**ĐẠI HỌC HUẾ**



KHOA KỸ THUẬT VÀ CÔNG NGHỆ

🙠🙟🕮🙝🙢

**Icon

Description automatically generated**

**BÁO CÁO**

**ĐỒ ÁN**

**NĂM HỌC 2020-2021**

**Giáo viên hướng dẫn: HỒ QUỐC DŨNG**

**Lớp: KHDL & TTNT**

|  |
| --- |
| Số phách  *(Do hội đồng chấm thi ghi)* |

**Thừa Thiên Huế, ngày 30 tháng 12 năm 2021**

**ĐẠI HỌC HUẾ**



KHOA KỸ THUẬT VÀ CÔNG NGHỆ

🙠🙟🕮🙝🙢

****

(MẪU BÌA PHỤ)

**BÁO CÁO**

**ĐỒ ÁN**

**NĂM HỌC 2020-2021**

**Giảng viên hướng dẫn: HỒ QUỐC DŨNG**

**Lớp: KHDL & TTNT**

**Sinh viên thực hiện: PHẠM QUỐC ĐẠT**

|  |
| --- |
| Số phách  *(Do hội đồng chấm thi ghi)* |

**Thừa Thiên Huế, ngày 30 tháng 12 năm 2021**

**MỤC LỤC**

[TÀI LIỆU THAM KHẢO 5](#_Toc92046849)

[Chương I: Crawl dữ liệu từ web 6](#_Toc92046850)

[1. Mục tiêu 6](#_Toc92046851)

[2.Crawl dữ liệu 6](#_Toc92046852)

[2.1 Khái niệm Crawl 6](#_Toc92046853)

[2.2 Cài đặt thư viện Scrapy, Selenium 7](#_Toc92046854)

[2.3 Crawl dữ liệu 7](#_Toc92046855)

[Chương II: Phân tích dữ liệu bằng R 21](#_Toc92046856)

[1.Mục tiêu 21](#_Toc92046857)

[2. Phân tích dữ liệu 21](#_Toc92046858)

[2.1 Phân tích thống kê mô tả 21](#_Toc92046859)

[2.2 Trực quan hóa dữ liệu 25](#_Toc92046860)

[2.3 Phân tích thống kê suy diễn 30](#_Toc92046861)

[Chương III: Kết quả kiểm tra đạo văn 34](#_Toc92046862)

# TÀI LIỆU THAM KHẢO

**TIẾNG ANH**

[1] Website Scraping with Python: Using BeautifulSoup and Scrapy - Gábor László Hajba

[2] Learning Scrapy - Dimitris Kouzis – Loukas

[3] Web Scraping with Python: Collecting More Data from the Modern Web - Ryan Mitchell

[4] Hands-On Web Scraping with Python: Perform advanced scraping operations using various Python libraries and tools such as Selenium, Regex, and others - Anish Chapagain

**INTERNET**

[1] [Web Scraping With Selenium & Scrapy - Karthikeyan P](https://medium.com/swlh/web-scraping-with-selenium-scrapy-9d9c2e9d83b1)

[2] [Scraping Javascript Enabled Websites using Scrapy-Selenium](https://www.geeksforgeeks.org/scraping-javascript-enabled-websites-using-scrapy-selenium/)

[3] [Web Scraping: A Brief Overview of Scrapy and Selenium, Part I - Anastasia Reusova](https://towardsdatascience.com/web-scraping-a-simple-way-to-start-scrapy-and-selenium-part-i-10367164c6c0)

[4] [Scrapy 2.5 documentation](https://docs.scrapy.org/en/latest/)

# Chương I: Crawl dữ liệu từ web

## 1. Mục tiêu

Mục đích của bài viết này là trình bày cách lấy dữ liệu cụ thể ở đây là dữ liệu bán xe ô tô từ trang web oto.com.vn

Graphical user interface, website

Description automatically generated

Hình 1

## 2.Crawl dữ liệu

### 2.1 Khái niệm Crawl

Crawl (Spider hay là Bot) là một thuật ngữ mô tả quá trình thu thập dữ liệu trên web được gọi chung là Web Crawler. Phần mềm này được thiết kế để có thể duyệt website trên mạng World Wide Web, chúng tiến hành phân tích mã nguồn HTML, đọc dữ liệu và lọc ra theo yêu cầu của người sử dụng. Quá trình này sẽ dừng lại cho đến khi con bot duyệt hết tất cả các đường liên kết trang đầu cũng như các trang có liên quan.

### 2.2 Cài đặt thư viện Scrapy, Selenium

Để cạo dữ liệu, chúng ta sẽ sử dụng Python và các thư viện hỗ trợ duyệt web và lấy dữ liệu bao gồm:

* Thư viện Scrapy: cung cấp cho bạn khả năng phân tích và truy cập vào các trang web là lọc dữ liệu
* Thư viện Selenium: cung cấp một giao diện, tương tác như một web browser

Để cài đặt thư viện Scrapy cần sử dụng phiên bản Python 3.6 trở lên đối với Scrapy, đối với Selenium thì áp dụng được với mọi phiên bản. Tôi giả sử rằng bạn đã cài đặt được Python (hoặc bạn có thể tham khảo [tại đây](https://www.python.org/downloads/), để cài đặt bất kì phiên bản Python nào bạn muốn). Có nhiều cách để cài đặt thư viện cụ thể:

* Anaconda hay Minicoda:

A picture containing chart

Description automatically generated

A picture containing text

Description automatically generated

* Pip

Shape, rectangle

Description automatically generated

Shape, rectangle

Description automatically generated with medium confidence

### 2.3 Crawl dữ liệu

**a. Tích hợp Selenium vào Scrapy**

Đầu tiên phải khởi tạo một dự án scrapy mới. Chạy câu lệnh theo cú pháp [scrapy startproject *tên thư mục*] trong Command Prompt (như hình bên dưới). Một thư mục có tên “Crawler” đã được tạo.

Text

Description automatically generated

Hình 2

Cấu trúc của một dự án scrapy có dạng như sau:

* Trong folder “Crawler” bao gồm một folder “spiders” và một file scrapy.cfg là một file tệp cấu hình của scrapy
* Trong folder “spiders” có các file python:
  + \_\_init\_\_.py:
  + items.py: file định nghĩa tên của các trường dữ liệu mà bạn muốn crawl về
  + middlewares.py: file phần mềm trung gian c
  + pipelines.py
  + Text

    Description automatically generatedsettings.py: file cài đặt của dự án

Tiếp theo là cài đặt chromedriver hoặc geckodriver (dành cho những bạn sử dụng Mozilla Firefox) vào scrapy, để thực hiện bước này trước đó bạn phải kiểm tra phiên bản Chrome đang sử dụng và tải chromedrive tương ứng với phiên bản đó. Cách kiểm tra phiên bản Chrome rất đơn giản, bạn hãy truy cập vào “**⁝**” tại góc trên bên phải => “Trợ giúp” => “Giới thiệu về Chrome” => Kiểm tra phiên bản Google Chrome => Tiến hành tải chromedriver về máy.

Graphical user interface, text, application, email

Description automatically generated

Hình 3

Sau khi tải được chromedriver về, bạn copy file chromedriver.exe vào folder Scrapy vừa tạo.

Graphical user interface, application

Description automatically generated

Hình 4

Tiếp tục, bạn viết đoạn mã dưới đây vào file settings.py. Như vậy, chúng ta đã hoàn tất việc tạo dự án và tích hợp selenium vào scrapy để tiến hành crawl dữ liệu.

Text

Description automatically generated

Hình 5

**b. Phân tích trang web và xây dựng hàm**

Tôi vào danh mục “Mua bán ô tô” và nhận thấy rằng có rất nhiều hãng xe khác nhau và có hơn 40000 chiếc xe được rao bán. Mục tiêu của tôi là crawl thông tin của tất cả chiếc xe về và phân loại chúng theo từng hãng xe (ví dụ: Mercedes-Benz, Audi, Honda, Madza, …) và theo từng dòng xe (Vd: Xét hãng xe Toyota thì chúng ta có những loại như Toyota Vios, Toyota Innova, Toyota Camry, Toyota Corolla, …) vậy nên tôi sẽ tiến hành crawl theo từng hãng xe. Nhưng có một vấn đề là có một số hãng xe trùng lặp và tôi sẽ không lấy những hãng xe này.

Graphical user interface, text, application

Description automatically generated

Hình 6

Tiến hành phân tích html, tôi thu được đường link dẫn bán xe của tùng hãng nằm trong một class là “sub-menu col” và “sub-menu col mgt\_30”.

Graphical user interface

Description automatically generated with medium confidence

Hình 7

Tiếp tục phân tích html như trên ta cũng thu được là các đường link dẫn đến từng nhóm xe nằm trong một class là “inner-type-brand” (hình 8). Truy cập tiếp vào từng dòng xe, chúng ta có một trang web gồm nhiều đường link dẫn đến thông tin bán chiếc xe (với hãng xe và dòng xe đã được chọn ở phía trên – hình 9). Trong mỗi đường link là thông tin của sản phẩm bao gồm: năm sản xuất, kiểu dáng, tình trạng, giá tiền, ảnh sản phẩm, … (hình 10)

Graphical user interface, text, application, email

Description automatically generated

Hình 8

Graphical user interface, text

Description automatically generated

Hình 9

Graphical user interface

Description automatically generated

Hình 10

Sau khi phân tích html xong, chúng ta sử dụng scrapy và selenium để lấy dữ liệu về. Khai báo các thư viện cần thiết để viết mã

Text

Description automatically generated

Hình 11

Tạo hàm để lấy đường liên kết đến các hãng xe. Như đã đề cập ở phía trước, các đường link năm trong các class có tên là “sub-menu col” và “sub-menu col mgt\_30” nên chúng ta sử dụng hàm ***find\_elements\_by\_xpath*** để tìm đến các class, bạn có thể sử dụng nhiều phương thức truy tìm khác (by\_CSS Selector, by\_class\_name, by\_id\_name,…) sử dụng ***get\_attribute(‘href’)*** để lấy đường link trong đó và lưu nó vào trong một list là “link\_trademark” (hình 12). Tương tự, bạn có thể lấy được tên của các hãng xe, tên của các dòng xe (hình 13)

Text

Description automatically generated

Hình 12

Text

Description automatically generated

Hình 13

Tiếp theo là hàm lấy link các dòng xe và link sản phẩm, vì link sản phẩm nằm trong một trang liên kết với trang gốc nên để lấy được dữ liệu của sản phẩm, tôi sử dụng vòng lặp for để trong một list là tất cả các link hãng xe đã lấy được ở phía trên từ đó truy cập vào trang liên kết để lấy link dòng xe (hình 14).

Text

Description automatically generated

Hình 14

Từ danh sách link của các dòng xe, khi yêu cầu ***driver.get(url)*** thì tôi sẽ truy cập được vào trang liên kết này (hình 15). Nhận thấy rằng, số lượng tin rao bán mỗi dòng xe trên từng hãng xe là khác nhau và trang web không thể hiển thị một lúc toàn bộ các tin rao bán (tối đa là 15 tin), vì vậy tôi sử dụng phương thức cuộn trang để click chuột vào ô “Hiển thị thêm” với số lần click là giá trị của phép chia lấy nguyên của số tin rao bán xe với 15. Để cuộn trang, tôi sử dụng phương thức ***send\_keys*** và gửi vào giá trị ***Keys.END*** điều này tương đương với việc nhấn nút END trên bàn phím. Cuối cùng tôi có hàm sau

Graphical user interface, application

Description automatically generated

Hình 15

Text

Description automatically generated

Hình 16

**c. Sử dụng Scrapy để crawl dữ liệu**

Trước khi scrapy và sau khi đưa ra ý tưởng, phân tích html, tôi chắc chắn rằng bạn đã định hình được dữ liệu bạn bao gồm những feature gì. Như đã trình bày trước, tệp ***“items.py”*** dùng để khai báo tên trường dữ liệu (cột dữ liệu). Với câu lệnh ***“tên\_trường\_dữ\_liệu = scrapy.Field()”***. Một cách khác là sử dụng lệnh ***“yield”*** trong scrapy. Trong bài viết này, tôi sử dụng lệnh yield để lưu dữ liệu (hình 17) (sẽ trình bày sau). Ở đây, tôi có các feature bao gồm:

* Name: Tên của sản phẩm
* Trade: Hãng xe
* Brand: Phân khúc xe
* Time\_up: Thời gian đăng tin rao bán
* Price: Giá xe
* Year\_product: Năm sản xuất sản phẩm
* Style: Kiểu dáng xe
* Status: Xe mới hay được bán lại
* Origin: Nguồn gốc của hãng xe
* KM: Số KM đã đi (0: đối với xe mới)
* Local: Tỉnh thành bán xe (
* Box\_car: Hộp số của sản phẩm
* Fuel: Nhiên liệu sử dụng của sản phẩm

Hình 17

Đầu tiên, tôi bắt đầu với ***start\_url*** là [***https://oto.com.vn/mua-ban-xe***](https://oto.com.vn/mua-ban-xe)để lấy đường liên kết đến tất cả hãng xe, kết quả trả về một list, sau đó sử dụng vòng lặp ***for*** để truy cập từng đường liên kết trong danh sách để lấy tên của dòng xe, đường liên kết đến dòng xe trong mỗi hãng và tên của dòng xe, chúng được lưu lần lượt vào các list là ***“name\_trade”, “link\_style”, “name\_style”*** bằng các hàm đã định nghĩa ở trước (xem Hình 12, 13, 14). Vậy là tôi đã có một danh sách chứa tất cả tên của hãng xe, dòng xe.

Text

Description automatically generated

Hình 18

Dùng scrapy để duyệt qua tất cả các ***“link\_style”*** để lấy tất cả đường liên kết các sản phẩm. Lúc này, selenium đã vào được các trang rao bán các sản phẩm và nhiệm vụ bây giờ là lấy tất cả các tin rao bán trong trang. Để lấy được hết các tin, selenium phải hiển thị hết toàn bộ các tin bán xe. Để thực hiện được điều đó tôi đã sử dụng lệnh ***“click button”*** và ấn phím ***“END”.*** Có một vấn đề nhỏ ở đây là lấy số lượng tin rao bán để từ đó tìm được số lần click button, đối với số lượng tin nhỏ hơn 1000 thì mọi thứ vẫn tiến triển tốt, nhưng khi số lượng tin vượt quá con số trên, một lỗi nhỏ xảy ra là trang web sử dụng dấu phẩy để phân cách hàng nghìn với hàng trăm. Mà trong số nguyên không sử dụng dấu “,” để phân cách, vậy nên tôi sử dụng hàm ***“replace”*** để xử lí. Với mỗi lần click button, tôi đã dừng chương trình một khoảng trễ là 5s để selenium có đủ thời gian để hiện thị cũng như lấy được tất cả các đường link sản phẩm. Tất cả đường link được lưu vào 1 list là ***“link\_product”***

Text

Description automatically generated

Hình 19

Tiếp đến dung lệnh yield để đưa lệnh Request cho mỗi đường link sản phẩm. Sử dụng lệnh ***“callback”*** để quay lại và lấy dữ liệu xuống

Text

Description automatically generated

Hình 20

Trước khi lấy dữ liệu xuống và lưu lại, tôi đã xử lí một chút để thuận tiện cho việc phân tích sau này (hình 21, 22).

Text

Description automatically generated

Hình 21

Text

Description automatically generated

Hình 22

Shape

Description automatically generated with medium confidence

Hình 23

A picture containing text

Description automatically generated

Hình 24

Bước cuối cùng là dẫn scrapy tìm đến dữ liệu muốn cạo xuống bằng xpath sau đó tiến hành lưu bằng lệnh yield. Chạy chương trình bằng lệnh “scrapy crawl X -o Y.Z” với X là tên file chương trình python tiến hành crawl, Y là tên file lưu dữ liệu và Z là định dạng của file Y bao gồm là file csv (Z=csv), file json (Z=json),…

Text

Description automatically generated with low confidence

Hình 25

Text

Description automatically generated

Hình 26

# Chương II: Phân tích dữ liệu bằng R

## 1.Mục tiêu

Phân tích dữ liệu bằng ngôn ngữ R, thực hiện các kiểm định và đưa ra những nhận định có ý nghĩa với tập dữ liệu

## 2. Phân tích dữ liệu

Trong phần này tôi sẽ tiến hành phân tích tập dữ liệu đã crawl được ở phần I bằng các phương pháp như phân tích thông kê mô tả, trực quan hóa dữ liệu và phân tích thống kê suy diễn.

### 2.1 Phân tích thống kê mô tả

Đầu tiên là chuyển cửa sổ R sang thư mục có lưu file dữ liệu, khai báo thư viện cần dùng và nhập dữ liệu vào R.

Graphical user interface, text, application, email

Description automatically generated

Hình 27

Xem thông tin dữ liệu và có một cái nhìn sơ bộ về tập dữ liệu bằng các hàm:

* Nrow(): xem số dòng của data
* Head(): Xem n số dòng đầu tiên của data
* Names(): Xem tên các feature data
* Str(): Xem kiểu dữ liệu của mỗi feature
* View(): Xem toàn bộ data

Background pattern

Description automatically generated

Hình 28

Graphical user interface, application

Description automatically generated

Hình 29

Graphical user interface, text, application, Word

Description automatically generated

Hình 30



Hình 31

Text

Description automatically generated

Hình 32

Tính các giá trị trung bình, trung vị, yếu vị của dữ liệu (đối với các feature là numeric). Sử dụng hàm ***“summary()”*** để khái quát toàn bộ dữ liệu. Hàm “table()” để xem được trong mỗi feature có bao nhiêu giá trị khác nhau

Graphical user interface, text, application, email

Description automatically generated

Hình 33

Graphical user interface, text, application, email

Description automatically generated

Hình 34

Graphical user interface, text, application

Description automatically generated

Hình 35

Graphical user interface, text, application

Description automatically generated

Hình 36

Tính các chỉ số tứ phân vị của các feature và chỉ số biểu thị mối tương quan giữa các biến numeric với nhau bằng các ***hàm “quantile(), cor()”*** . Đối với mức độ tương quan giữa các biến, giá trị nằm trong [-1;1]. Chỉ số có giá trị tuyệt đối càng lớn chứng tỏ chúng có mối liên hệ mạnh, với giá trị dương chúng có mối quan hệ đồng biến tức a tăng thì b tăng và ngược lại nếu âm thì chúng có mối quan hệ nghịch biến. Với giá trị bằng 0 tức hai biến độc lập.

Graphical user interface, text, application

Description automatically generated

Hình 37

Graphical user interface, text, application

Description automatically generated

Hình 38

### 2.2 Trực quan hóa dữ liệu

Biểu đồ 1: Biểu diễn số lượng xe đang bán ở mỗi hãng

Chart

Description automatically generated

Hình 39

Nhận xét:

* Biểu đồ là dạng cột ngang
* Biểu đồ biểu diễn số lượng xe đang được rao bán của mỗi hãng, trong đó:
  + Trục x biểu diễn tên của các hãng xe
  + Trục y biểu diễn số lượng xe
  + Nhỏ nhất là 1 xe đang được bán với 4 hãng (Volvo, Lada, Geely, Alfa Romeo)
  + Lớn nhất là hãng Toyota với số lượng 1325 chiếc đang được bán

Biểu đồ 2: Tương quan giữa năm sản xuất và giá xe

Text

Description automatically generated

Chart

Description automatically generated

Hình 40

Nhận xét:

* Biểu đồ dạng box plot
* Biểu đồ biểu diễn số mối quan hệ giữa giá xe và năm sản xuất của sảng phẩm, trong đó:
  + Trục x biểu diễn năm sản xuất của sản phẩm
  + Trục y biểu diễn giá tiền
  + Nhận thấy rằng, trong cùng một model xe (Toyota Innova) giá xe tăng khi năm sản xuất tăng, cụ thể:
    - Năm 2006, trung bình giá xe là 210 triệu (max: 280 triệu, min: 145 triệu)
    - Năm 2021, trung bình giá xe là 750 triệu (max: 887 triệu, min: 700 triệu)

Biểu đồ 3: Xe có tỉ lệ bán nhiều nhất

Text

Description automatically generated

Chart

Description automatically generated

Hình 41

Nhận xét:

* Biểu đồ dạng mật độ
* Biểu đồ biểu diễn tỉ lệ mỗi dòng xe đang được bán của hãng xe Porsche, trong đó:
  + Trục x biểu diễn năm sản xuất ra sản phẩm
  + Trục y biểu diễn tỉ lệ
  + Xét dòng xe Porsche Cayman:
    - Dòng xe này được rao bán từ năm 2015 cho đến nay
    - Từ 2015-2019, tỉ lệ rao bán của dòng xe này tăng liên tục
    - Vào năm 2019, dòng xe này có tỉ lệ rao bán lớn nhất (0.214)
    - Từ 2019-2021, tỉ lệ này giảm nhẹ và đến 2021 đạt chỉ số 0.168

Biểu đồ 4: Xe sang

A picture containing text

Description automatically generated

Chart, scatter chart

Description automatically generated

Hình 42

Nhận xét:

* Biều đồ dạng bóng bóng
* Biểu đồ biểu diễn các hãng xe có giá trên 4 tỉ đồng và các tỉnh thành đang bán những chiếc xe đó, trong đó:
  + Trục x biểu diễn các hãng có các dòng xe sang
  + Trục y biểu diễn giá tiền
  + Dòng xe Rolls-Royce là hãng xe có giá cao nhất (> 40 tỉ) đang được rao bán tại Hà Nội

### 2.3 Phân tích thống kê suy diễn

Đối với dữ liệu thực, rất khó để chúng có thể tuân theo phân phối chuẩn hay đồ thị của chúng có hình dạng úp chuông nên không thể thực hiện được các loại kiểm định T-test, hay ANOVA test để kiểm định giả thuyết thống kê. Do đó, tôi chuyển sang phân tích thống kê phi tham số

**2.3.1 Phân tích mối quan hệ tương quan**

Phân tích mối quan hệ tương quan hay tìm ra được giữa các feature chúng có phụ thuộc hay độc lập với nhau. Từ đó xác định sử dụng được những thống kê thích hợp.

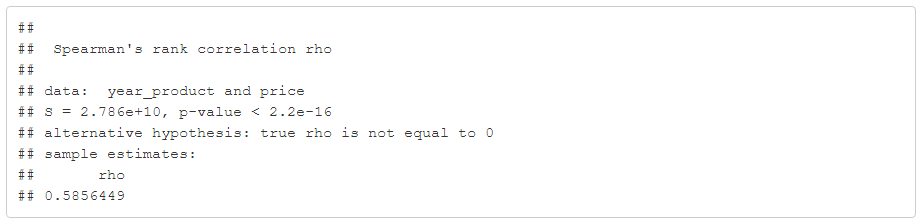
Hình 43 đang thực hiện kiểm định mối quan hệ giữa dòng xe (style) và giá xe (price) có phụ thuộc lẫn nhau hay không. Với giả thuyết rỗng là giữa 2 feature độc lập với nhau và đối thiết của nó là giữa 2 feature có mối quan hệ với nhau. Sử dụng hàm ***“cor.test()”*** để thực hiện kiểm định với các đối số của nó có dạng numeric. Một lưu ý là sử dụng thước đo là ***Spearman*** hoặc ***Kendall*** vì chúng ta đang áp dụng kiểm định với phân phối không chuẩn

Kết quả kiểm định nhận được với ***p\_value*** rất rất nhỏ nên tôi bác bỏ giả thiết H0 tức là giữa 2 feature giá xe và dòng xe là phụ thuộc lẫn nhau. Giá trị ***tau*** = 0.25 chứng tỏ mối quan hệ ở mức yếu



Hình 43

Tương tự tôi đã thực hiện để kiểm định mối quan hệ giữa năm sản xuất sản phẩm (year\_product) với giá xe (price), giữa KM và giá xe, giữa giá xe và nguồn gốc (origin). Kết quả lần lượt ở hình bên dưới (hình 44, 45, 46)



Hình 44



Hình 45

A picture containing scatter chart

Description automatically generated

Hình 46

**2.3.2 Kiểm định giả thuyết**

Có người cho rằng giá xe (price) của mỗi dòng xe (style) là khác nhau. Để kiểm tra xem nhận định trên có đúng hay không. Tôi sẽ thực hiện một bài kiểm tra để xác định tính đúng sai của nhận định. Nhắc lại dữ liệu trên không tuân theo phân phối chuẩn nên tôi sử dụng kiểm định ***“Wilcoxon Signed Rank Test”*** một kiểm định thay thế cho kiểm định T. Dùng hàm ***tapply()*** với đối số ***median*** để xem trung vị của các nhóm.

Graphical user interface, text, application

Description automatically generated

Hình 47

Sau đó thực hiện kiểm định bằng hàm ***wilcox.test()***. Kết quả nhận được với p\_value rất nhỏ, điều này ngụ ý rằng chúng ta bác bỏ giả thuyết H0 là các dòng xe không có sự khác biệt về giá và chấp nhận đối thiết H1.

Graphical user interface, text, application, email

Description automatically generated

Hình 48

Ngoài ra, chúng ta có thể sử dụng nhiều loại kiểm định phi tham số khác như kiểm định Chi2, kiểm định Kruskal Wallis để kiểm tra nhận định có đúng hay không. Hình 49 biểu diễn một bài kiểm định Chi2 để kiểm tra giữa xe nhập khẩu và xe lắp ráp trong nước liệu có khác biệt hay không. Cụ thể, trong nhóm xe Toyota, tôi lấy một dòng xe là Innova để thực hiện kiểm định. Với hàm ***chisq.test()*** và ***p\_value*** = 0.2. Tôi đưa ra kết luận rằng giữa xe nhập khẩu và xe lắp ráp trong nước không có sự khác biệt về giá.

Graphical user interface, text, application, email

Description automatically generated

Hình 49

# Chương III: Kết quả kiểm tra đạo văn