệu sẽ được thể hiện với màu xanh dương.



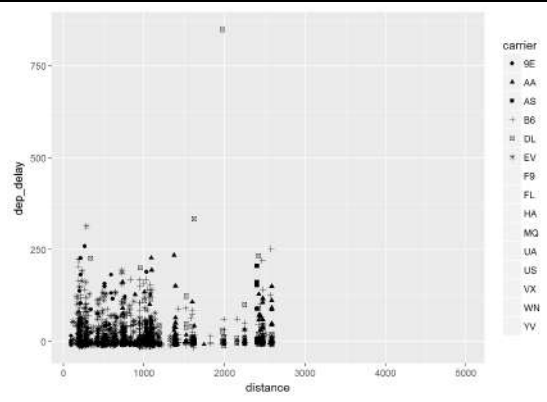
```
ggplot(data, aes(x=distance, y=
dep_delay, size=air_time)) +
geom_point()
```

size=air_time: thuộc tính thẩm mỹ về kích thước của điểm trong biểu đồ được xác định dựa trên thuộc tính air_time (cột ở bên phải biểu đồ)



```
ggplot(data, aes(x=distance, y=
dep_delay, shape = carrier)) +
geom_point()
```

shape=carrier: thuộc tính thẩm mỹ về hình dạng của điểm trong biểu đồ được xác định dựa trên thuộc tính carrier (cột ở bên phải biểu đồ)



Kết hợp cả ba yếu tố thẩm mỹ vào biểu đồ

```
ggplot(data, aes(x=distance, y= dep_delay, color=carrier, size=air_time,
  shape = carrier)) + geom_point()
```

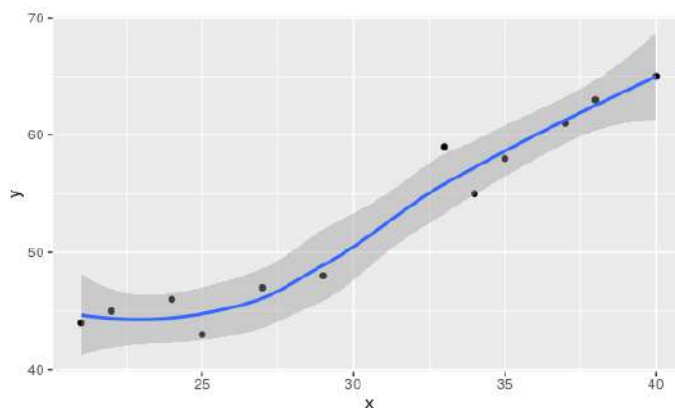
Làm việc nhiều hơn với ggplot

Làm mượt (smoothing)

Làm mượt cho phép phát hiện xu hướng dữ liệu ngay cả khi bạn không thể dễ dàng nhận thấy xu hướng từ các điểm dữ liệu được vẽ biểu đồ, giúp dữ liệu có ý nghĩa hơn đối với người quan sát bình thường.

Chức năng làm mượt của ggplot2 rất hữu ích vì đường làm mượt được thêm vào biểu đồ như một lớp mới.

```
ggplot(data, aes(x=distance,
  y= dep_delay)) +
  geom_point() +
  geom_smooth()
```



Có hai dạng làm trơn được hỗ trợ trong ggplot2, đó là:

- Làm trơn Loess: phù hợp nhất để làm trơn cho biểu đồ dưới 1000 điểm.
- Làm trơn GAM (Generalized Additive Model): phù hợp để làm trơn cho biểu đồ với lượng điểm lớn.

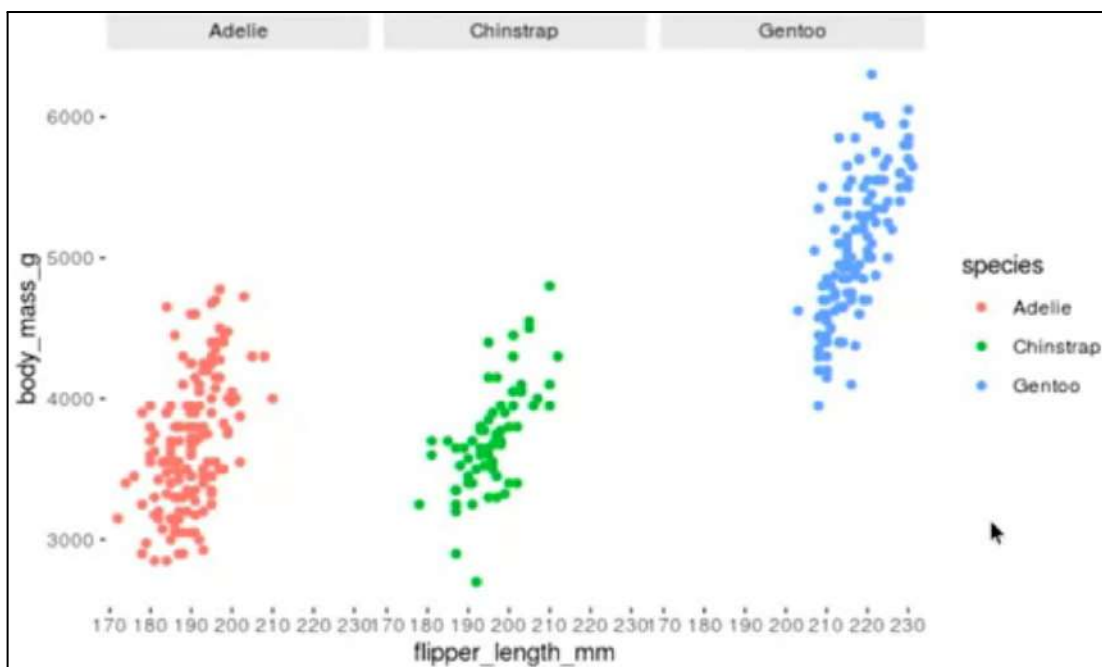
Chức năng Facet

Facet cho phép trích chọn hiển thị các nhóm nhỏ hơn hoặc tập hợp con từ dữ liệu ban đầu. Chức năng này hiển thị các mặt khác nhau của dữ liệu bằng cách đặt mỗi tập hợp con trên biểu đồ của riêng nó, từ đó giúp khám phá mẫu trong dữ liệu và tập trung vào mối quan hệ giữa các biến khác nhau. R hỗ trợ facet với hai hàm chủ đạo là **facet_wrap()** và **facet_grid()**.

Ví dụ, giả sử bạn đang xem dữ liệu doanh số của một công ty quần áo. Bạn có thể muốn chia nhỏ dữ liệu này theo danh mục để hiển thị các xu hướng cụ thể: quần áo trẻ em – quần áo người lớn hoặc thời trang mùa xuân – thời trang mùa thu. Hoặc nếu bạn đang thực hiện một cuộc khảo sát về mức độ tham gia của nhân viên, bạn có thể muốn chia nhỏ dữ liệu của mình theo nhiệm kỳ và so sánh nhân viên cấp cao với nhân viên mới.

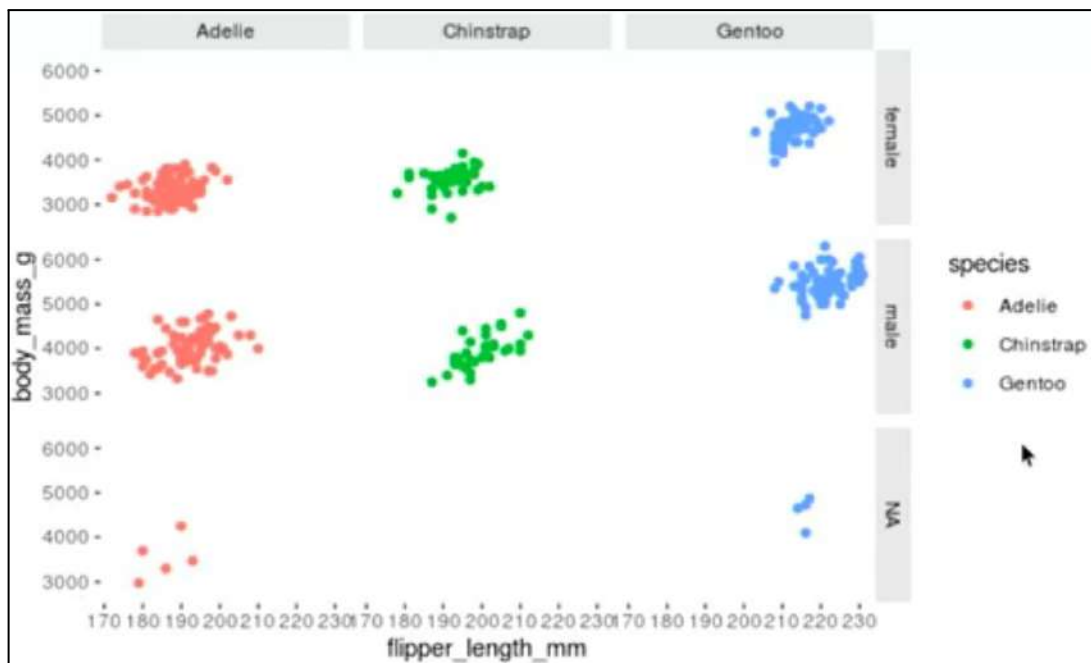
Dưới đây là ví dụ sử dụng `facet_wrap` trên tập dữ liệu `penguins`. Ba ô riêng biệt cho thấy mối quan hệ giữa khối lượng cơ thể và chiều dài cánh trong mỗi loài chim cánh cụt. Facet giúp tập trung vào những phần quan trọng của dữ liệu mà ta có thể không nhận ra chỉ trong một biểu đồ.

```
ggplot(data=penguins)+  
  geom_point(mapping=aes(x=flipper_length_mm,y=body_mass_g,color=species))+  
  facet_wrap(~species)
```



Dưới đây là ví dụ sử dụng `facet_grid` trên tập dữ liệu `penguins`. Ta thu được 9 biểu đồ con, tạo thành lưới 3 ô, mỗi ô dựa trên sự kết hợp của ba loài chim cánh cụt và ba loại giới tính. `Facet_grid` cho phép nhanh chóng tổ chức lại và hiển thị dữ liệu phức tạp, đồng thời giúp phát hiện mối quan hệ giữa các nhóm khác nhau dễ dàng hơn.

```
ggplot(data=penguins)+
  geom_point(mapping=aes(x=flipper_length_mm,y=body_mass_g,color=species))+
  facet_grid(sex~species)
```



Lọc dữ liệu (filter)

Lọc dữ liệu trước khi vẽ biểu đồ giúp tập trung vào tập con cụ thể của dữ liệu và có được nhiều chi tiết hướng đến mục tiêu hơn.

`ggplot` hỗ trợ chức năng này bằng hàm **`filter()`** trong gói `dplyr` của `tidyverse`.

Ví dụ:

```
data %>%
```

```
filter(variable1 == "DS") %>%
```

```
ggplot(aes(x = weight, y = variable2, colour = variable1)) +
```



```
geom_point(alpha = 0.3, position = position_jitter()) + stat_smooth(method = "lm").
```

3. Chú thích và sao lưu phép trực quan hóa

Thêm các chú thích trong R

Nhãn (label) và *chú thích* (annotation) thực sự hữu ích khi cần làm nổi bật các phần quan trọng trong dữ liệu của bạn và truyền đạt các điểm chính.

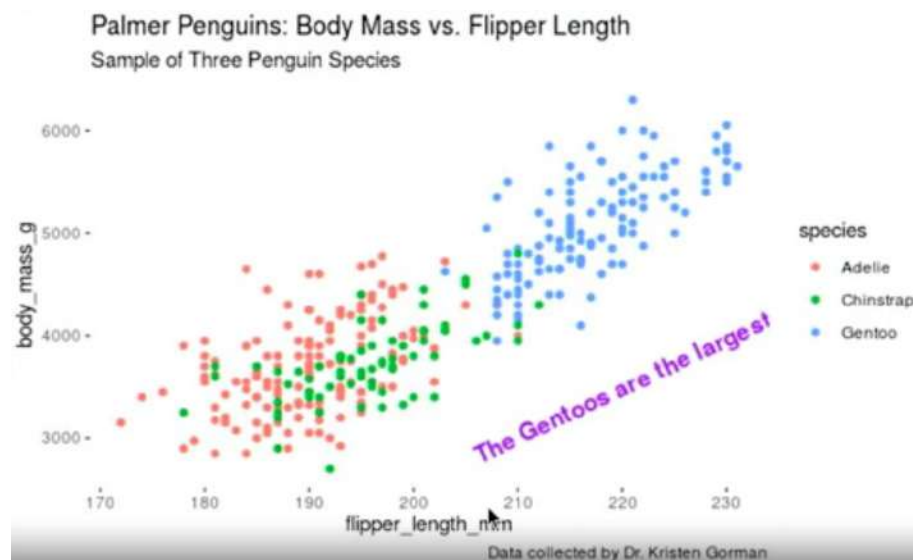
Đoạn mã dưới đây thêm tiêu đề vào biểu đồ bằng hàm **labs()**.

```
ggplot(data=penguins)+
  geom_point(mapping=aes(x=flipper_length_mm,y=body_mass_g,color=species))+
  labs(title="Palmer Penguins: Body Mass vs. Flipper Length")
```

Đoạn mã dưới đây thêm chú thích văn bản vào vị trí (x, y) trên biểu đồ bằng hàm **annotate()**.

```
annotate("text", x=220,y=3500,label="The Gentoos are the largest")
```

Đây là hình kết quả sau khi chạy các đoạn mã trước đó, và thêm một số chú thích khác.



Sao lưu phép trực quan hóa của bạn

Nội dung trực quan hóa có thể được lưu bằng cách sử dụng tùy chọn Export trong tab Plot của RStudio hoặc hàm `ggsave()` trong gói `ggplot2`.

Bạn có cũng thể mở một thiết bị đồ họa R như **`png()`** hoặc **`pdf()`** để lưu biểu đồ dưới dạng tập tin `.png` hoặc `.pdf`, rồi sau đó đóng thiết bị bằng cách sử dụng **`dev.off()`**.

```
pdf(file = "/Desktop/example.pdf", width = 4, height = 4)
```

```
plot(x = 1:10, y = 1:10)
```

```
abline(v = 0)
```

```
text(x = 0, y = 1, labels = "Random text")
```

```
dev.off()
```

Bài đọc 5: Tài liệu và Báo cáo

R có nhiều lựa chọn khác nhau để người học cân nhắc khi họ đã sẵn sàng sao lưu và trình diễn kết quả phân tích. Trong phần này của khóa học, họ sẽ khám phá R Markdown, một định dạng tập tin để tạo các tài liệu động với R. Họ tìm hiểu cách định dạng và xuất R Markdown, bao gồm cả việc tích hợp mã R trong tài liệu.

Mục tiêu học tập:

- Thể hiện sự hiểu biết về cách xuất R Markdown notebooks.
- Tích hợp các đoạn mã R vào R Markdown notebooks.
- Sử dụng định dạng cơ bản trong R Markdown để tạo cấu trúc và nhấn mạnh nội dung.
- Mô tả R Markdown notebook và sử dụng chúng để tạo tài liệu mã nguồn lập trình R.
- Tạo và phác thảo cấu trúc cho R Markdown notebook.
- Truy cập và sử dụng một mẫu R Markdown tùy chỉnh có trong gói R.
- Thể hiện sự hiểu biết về cách sử dụng các mẫu R Markdown.

1. Phát triển tài liệu và báo cáo trong RStudio

Tổng quan về R Markdown

R Markdown là định dạng tập tin để tạo tài liệu động với R, trong đó bạn có thể liên kết mã nguồn với báo cáo để có thể chia sẻ từng bước phân tích của mình. Tài liệu R Markdown được viết bằng ngôn ngữ Markdown.

Markdown là một cú pháp để định dạng các tập tin văn bản thuần túy. Sử dụng Markdown giúp viết và định dạng văn bản trong tài liệu dễ dàng hơn. Markdown cũng dễ đọc và dễ học. Ví dụ, nếu bạn muốn in nghiêng một từ hoặc cụm từ trong markdown, chỉ cần thêm một dấu gạch dưới (_) hoặc dấu hoa thị (*) ngay trước và sau từ đó.

R Markdown cho phép bạn chuyển đổi tập tin của mình sang nhiều định dạng khác nhau, chẳng hạn như tạo tài liệu HTML, PDF và Word, slide trình bày, hoặc bảng tổng quan (dashboard). Ngôn ngữ Markdown ban đầu được thiết kế cho đầu ra HTML do đó R Markdown có nhiều tính năng khả dụng nhất cho định dạng này, nhưng bạn cũng có thể đạt được kết quả tốt ở bất kỳ định dạng nào.

R Notebook cho phép người dùng chạy mã nguồn và hiển thị các biểu đồ và biểu đồ trực quan hóa mã. Bất kỳ tài liệu R Markdown nào cũng có thể được sử dụng như một sổ ghi chép. Điều này tạo ra một bức tranh tổng thể rõ ràng về phân tích và kết luận của bạn.

Bạn có thể tìm hiểu R Markdown thông qua:

- Tài liệu về R Markdown trong Rstudio: bao gồm một loạt các hướng dẫn sẽ giúp bạn tìm hiểu về các tính năng chính của R Markdown, bao gồm các đoạn mã, định dạng đầu ra, sổ ghi chép, tài liệu tương tác, v.v. Các hướng dẫn bao gồm các bài học trực tuyến mà bạn có thể hoàn thành trực tiếp trong không gian làm việc RStudio Cloud của mình.
- RStudio đã phát triển R Markdown Reference Guide và R Markdown Cheatsheet mà bạn có thể đánh dấu trang và sử dụng bất cứ khi nào bạn thực hành viết tập tin R Markdown.
- Sách tham khảo R for Data Science và R Markdown: The Definitive Guide.

Câu hỏi thảo luận

Trong hoạt động này, giả sử bạn đã tạo một R Markdown notebook. Hãy viết 2-3 câu (40-60 từ) để trả lời cho mỗi câu hỏi sau:

- Làm thế nào bạn có thể sử dụng R Markdown notebook trong tương lai?
- Bạn đã sử dụng định dạng nào trong R Markdown notebook này để giúp người khác hiểu phân tích của bạn dễ dàng hơn?

Đối chiếu R Markdown notebook với Jupyter notebook

Jupyter notebook là tài liệu chứa mã máy tính và các phần tử văn bản đa dạng thức - chẳng hạn như ghi chú, liên kết hoặc mô tả về phân tích và kết quả

của bạn. Jupyter notebook có thể hữu ích với mọi thứ, từ làm sạch và chuyển đổi dữ liệu, đến mô hình thống kê và hình ảnh hóa.

Bạn có thể tìm thấy Jupyter notebook trong nhiều công cụ trực tuyến, bao gồm Project Jupyter, Kaggle và Google Colaboratory (viết tắt là "Colab"). Những notebook này có thể là các tài liệu thực thi mà bạn có thể chạy để thực hiện phân tích.

Jupyter notebook tương thích với R, và bạn có thể coi chúng như một sự thay thế cho R Markdown. Cũng giống như tài liệu R Markdown, bạn có thể dễ dàng chia sẻ Jupyter notebook với các thành viên trong nhóm và các bên liên quan.

2. Tạo tài liệu R Markdown

Tài liệu Markdown được lưu ở định dạng tập tin RMD. Phần tiêu đề YAML cơ bản dành cho siêu dữ liệu (tức là dữ liệu về dữ liệu) của phần nội dung còn lại trong tập tin. Tiêu đề, tác giả, ngày tháng và loại tập tin đầu ra là những hạng mục thông tin được tự động đưa vào khi bạn tạo một tài liệu R Markdown mới. Phần còn lại trong vùng màu trắng dành cho văn bản soạn thảo. Bạn có thể định dạng văn bản để bao gồm các liên kết, danh sách có thứ tự, phương trình và hơn thế nữa. Ví dụ, định dạng tiêu đề bằng dấu hashtag #, tiêu đề được dẫn trước bằng càng nhiều dấu hashtag sẽ có kích thước càng nhỏ, giúp tạo tập hợp tiêu đề rõ ràng và hệ thống.

```
---
title: "Untitled"
output: html_document
date: '2022-07-15'
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the Knit button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```{r cars}
summary(cars)
```
```

Câu hỏi thảo luận

Với R Markdown notebook, bạn có thể tạo tài liệu R Markdown tương tác cho phép người dùng chạy mã nguồn và khám phá các biểu đồ và biểu đồ trực

quan hóa mã đó. Bây giờ bạn đã quen thuộc hơn với R Markdown, hãy nghĩ về cách sử dụng nó trong quá trình phân tích của riêng bạn.

Một số lợi ích của việc sử dụng sổ ghi chép R Markdown là gì? Và có dự án nào bạn đã hoàn thành mà bạn có thể ghi lại bằng R Markdown notebook không? Viết hai đoạn văn (150-200 từ) thảo luận về lợi ích của R Markdown notebook và cách bạn có thể sử dụng chúng.

3. Hiểu biết đoạn mã và thao tác xuất tài liệu

Thêm đoạn mã vào R Markdown notebook

Dấu phân tách (delimiter) là một ký tự chỉ ra phần đầu hoặc phần cuối của một mục dữ liệu. Bạn cũng có thể nhập các dấu phân cách này trực tiếp vào tập tin như sau: đầu tiên là gõ ba dấu gạch ngược, theo sau là dấu r trong dấu ngoặc nhọn để bắt đầu đoạn mã, và cuối cùng ba dấu gạch ngược để kết thúc nội dung phân cách.

Trong ví dụ bên dưới, hình trái là trong lúc biên soạn tài liệu ta chèn mã để tải gói tidyverse, đọc tập dữ liệu diamonds (đã gặp trước đó) và hiển thị; hình phải là tài liệu Markdown kết quả được tạo ra như một tài liệu HTML.

```

{r loading packages}
library(tidyverse)
data(diamonds)
view(diamonds)

```

2022-07-15

```

library(tidyverse)

## — Attaching packages — tidyverse 1.3.1 —

## ✓ ggplot2 3.3.6   ✓ purrr  0.3.4
## ✓ tibble  3.1.7   ✓ dplyr  1.0.9
## ✓ tidyr   1.2.0   ✓ stringr 1.4.0
## ✓ readr   2.1.2   ✓ forcats 0.5.1

## — Conflicts — tidyverse_conflicts() —
## X dplyr::filter() masks stats::filter()
## X dplyr::lag()    masks stats::lag()

data(diamonds)
view(diamonds)

```

Câu hỏi thảo luận

Trong hoạt động này, bạn đã tạo các đoạn mã trong tập tin R Markdown có hình ảnh trực quan với ggplot2. Hãy viết 2-3 câu (40-60 từ) để trả lời cho mỗi câu hỏi sau:

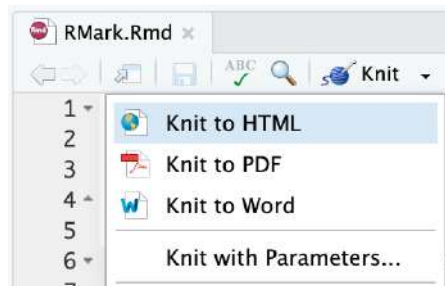
- Làm thế nào để thêm các đoạn mã cải thiện khả năng sử dụng của tập R Markdown của bạn?

- Trực quan hóa dữ liệu trong tệp R Markdown khác với trực quan hóa dữ liệu trong một chương trình như Tableau như thế nào?

Kết xuất tài liệu R Markdown

Tài liệu R Markdown được kết xuất theo định dạng chỉ định trong YML header hoặc định dạng khác thông qua chức năng Knit trên giao diện.

```
---  
title: "Untitled"  
output: html_document  
date: '2022-07-15'  
---
```



Câu hỏi thảo luận

Trong hoạt động này, bạn đã sử dụng một gói R để mở một mẫu Markdown và chỉnh sửa nó với thông tin của riêng bạn. Hãy viết 2-3 câu (40-60 từ) để trả lời câu hỏi sau: Bạn có thể sử dụng các mẫu R Markdown trong phân tích trong tương lai của mình như thế nào?

Phần 2

HƯỚNG DẪN

TRẢ LỜI CÂU HỎI

Lập trình và phân tích dữ liệu

Thế giới lập trình

1. Viết 2-3 câu (tổng cộng 20-60 từ) để trả lời các câu hỏi sau:

- Nền tảng chuyên môn của bạn là gì?
- Điều gì khiến bạn đăng ký tham gia khóa học này?

2. Tiếp theo, viết 3-5 câu (tổng cộng 60-100 từ) để chia sẻ suy nghĩ của bạn về nền tảng trong lập trình:

- Bạn đã từng làm việc với ngôn ngữ lập trình nào?
- Bạn đã sử dụng lập trình cho các dự án cá nhân hay chuyên nghiệp nào?
- Bạn thích điều gì nhất về lập trình?
- Nếu bạn là người mới học lập trình, bạn có cảm nghĩ gì về việc học lập trình bằng R: bạn thấy hào hứng, có chút lo lắng, hay cả hai?

3. Viết 2-3 câu (40-60 từ) trả lời cho mỗi câu hỏi dưới đây:

- Điều gì khiến bạn quyết định tìm hiểu về R?
- Bạn hào hứng tìm hiểu những phần nào của R? Những phần nào có vẻ khó?

Lập trình như nhà phân tích dữ liệu

1. Bạn đã tải xuống và cài đặt các tập tin cho ngôn ngữ lập trình R. Hãy viết 2-3 câu (40-60 từ) để trả lời cho mỗi câu hỏi sau:

- Ưu điểm của việc cài đặt R thay vì sử dụng nó trên nền tảng trực tuyến là gì?
- Học R sẽ giúp bạn xây dựng kỹ năng phân tích dữ liệu như thế nào?

2. Bạn đã sử dụng R Console để viết một số chức năng cơ bản. Hãy viết 2-3 câu (40-60 từ) để trả lời cho mỗi câu hỏi sau:

- R Console cho bạn biết điều gì về lập trình trong giao diện R?

- Sự khác biệt giữa việc sử dụng R Console so với việc viết mã R trong tệp tin văn bản là gì?

Học lập trình với RStudio

1. Bạn đã truy cập Rstudio như một IDE để lập trình R. Hãy viết 2-3 câu (40-60 từ) để trả lời cho mỗi câu hỏi sau:

- Trải nghiệm sử dụng RStudio khác với các môi trường khác như chương trình R chuẩn như thế nào? (Nếu bạn không cài đặt R vào thiết bị của mình, làm thế nào để so sánh các tính năng?)
- Lợi thế của việc sử dụng RStudio Cloud là gì? Ngược lại, lợi thế của việc sử dụng RStudio Desktop là gì?
- Bạn có gặp những trở ngại nào hay không?

2. RStudio cung cấp cho bạn một không gian làm việc duy nhất, nơi bạn có thể sử dụng R cho tất cả các giai đoạn của quá trình phân tích dữ liệu. Hãy viết một văn bản gồm hai hoặc nhiều đoạn văn (150-200 từ) mô tả những suy nghĩ ban đầu của bạn về RStudio:

- Bạn nghĩ RStudio có thể giúp bạn như thế nào trong vai trò nhà phân tích dữ liệu trong tương lai?
- Bạn yêu thích những tính năng nào của RStudio?
- Nếu bạn chưa quen với R và RStudio, bạn nghĩ tính năng nào sẽ hữu ích nhất cho bạn với tư cách là một người học?

Lập trình với RStudio

Khám phá lập trình trong R

1. Bây giờ bạn đã viết các truy vấn bằng SQL và sử dụng mã để lập trình trong R, bạn có thể nhận thấy một số điểm tương đồng giữa hai điều này:

Hãy viết một thảo luận gồm hai hoặc nhiều đoạn văn (tổng cộng 100-150 từ) về bất kỳ điểm tương đồng nào mà bạn có thể gặp phải.

Làm việc với dữ liệu trong R

Khảo sát dữ liệu trong R

1. R là ngôn ngữ lập trình thường được sử dụng để phân tích thống kê, hình ảnh hóa và phân tích dữ liệu khác. R có một chút khác biệt so với các công cụ phân tích dữ liệu khác mà bạn đã khám phá cho đến nay:

- Hãy chia sẻ suy nghĩ của bạn về cách R quản lý tập dữ liệu so với SQL hoặc bảng tính? Ưu điểm và nhược điểm của từng công cụ này là gì?
- Vui lòng gửi câu trả lời bằng văn bản gồm hai đoạn văn trở lên (tổng cộng 100-150 từ) trả lời câu hỏi này.

Quan sát dữ liệu

1. Bạn hãy xem xét những điểm giống và khác nhau của R và bảng tính cho quy trình làm sạch dữ liệu:

Viết câu trả lời gồm hai hoặc nhiều đoạn văn (tổng cộng 100-150 từ) thảo luận về những gì bạn nhận thấy trong khi làm sạch dữ liệu.

Trực quan hóa - Tính mỹ thuật - Chú thích

Trực quan hóa trong R

1. Vui lòng viết một đến hai đoạn văn (tổng cộng 150-200 từ) mô tả suy nghĩ ban đầu của bạn về sự khác biệt giữa Tableau và ggplot2 khi nói đến trực quan hóa dữ liệu.

Suy ngẫm về những câu hỏi sau:

- Điểm mạnh và hạn chế của Tableau khi nói đến trực quan hóa dữ liệu là gì? Bạn yêu thích những tính năng nào của Tableau?
- Nếu chưa quen với ggplot2, bạn nghĩ tính năng nào hữu ích nhất để hiển thị dữ liệu?
- Công cụ trực quan hóa trong Tableau khác với công cụ trong ggplot2 như thế nào?

Tài liệu và báo cáo

Xây dựng tài liệu và báo cáo trong RStudio

1. Trong hoạt động này, giả sử bạn đã tạo một R Markdown notebook.

Hãy viết 2-3 câu (40-60 từ) để trả lời cho mỗi câu hỏi sau:

- Làm thế nào bạn có thể sử dụng R Markdown notebook trong tương lai?
- Bạn đã sử dụng định dạng nào trong R Markdown notebook này để giúp người khác hiểu phân tích của bạn dễ dàng hơn?

Tạo tài liệu R Markdown

1. Một số lợi ích của việc sử dụng sổ ghi chép R Markdown là gì?

- Và có dự án nào bạn đã hoàn thành mà bạn có thể ghi lại bằng sổ ghi chép R Markdown không?
- Viết hai đoạn văn (150-200 từ) thảo luận về lợi ích của R Markdown notebook và cách bạn có thể sử dụng chúng.

Hiểu đoạn mã và kết xuất tài liệu

1. Trong hoạt động này, bạn đã tạo các đoạn mã trong tập tin R Markdown có hình ảnh trực quan với ggplot2.

Hãy viết 2-3 câu (40-60 từ) để trả lời cho mỗi câu hỏi sau:

- Làm thế nào để thêm các đoạn mã cải thiện khả năng sử dụng của tập R Markdown của bạn?
- Trực quan hóa dữ liệu trong tập R Markdown khác với trực quan hóa dữ liệu trong một chương trình như Tableau như thế nào?

2. Trong hoạt động này, bạn đã kết xuất tập tin Rmd dưới dạng html và pdf.

Hãy viết 2-3 câu (40-60 từ) để trả lời cho mỗi câu hỏi sau:

- Lợi thế của việc kết xuất dưới dạng tập html hoặc pdf là gì? Có định dạng nào bạn nghĩ mình sẽ sử dụng thường xuyên hơn không?

- Bạn định sử dụng hoặc lưu trữ các tập tin Rmd vừa kết xuất như thế nào?

3. Trong hoạt động này, bạn đã sử dụng một gói R để mở một mẫu Markdown và chỉnh sửa nó với thông tin của riêng bạn:

Hãy viết 2-3 câu (40-60 từ) để trả lời câu hỏi sau: Bạn có thể sử dụng các mẫu R Markdown trong phân tích trong tương lai của mình như thế nào?