

CHƯƠNG I: KHÁI NIỆM CHUNG.....	3
1.1 Học máy với Hướng dẫn Python.....	3
1.2 Học máy với Python - Khái niệm cơ bản	3
1.1.1 Học máy là gì?	3
1.1.2 Cần cho Học máy.....	4
1.1.3 Tại sao & Khi nào nên Học Máy móc	4
1.3 Mô hình học máy.....	5
1.3.1 Nhiệm vụ (T)	5
1.3.2 Kinh nghiệm (E)	6
1.3.3 Hiệu suất (P)	6
1.4 Những thách thức trong học máy	6
1.5 Các ứng dụng của Học máy	7
CHƯƠNG II: HỌC MÁY VỚI PYTHON – HỆ SINH THÁI.....	7
2.1 Giới thiệu về Python.....	7
2.2 Điểm mạnh và điểm yếu của Python.....	8
2.3 Cài đặt Python	9
2.4 Tại sao sử dụng Python cho Khoa học Dữ liệu?.....	11
2.5 Các thành phần của Hệ sinh thái Python ML	11
Máy tính xách tay Jupyter	11
Các loại ô trong Máy tính xách tay Jupyter.....	13
2.6 Học máy với Python - Phương pháp	17
Các loại phương pháp khác nhau	17
Các nhiệm vụ phù hợp với học máy	21
2.7 Học máy với Python - Chuẩn bị dữ liệu.....	23
2.7.1 Giới thiệu.....	23
2.7.2 Tại sao phải xử lý trước dữ liệu?	23
2.7.3 Kỹ thuật xử lý trước dữ liệu	23
2.7.4 Các loại chuẩn hóa.....	25
2.7.5 Binarization	27
2.7.6 Tiêu chuẩn hóa.....	28
2.7.7 ML với Python - Lựa chọn tính năng dữ liệu.....	29

CHƯƠNG III: ỨNG DỤNG MACHINE LEARNING TRONG PHÂN TÍCH VÀ DỰ BÁO LÚN CÔNG TRÌNH	31
3.1 Các mô hình ML	31
3.1.1 Thuật toán ML - Phân loại	31
3.1.2 Thuật toán ML - Phân cụm.....	31
3.1.3 Thuật toán ML - Thuật toán KNN.....	31
3.1.4 Thuật toán ML - Hồi quy	31
3.2 Lựa chọn mô hình ML trong phân tích kết quả dự báo lún.....	31
3.3 Các ứng dụng của thuật toán hồi quy ML như sau:	35
3.4 Thực nghiệm trên số liệu thực tế:.....	35

CHƯƠNG I: KHÁI NIỆM CHUNG

1.1 Học máy với Hướng dẫn Python

Machine Learning (ML) về cơ bản là lĩnh vực khoa học máy tính với sự trợ giúp của hệ thống máy tính có thể cung cấp cảm giác cho dữ liệu theo cách giống như cách con người làm. Nói cách đơn giản, ML là một loại trí tuệ nhân tạo trích xuất các mẫu từ dữ liệu thô bằng cách sử dụng một thuật toán hoặc phương pháp. Trọng tâm chính của ML là cho phép các hệ thống máy tính học hỏi kinh nghiệm mà không cần được lập trình rõ ràng hoặc có sự can thiệp của con người.

Hướng dẫn này sẽ hữu ích cho sinh viên tốt nghiệp, sinh viên sau đại học và sinh viên nghiên cứu, những người có hứng thú với chủ đề này hoặc có chủ đề này như một phần của chương trình giảng dạy của họ. Người đọc có thể là người mới bắt đầu hoặc người học nâng cao. Hướng dẫn này đã được chuẩn bị cho sinh viên cũng như các chuyên gia để tăng tốc độ nhanh chóng. Hướng dẫn này là bước đệm cho hành trình Học máy của bạn.

Người đọc phải có kiến thức cơ bản về trí tuệ nhân tạo. Anh / cô ấy cũng nên biết Python, NumPy, Scikit-learning, Scipy, Matplotlib. Nếu bạn chưa quen với bất kỳ khái niệm nào trong số này, chúng tôi khuyên bạn nên xem các hướng dẫn liên quan đến các chủ đề này, trước khi bạn tìm hiểu sâu hơn về hướng dẫn này.

1.2 Học máy với Python - Khái niệm cơ bản

Chúng ta đang sống trong 'thời đại dữ liệu' được làm giàu với sức mạnh tính toán tốt hơn và nhiều tài nguyên lưu trữ hơn,. Dữ liệu hoặc thông tin này đang tăng lên từng ngày, nhưng thách thức thực sự là hiểu được tất cả dữ liệu. Các doanh nghiệp và tổ chức đang cố gắng giải quyết vấn đề này bằng cách xây dựng các hệ thống thông minh sử dụng các khái niệm và phương pháp luận từ Khoa học dữ liệu, Khai thác dữ liệu và Máy học. Trong số đó, học máy là lĩnh vực khoa học máy tính thú vị nhất. Sẽ không sai nếu chúng ta gọi học máy là ứng dụng và khoa học của các thuật toán mang lại cảm giác cho dữ liệu.

1.1.1 Học máy là gì?

Học máy (ML) là lĩnh vực khoa học máy tính với sự trợ giúp của hệ thống máy tính có thể cung cấp cảm giác cho dữ liệu theo cách giống như cách con người làm.

Nói cách đơn giản, ML là một loại trí tuệ nhân tạo trích xuất các mẫu ra khỏi dữ liệu thô bằng cách sử dụng một thuật toán hoặc phương pháp. Trọng tâm chính của ML là cho phép các hệ thống máy tính học hỏi kinh nghiệm mà không cần được lập trình rõ ràng hoặc có sự can thiệp của con người.

1.1.2 Cần cho Học máy

Con người, tại thời điểm này, là loài thông minh và tiên tiến nhất trên trái đất vì họ có thể suy nghĩ, đánh giá và giải quyết các vấn đề phức tạp. Mặt khác, AI vẫn đang trong giai đoạn sơ khai và chưa vượt qua trí thông minh của con người về nhiều mặt. Sau đó, câu hỏi đặt ra là cần làm cho máy học là gì? Lý do phù hợp nhất để làm điều này là “đưa ra quyết định, dựa trên dữ liệu, với hiệu quả và quy mô”.

Gần đây, các tổ chức đang đầu tư mạnh mẽ vào các công nghệ mới hơn như Trí tuệ nhân tạo, Học máy và Học sâu để lấy thông tin quan trọng từ dữ liệu nhằm thực hiện một số nhiệm vụ trong thế giới thực và giải quyết vấn đề. Chúng ta có thể gọi đó là các quyết định dựa trên dữ liệu do máy móc thực hiện, đặc biệt là để tự động hóa quy trình. Những quyết định theo hướng dữ liệu này có thể được sử dụng, thay vì sử dụng logic lập trình, trong các vấn đề vốn dĩ không thể lập trình được. Thực tế là chúng ta không thể làm được nếu không có trí thông minh của con người, nhưng khía cạnh khác là tất cả chúng ta cần giải quyết các vấn đề trong thế giới thực với hiệu quả ở quy mô lớn. Đó là lý do tại sao nhu cầu về học máy phát sinh.

1.1.3 Tại sao & Khi nào nên Học Máy móc

Chúng ta đã thảo luận về sự cần thiết của việc học máy, nhưng một câu hỏi khác đặt ra là chúng ta phải làm cho máy học trong những tình huống nào? Có thể có một số trường hợp mà chúng ta cần máy móc để đưa ra các quyết định dựa trên dữ liệu với hiệu quả và ở quy mô lớn. Sau đây là một số trường hợp như vậy mà việc chế tạo máy học sẽ hiệu quả hơn.

a. *Thiếu chuyên môn của con người*

Kịch bản đầu tiên trong đó chúng ta muốn một cỗ máy học hỏi và đưa ra các quyết định theo hướng dữ liệu, có thể là lĩnh vực thiếu chuyên môn của con người. Các ví dụ có thể là điều hướng trong các lãnh thổ hoặc hành tinh không gian không xác định.

b. *1.3.2 Các tình huống động*

Có một số kịch bản có bản chất động tức là chúng liên tục thay đổi theo thời gian. Trong trường hợp của những tình huống và hành vi này, chúng tôi muốn một cỗ máy học hỏi và đưa ra các quyết định theo hướng dữ liệu. Một số ví dụ có thể là kết nối mạng và tính sẵn có của cơ sở hạ tầng trong một tổ chức.

c. *1.3.3 Khó khăn trong việc chuyển đổi kiến thức chuyên môn thành các nhiệm vụ tính toán*

Có thể có nhiều lĩnh vực khác nhau mà con người có chuyên môn của họ ; tuy nhiên, họ không thể chuyển kiến thức chuyên môn này thành các nhiệm vụ tính toán. Trong những trường hợp như vậy, chúng tôi muốn học máy. Các ví dụ có thể là các lĩnh vực nhận dạng giọng nói, các nhiệm vụ nhận thức, v.v.

1.3 Mô hình học máy

Trước khi thảo luận về mô hình học máy, chúng ta phải hiểu định nghĩa chính thức sau đây của ML do giáo sư Mitchell đưa ra:

“Một chương trình máy tính được cho là học hỏi từ trải nghiệm E đối với một số loại nhiệm vụ T và thước đo hiệu suất P, nếu hiệu suất của nó ở các nhiệm vụ trong T, được đo bằng P, cải thiện theo kinh nghiệm E.”

Định nghĩa trên về cơ bản tập trung vào ba tham số, cũng là thành phần chính của bất kỳ thuật toán học tập nào, đó là Nhiệm vụ (T), Hiệu suất (P) và kinh nghiệm (E). Trong ngữ cảnh này, chúng ta có thể đơn giản hóa định nghĩa này là:

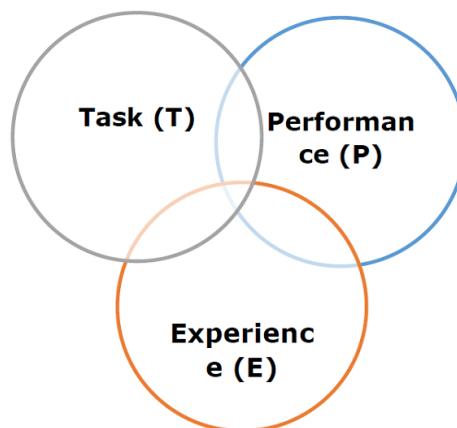
ML là một lĩnh vực AI bao gồm các thuật toán học tập -

Cải thiện hiệu suất của họ (P)

Khi thực hiện một số nhiệm vụ (T)

Theo thời gian với kinh nghiệm (E)

Dựa trên những điều trên, sơ đồ sau đại diện cho Mô hình Học máy -



Hãy để chúng tôi thảo luận chi tiết hơn về chúng ngay bây giờ.

1.3.1 Nhiệm vụ (T)

Từ góc độ vấn đề, chúng ta có thể định nghĩa nhiệm vụ T là vấn đề trong thế giới thực cần giải quyết. Vấn đề có thể là bất cứ điều gì như tìm giá nhà tốt nhất ở một vị trí cụ thể hoặc tìm chiến lược tiếp thị tốt nhất, v.v. Mặt khác, nếu chúng ta nói về máy học, định nghĩa của nhiệm vụ là khác bởi vì rất khó để giải quyết các nhiệm vụ dựa trên ML bằng cách tiếp cận lập trình thông thường.

Nhiệm vụ T được cho là một nhiệm vụ dựa trên ML khi nó dựa trên quy trình và hệ thống phải tuân theo để hoạt động trên các điểm dữ liệu. Ví dụ về các nhiệm vụ dựa trên ML là Phân loại, Hồi quy, Chú thích có cấu trúc, Phân cụm, Phiên mã, v.v.

1.3.2 Kinh nghiệm (E)

Như tên cho thấy, nó là kiến thức thu được từ các điểm dữ liệu được cung cấp cho thuật toán hoặc mô hình. Sau khi được cung cấp tập dữ liệu, mô hình sẽ chạy lặp đi lặp lại và sẽ học một số mẫu vốn có. Do đó, sự học hỏi có được gọi là kinh nghiệm (E). Tương tự với việc học tập của con người, chúng ta có thể nghĩ về tình huống này như trong đó một con người đang học hỏi hoặc đạt được một số kinh nghiệm từ các thuộc tính khác nhau như tình huống, mối quan hệ, v.v. Học tập có giám sát, không giám sát và tăng cường là một số cách để học hỏi hoặc tích lũy kinh nghiệm. Kinh nghiệm thu được từ mô hình ML hoặc thuật toán sẽ được sử dụng để giải quyết nhiệm vụ T.

1.3.3 Hiệu suất (P)

Một thuật toán ML được cho là thực hiện nhiệm vụ và tích lũy kinh nghiệm theo thời gian. Thước đo cho biết thuật toán ML có hoạt động như mong đợi hay không là hiệu suất của nó (P). P về cơ bản là một số liệu định lượng cho biết cách một mô hình đang thực hiện nhiệm vụ, T, sử dụng kinh nghiệm của nó, E. Có nhiều số liệu giúp hiểu hiệu suất ML, chẳng hạn như điểm chính xác, điểm F1, ma trận nhầm lẫn, độ chính xác, thu hồi, độ nhạy, v.v.

1.4 Những thách thức trong học máy

Trong khi Học máy đang phát triển nhanh chóng, tạo ra những bước tiến đáng kể với an ninh mạng và ô tô tự hành, thì toàn bộ mảng AI này vẫn còn một chặng đường dài phía trước. Lý do đằng sau là ML đã không thể vượt qua một số thử thách. Những thách thức mà ML đang phải đối mặt hiện tại là:

Quality of data- Có dữ liệu chất lượng tốt cho các thuật toán ML là một trong những thách thức lớn nhất. Việc sử dụng dữ liệu chất lượng thấp dẫn đến các vấn đề liên quan đến tiền xử lý dữ liệu và trích xuất tính năng.

Time-Consuming task - Một thách thức khác mà các mô hình ML phải đối mặt là tiêu tốn thời gian đặc biệt cho việc thu thập dữ liệu, trích xuất và truy xuất tính năng.

Lack of specialist persons - Vì công nghệ ML vẫn còn trong giai đoạn sơ khai, nên việc có sẵn nguồn lực chuyên gia là một công việc khó khăn.

No clear objective for formulating business problems - Không có mục tiêu rõ ràng và mục tiêu được xác định rõ ràng cho các vấn đề kinh doanh là một thách thức quan trọng khác đối với ML vì công nghệ này vẫn chưa trưởng thành.

Issue of overfitting & underfitting - Nếu mô hình được trang bị quá mức hoặc thiếu trang bị, nó không thể được đại diện tốt cho sự cố.

Curse of dimensionality- Một thách thức khác mà mô hình ML gặp phải là quá nhiều tính năng của các điểm dữ liệu. Đây có thể là một trở ngại thực sự.

Difficulty in deployment - Sự phức tạp của mô hình ML khiến nó khá khó triển khai trong cuộc sống thực.

1.5 Các ứng dụng của Học máy

Học máy là công nghệ phát triển nhanh chóng nhất và theo các nhà nghiên cứu, chúng ta đang ở trong năm vàng của AI và ML. Nó được sử dụng để giải quyết nhiều vấn đề phức tạp trong thế giới thực mà không thể giải quyết bằng phương pháp truyền thống. Sau đây là một số ứng dụng trong thế giới thực của ML -

- Phân tích cảm xúc
- Phân tích tình cảm
- Phát hiện và ngăn ngừa lỗi
- Dự báo thời tiết và dự đoán
- Phân tích và dự báo thị trường chứng khoán
- Tổng hợp giọng nói
- Nhận dạng giọng nói
- Phân khúc khách hàng
- Nhận dạng đối tượng
- Phát hiện gian lận
- Phòng chống gian lận
- Giới thiệu sản phẩm cho khách hàng khi mua sắm trực tuyến.

CHƯƠNG II: HỌC MÁY VỚI PYTHON – HỆ SINH THÁI

2.1 Giới thiệu về Python

Python là một ngôn ngữ lập trình hướng đối tượng phổ biến có các tính năng của ngôn ngữ lập trình cấp cao. Cú pháp dễ học và khả năng di động của nó khiến nó trở nên phổ biến ngày nay. Các dữ kiện sau đây cho chúng ta sự giới thiệu về Python:

- Python được phát triển bởi Guido van Rossum tại Stichting Mathematisch Centrum ở Hà Lan.
- Nó được viết như là sự kế thừa của ngôn ngữ lập trình có tên là 'ABC'.
- Phiên bản đầu tiên được phát hành vào năm 1991.

- Cái tên Python do Guido van Rossum chọn từ một chương trình truyền hình có tên Monty Python's Flying Circus.
- Nó là một ngôn ngữ lập trình mã nguồn mở, có nghĩa là chúng ta có thể tự do tải xuống và sử dụng nó để phát triển các chương trình. Nó có thể được tải xuống từ www.python.org.
- Ngôn ngữ lập trình Python có các tính năng của cả Java và C. Nó có mã 'C' thanh lịch và mặt khác, nó có các lớp và đối tượng như Java để lập trình hướng đối tượng.
- Nó là một ngôn ngữ thông dịch, có nghĩa là mã nguồn của chương trình Python trước tiên sẽ được chuyển đổi thành bytecode và sau đó được thực thi bởi máy ảo Python.

2.2 Điểm mạnh và điểm yếu của Python

Mỗi ngôn ngữ lập trình đều có một số điểm mạnh cũng như điểm yếu, Python cũng vậy.

2.2.1 Điểm mạnh

Theo các nghiên cứu và khảo sát, Python là ngôn ngữ quan trọng thứ năm cũng như là ngôn ngữ phổ biến nhất cho học máy và khoa học dữ liệu. Chính vì những điểm mạnh sau đây mà Python có:

Easy to learn and understand- Cú pháp của Python đơn giản hơn; do đó nó tương đối dễ dàng, ngay cả đối với người mới bắt đầu, học và hiểu ngôn ngữ.

Multi-purpose language - Python là một ngôn ngữ lập trình đa mục đích vì nó hỗ trợ lập trình cấu trúc, lập trình hướng đối tượng cũng như lập trình chức năng.

Huge number of modules- Python có số lượng lớn các mô-đun để bao gồm mọi khía cạnh của lập trình. Các mô-đun này dễ dàng có sẵn để sử dụng do đó làm cho Python trở thành một ngôn ngữ có thể mở rộng.

Support of open source community- Là ngôn ngữ lập trình mã nguồn mở, Python được hỗ trợ bởi một cộng đồng nhà phát triển rất lớn. Do đó, cộng đồng Python dễ dàng sửa lỗi. Đặc điểm này làm cho Python rất mạnh mẽ và thích nghi.

Scalability - Python là một ngôn ngữ lập trình có thể mở rộng vì nó cung cấp một cấu trúc cải tiến để hỗ trợ các chương trình lớn hơn so với các kịch bản lệnh shell.

2.2.2 Điểm yếu

Mặc dù Python là một ngôn ngữ lập trình phổ biến và mạnh mẽ nhưng nó có điểm yếu là tốc độ thực thi chậm.

Tốc độ thực thi của Python chậm hơn so với các ngôn ngữ biên dịch vì Python là ngôn ngữ thông dịch. Đây có thể là lĩnh vực cải tiến chính của cộng đồng Python.

2.3 Cài đặt Python

Để làm việc bằng Python, trước tiên chúng ta phải cài đặt nó. Bạn có thể thực hiện cài đặt Python theo bất kỳ cách nào trong hai cách sau:

- Cài đặt Python riêng lẻ
- Sử dụng phân phối Python được đóng gói sẵn - Anaconda

Hãy để chúng tôi thảo luận chi tiết từng vấn đề này.

Cài đặt Python riêng lẻ

Nếu bạn muốn cài đặt Python trên máy tính của mình, thì bạn chỉ cần tải xuống mã nhị phân áp dụng cho nền tảng của mình. Bản phân phối Python có sẵn cho các nền tảng Windows, Linux và Mac.

Sau đây là tổng quan nhanh về cài đặt Python trên các nền tảng nêu trên:

On Unix and Linux platform

Với sự trợ giúp của các bước sau, chúng ta có thể cài đặt Python trên nền tảng Unix và Linux -

- Đầu tiên, hãy truy cập www.python.org/downloads/ .
- Tiếp theo, nhấp vào liên kết để tải xuống mã nguồn nén có sẵn cho Unix / Linux.
- Bây giờ, Tải xuống và giải nén tệp.
- Tiếp theo, chúng ta có thể chỉnh sửa tệp Modules / Setup nếu chúng ta muốn tùy chỉnh một số tùy chọn.
 - Tiếp theo, viết lệnh **run ./configure script**
 - make
 - thực hiện cài đặt

On Windows platform

Với sự trợ giúp của các bước sau, chúng ta có thể cài đặt Python trên nền tảng Windows Đầu tiên, hãy truy cập www.python.org/downloads/ .

- Tiếp theo, nhấp vào liên kết cho tệp python-XYZ.msi của trình cài đặt Windows. Ở đây XYZ là phiên bản chúng tôi muốn cài đặt.
- Bây giờ, chúng ta phải chạy tệp được tải xuống. Nó sẽ đưa chúng ta đến trình hướng dẫn cài đặt Python, rất dễ sử dụng. Bây giờ, hãy chấp nhận cài đặt mặc định và đợi cho đến khi quá trình cài đặt hoàn tất.

On Macintosh platform

Đối với Mac OS X, Homebrew, một trình cài đặt gói tuyệt vời và dễ sử dụng được khuyến khích để cài đặt Python 3. Trong trường hợp không có Homebrew, bạn có thể cài đặt nó với sự trợ giúp của lệnh sau:

```
$ ruby -e "$(curl -fsSL
```

```
https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Nó có thể được cập nhật bằng lệnh dưới đây:

```
$ brew update
```

Bây giờ, để cài đặt Python3 trên hệ thống của bạn, chúng ta cần chạy lệnh sau:

```
$ brew install python3
```

Sử dụng phân phối Python được đóng gói sẵn: Anaconda

Anaconda là một bản biên dịch đóng gói của Python có tất cả các thư viện được sử dụng rộng rãi trong Khoa học dữ liệu. Chúng ta có thể làm theo các bước sau để thiết lập môi trường Python bằng Anaconda:

- **Step 1-** Đầu tiên, chúng ta cần tải gói cài đặt yêu cầu từ bản phân phối Anaconda. Liên kết tương tự là www.anaconda.com/distribution/ . Bạn có thể chọn từ HĐH Windows, Mac và Linux theo yêu cầu của mình.
- **Step 2-** Tiếp theo, chọn phiên bản Python mà bạn muốn cài đặt trên máy của mình. Phiên bản Python mới nhất là 3.7. Ở đó, bạn sẽ nhận được các tùy chọn cho cả hai trình cài đặt Đồ họa 64-bit và 32-bit.
- **Step 3-** Sau khi chọn hệ điều hành và phiên bản Python, nó sẽ tải xuống trình cài đặt Anaconda trên máy tính của bạn. Bây giờ, nhấp đúp vào tệp và trình cài đặt sẽ cài đặt gói Anaconda.
- **Step 4 -** Để kiểm tra xem nó đã được cài đặt hay chưa, hãy mở dấu nhắc lệnh và nhập Python như sau:

A screenshot of a Windows Command Prompt window titled "Command Prompt - Python". The window shows the following text: "Microsoft Windows [Version 10.0.17134.590] (c) 2018 Microsoft Corporation. All rights reserved. C:\Users\Leekha>cd Anaconda3 C:\Users\Leekha\Anaconda3>Python Python 3.7.1 (default, Dec 10 2018, 22:54:23) [MSC v.1915 64 bit (AMD64)] :: Anaconda, Inc. on win32 Type "help", "copyright", "credits" or "license" for more information. >>>". The window has a standard Windows title bar with minimize, maximize, and close buttons.

Bạn cũng có thể kiểm tra điều này trong video bài giảng chi tiết tại

2.4 Tại sao sử dụng Python cho Khoa học Dữ liệu?

Python là ngôn ngữ quan trọng thứ năm cũng như ngôn ngữ phổ biến nhất cho Máy học và khoa học dữ liệu. Sau đây là các tính năng của Python khiến nó trở thành lựa chọn ngôn ngữ ưu tiên cho khoa học dữ liệu.

Bộ gói mở rộng

Python có một bộ gói mở rộng và mạnh mẽ sẵn sàng được sử dụng trong các miền khác nhau. Nó cũng có các gói như **numpy**, **scipy**, **pandas**, **scikit-learn** v.v ... cần thiết cho học máy và khoa học dữ liệu.

Tạo mẫu dễ dàng

Một tính năng quan trọng khác của Python khiến nó trở thành ngôn ngữ được lựa chọn cho khoa học dữ liệu là tạo mẫu dễ dàng và nhanh chóng. Tính năng này rất hữu ích cho việc phát triển thuật toán mới.

Tính năng cộng tác

Lĩnh vực khoa học dữ liệu về cơ bản cần sự hợp tác tốt và Python cung cấp nhiều công cụ hữu ích giúp làm được điều này.

Một ngôn ngữ cho nhiều miền

Một dự án khoa học dữ liệu điển hình bao gồm các lĩnh vực khác nhau như trích xuất dữ liệu, thao tác dữ liệu, phân tích dữ liệu, trích xuất tính năng, mô hình hóa, đánh giá, triển khai và cập nhật giải pháp. Vì Python là một ngôn ngữ đa mục đích, nó cho phép nhà khoa học dữ liệu giải quyết tất cả các miền này từ một nền tảng chung.

2.5 Các thành phần của Hệ sinh thái Python ML

Trong phần này, chúng ta hãy thảo luận về một số thư viện Khoa học Dữ liệu cốt lõi hình thành các thành phần của hệ sinh thái Máy học Python. Những thành phần hữu ích này làm cho Python trở thành một ngôn ngữ quan trọng cho Khoa học dữ liệu. Mặc dù có nhiều thành phần như vậy, chúng ta hãy thảo luận một số thành phần quan trọng của hệ sinh thái Python tại đây -

Máy tính xách tay Jupyter

Máy tính xách tay Jupyter về cơ bản cung cấp một môi trường tính toán tương tác để phát triển các ứng dụng Khoa học dữ liệu dựa trên Python. Chúng trước đây được gọi là máy tính xách tay ipython. Sau đây là một số tính năng của sổ ghi chép Jupyter khiến nó trở thành một trong những thành phần tốt nhất của hệ sinh thái Python ML:

- Máy tính xách tay Jupyter có thể minh họa quá trình phân tích từng bước bằng cách sắp xếp các nội dung như mã, hình ảnh, văn bản, đầu ra, v.v. theo cách từng bước.

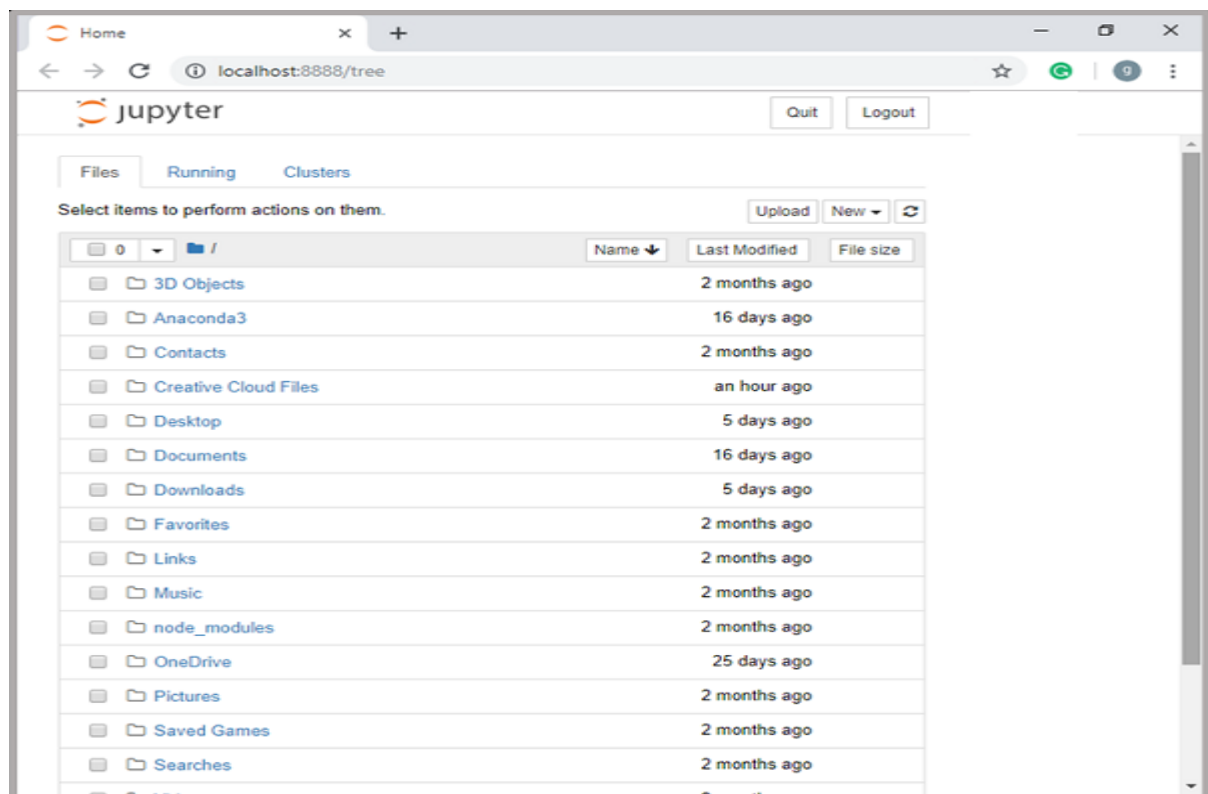
- Nó giúp một nhà khoa học dữ liệu ghi lại quá trình suy nghĩ trong khi phát triển quá trình phân tích.
- Người ta cũng có thể ghi lại kết quả như một phần của sổ ghi chép.
- Với sự trợ giúp của sổ ghi chép jupyter, chúng ta cũng có thể chia sẻ công việc của mình với một người ngang hàng.

Cài đặt và Thực hiện

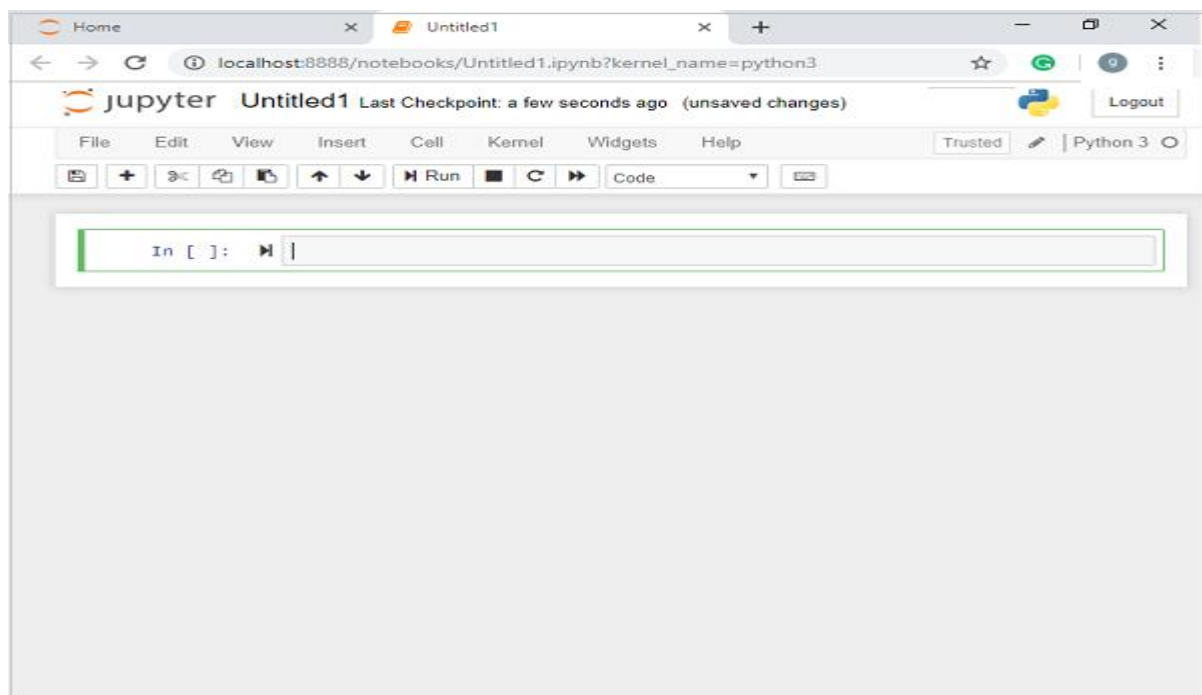
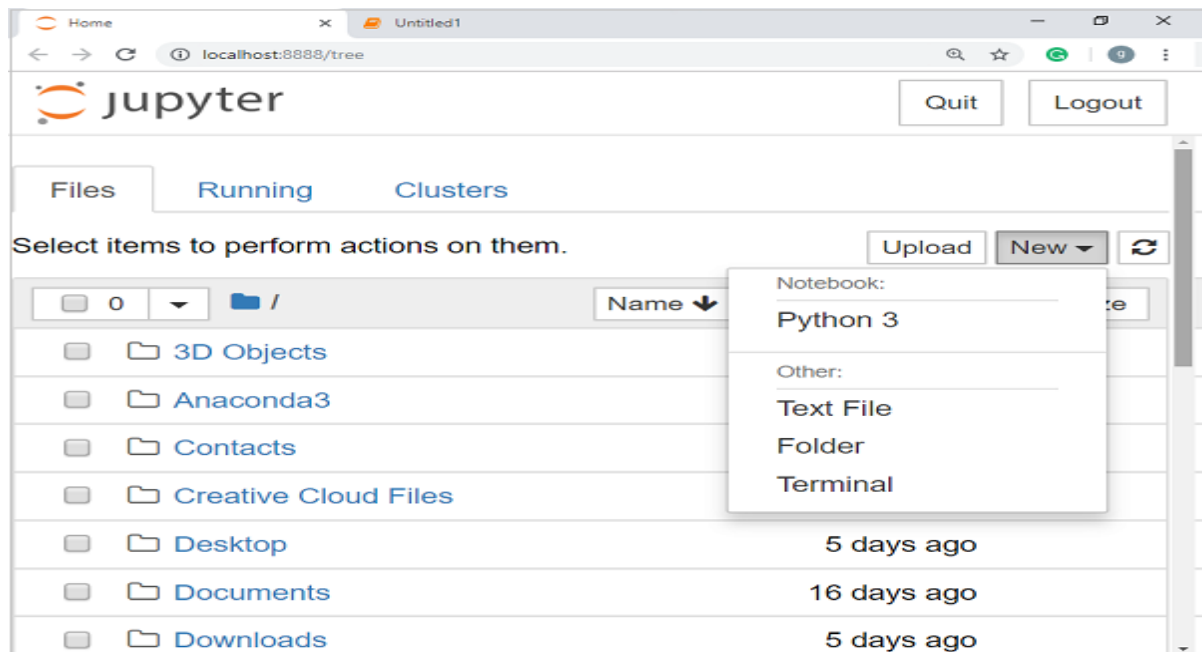
Nếu bạn đang sử dụng phân phối Anaconda, thì bạn không cần cài đặt riêng sổ ghi chép jupyter vì nó đã được cài đặt với nó. Bạn chỉ cần vào Anaconda Prompt và gõ lệnh sau:

```
C:\>jupyter notebook
```

Sau khi nhấn enter, nó sẽ khởi động một máy chủ sổ ghi chép tại localhost: 8888 của máy tính của bạn. Nó được hiển thị trong ảnh chụp màn hình sau:



Bây giờ, sau khi nhấp vào tab Mới, bạn sẽ nhận được danh sách các tùy chọn. Chọn Python 3 và nó sẽ đưa bạn đến sổ ghi chép mới để bắt đầu làm việc với nó. Bạn sẽ nhìn thấy nó trong các ảnh chụp màn hình sau:



Mặt khác, nếu bạn đang sử dụng phân phối Python chuẩn thì sở ghi chép jupyter có thể được cài đặt bằng trình cài đặt gói python phổ biến, pip.

```
pip install jupyter
```

Các loại ô trong Máy tính xách tay Jupyter

Sau đây là ba loại ô trong sở ghi chép jupyter -

Code cells- Đúng như tên gọi, chúng ta có thể sử dụng các ô này để viết mã. Sau khi viết mã / nội dung, nó sẽ gửi nó đến hạt nhân được liên kết với sở ghi chép.

Markdown cells- Chúng ta có thể sử dụng các ô này để ghi chú quá trình tính toán. Chúng có thể chứa những thứ như văn bản, hình ảnh, phương trình Latex, thẻ HTML, v.v.

Raw cells- Văn bản được viết trong chúng được hiển thị như nó vốn có. Các ô này về cơ bản được sử dụng để thêm văn bản mà chúng tôi không muốn được chuyển đổi bằng cơ chế chuyển đổi tự động của jupyter notebook.

Để nghiên cứu chi tiết hơn về sổ ghi chép jupyter, bạn có thể truy cập liên kết www.tutorialspoint.com/jupyter/index.htm.

NumPy

Đó là một thành phần hữu ích khác khiến Python trở thành một trong những ngôn ngữ yêu thích cho Khoa học dữ liệu. Về cơ bản, nó là viết tắt của Numerical Python và bao gồm các đối tượng mảng đa chiều. Bằng cách sử dụng NumPy, chúng ta có thể thực hiện các thao tác quan trọng sau:

- Các phép toán toán học và logic trên mảng.
- Biến đổi Fourier
- Các phép toán kết hợp với đại số tuyến tính.

Chúng ta cũng có thể xem NumPy là sự thay thế của MatLab vì NumPy chủ yếu được sử dụng cùng với Scipy (Python khoa học) và Matplotlib (thư viện vẽ đồ thị).

Installation and Execution

Nếu bạn đang sử dụng phân phối Anaconda, thì không cần cài đặt riêng NumPy vì nó đã được cài đặt sẵn. Bạn chỉ cần nhập gói vào tập lệnh Python của mình với sự trợ giúp của các bước sau:

```
import numpy as np
```

Mặt khác, nếu bạn đang sử dụng phân phối Python chuẩn thì NumPy có thể được cài đặt bằng trình cài đặt gói python phổ biến, pip.

```
pip install NumPy
```

Để nghiên cứu chi tiết hơn về NumPy, bạn có thể truy cập liên kết www.tutorialspoint.com/numpy/index.htm.

Gấu trúc

Đó là một thư viện Python hữu ích khác làm cho Python trở thành một trong những ngôn ngữ yêu thích cho Khoa học dữ liệu. Gấu trúc về cơ bản được sử dụng để thao tác, quản và phân tích dữ liệu. Nó được phát triển bởi Wes McKinney vào năm 2008. Với sự trợ giúp của Pandas, trong quá trình xử lý dữ liệu, chúng ta có thể thực hiện năm bước sau:

- Load

- Prepare
- Manipulate
- Model
- Analyze

Biểu diễn dữ liệu trong Pandas

Toàn bộ biểu diễn dữ liệu trong Pandas được thực hiện với sự trợ giúp của ba cấu trúc dữ liệu sau:

Series- Về cơ bản nó là một ndarray một chiều với nhãn trục có nghĩa là nó giống như một mảng đơn giản với dữ liệu đồng nhất. Ví dụ, chuỗi sau là tập hợp các số nguyên 1,5,10,15,24,25

...

1	5	10	15	24	25	28	36	40
---	---	----	----	----	----	----	----	----

Data frame- Đây là cấu trúc dữ liệu hữu ích nhất và được sử dụng cho hầu hết các loại biểu diễn và thao tác dữ liệu ở gấu trúc. Về cơ bản, nó là một cấu trúc dữ liệu hai chiều có thể chứa dữ liệu không đồng nhất. Nói chung, dữ liệu dạng bảng được biểu diễn bằng cách sử dụng các khung dữ liệu. Ví dụ: bảng sau đây cho thấy dữ liệu về các học sinh có tên và số cuộn, tuổi và giới tính -

Tên	Số cuộn	Tuổi tác	Giới tính
Aarav	1	15	Nam giới
Nhâm nhí	2	14	Nam giới
Kanika	3	16	Giống cái
Mayank	4	15	Nam giới

Panel- Là cấu trúc dữ liệu 3 chiều chứa dữ liệu không đồng nhất. Rất khó để biểu diễn bảng điều khiển dưới dạng biểu diễn đồ họa, nhưng nó có thể được minh họa như một vùng chứa DataFrame.

Bảng sau đây cho chúng ta thứ nguyên và mô tả về cấu trúc dữ liệu được đề cập ở trên được sử dụng trong Pandas -

Cấu trúc dữ liệu	Kích thước	Sự miêu tả
------------------	------------	------------

Loại	1-D	Kích thước không thay đổi, dữ liệu đồng nhất 1-D
DataFrames	2-D	Kích thước có thể thay đổi, dữ liệu không đồng nhất ở dạng bảng
Bảng điều khiển	3-D	Mảng có thể thay đổi kích thước, vùng chứa DataFrame.

Chúng ta có thể hiểu các cấu trúc dữ liệu này là cấu trúc dữ liệu chiều cao hơn là nơi chứa cấu trúc dữ liệu chiều thấp hơn.

Cài đặt và Thực hiện

Nếu bạn đang sử dụng phân phối Anaconda, thì không cần cài đặt riêng Pandas vì nó đã được cài đặt sẵn. Bạn chỉ cần nhập gói vào tập lệnh Python của mình với sự trợ giúp của các bước sau:

```
import pandas as pd
```

Mặt khác, nếu bạn đang sử dụng phân phối Python chuẩn thì Pandas có thể được cài đặt bằng trình cài đặt gói python phổ biến, pip.

```
pip install Pandas
```

Sau khi cài đặt Pandas, bạn có thể nhập nó vào tập lệnh Python của mình như đã làm ở trên.

Thí dụ

Sau đây là một ví dụ về cách tạo một chuỗi từ ndarray bằng cách sử dụng Pandas:

```
In [1]: import pandas as pd
In [2]: import numpy as np
In [3]: data = np.array(['g', 'a', 'u', 'r', 'a', 'v'])
In [4]: s = pd.Series(data)
In [5]: print (s)

0 g
1 a
2 u
3 r
4 a
5 v
dtype: object
```

Để nghiên cứu chi tiết hơn về Cấu trúc, bạn có thể truy cập liên kết

www.tutorialspoint.com/python_pandas/index.htm .

Scikit-learning

Một thư viện python hữu ích và quan trọng nhất cho Khoa học dữ liệu và học máy bằng Python là Scikit-learning. Sau đây là một số tính năng của Scikit-learning giúp nó trở nên hữu ích -

- Nó được xây dựng trên NumPy, SciPy và Matplotlib.
- Nó là một mã nguồn mở và có thể được sử dụng lại theo giấy phép BSD.
- Nó có thể truy cập được cho tất cả mọi người và có thể được sử dụng lại trong các ngữ cảnh khác nhau.
- Một loạt các thuật toán học máy bao gồm các lĩnh vực chính của ML như phân loại, phân cụm, hồi quy, giảm kích thước, lựa chọn mô hình, v.v. có thể được thực hiện với sự trợ giúp của nó.

Cài đặt và Thực hiện

Nếu bạn đang sử dụng bản phân phối Anaconda, thì không cần cài đặt riêng Scikit-learning vì nó đã được cài đặt sẵn. Bạn chỉ cần sử dụng gói vào tập lệnh Python của mình. Ví dụ: với dòng tập lệnh sau, chúng tôi đang nhập tập dữ liệu về bệnh nhân ung thư vú từ **Scikit-learn** -

```
from sklearn.datasets import load_breast_cancer
```

Mặt khác, nếu bạn đang sử dụng phân phối Python chuẩn và có NumPy và SciPy thì Scikit-learning có thể được cài đặt bằng cách sử dụng trình cài đặt gói python phổ biến, pip.

```
pip install -U scikit-learn
```

Sau khi cài đặt Scikit-learning, bạn có thể sử dụng nó vào tập lệnh Python của mình như bạn đã làm ở trên.

2.6 Học máy với Python - Phương pháp

Có nhiều thuật toán, kỹ thuật và phương pháp ML khác nhau có thể được sử dụng để xây dựng các mô hình giải quyết các vấn đề trong cuộc sống thực bằng cách sử dụng dữ liệu. Trong chương này, chúng ta sẽ thảo luận về các loại phương pháp khác nhau.

Các loại phương pháp khác nhau

Sau đây là các phương pháp ML khác nhau dựa trên một số danh mục rộng:

2.6.1 Dựa trên sự giám sát của con người

Trong quá trình học, một số phương pháp dựa trên sự giám sát của con người như sau:

Supervised Learning

Các thuật toán hoặc phương pháp học có giám sát là các thuật toán ML được sử dụng phổ biến nhất. Phương pháp hoặc thuật toán học tập này lấy mẫu dữ liệu tức là dữ liệu đào tạo và kết quả đầu ra liên quan của nó, tức là nhãn hoặc phản hồi với mỗi mẫu dữ liệu trong quá trình đào tạo.

Mục tiêu chính của thuật toán học có giám sát là tìm hiểu mối liên hệ giữa các mẫu dữ liệu đầu vào và đầu ra tương ứng sau khi thực hiện nhiều phiên bản dữ liệu đào tạo.

Ví dụ, chúng tôi có

X: Các biến đầu vào và

Y: Biến đầu ra

Bây giờ, hãy áp dụng một thuật toán để học hàm ánh xạ từ đầu vào đến đầu ra như sau:

$$Y = f(x)$$

Bây giờ, mục tiêu chính là xấp xỉ hàm ánh xạ tốt đến mức ngay cả khi chúng ta có dữ liệu đầu vào mới (x), chúng ta có thể dễ dàng dự đoán biến đầu ra (Y) cho dữ liệu đầu vào mới đó.

Nó được gọi là có giám sát bởi vì toàn bộ quá trình học tập có thể được coi như nó đang được giám sát bởi một giáo viên hoặc người giám sát. Ví dụ về thuật toán học máy được giám sát bao gồm **Decision tree, Random Forest, KNN, Logistic Regression** Vân vân.

Dựa trên các nhiệm vụ ML, các thuật toán học tập có giám sát có thể được chia thành hai lớp sau:

- Classification
- Regression

Classification

Mục tiêu chính của các nhiệm vụ dựa trên phân loại là dự đoán các nhãn đầu ra phân loại hoặc phản hồi cho dữ liệu đầu vào nhất định. Đầu ra sẽ dựa trên những gì mô hình đã học được trong giai đoạn đào tạo. Như chúng ta biết rằng các phản hồi đầu ra phân loại có nghĩa là các giá trị không có thứ tự và rời rạc, do đó mỗi phản hồi đầu ra sẽ thuộc về một lớp hoặc danh mục cụ thể. Chúng ta cũng sẽ thảo luận chi tiết về Phân loại và các thuật toán liên quan trong các chương sắp tới.

Regression

Mục tiêu chính của các tác vụ dựa trên hồi quy là dự đoán các nhãn hoặc phản hồi đầu ra là các giá trị số tiếp tục, cho dữ liệu đầu vào đã cho. Đầu ra sẽ dựa trên những gì mô hình đã học được trong giai đoạn đào tạo của nó. Về cơ bản, các mô hình hồi quy sử dụng các đặc trưng dữ liệu đầu vào (các biến độc lập) và các giá trị đầu ra số liên tục tương ứng của chúng (biến phụ thuộc hoặc kết quả) để tìm hiểu mối liên hệ cụ thể giữa đầu vào và đầu ra tương ứng. Chúng tôi cũng sẽ thảo luận chi tiết về hồi quy và các thuật toán liên quan trong các chương tiếp theo.

2.6.1.1 Học tập không giám sát

Như tên cho thấy, nó đối lập với các phương pháp hoặc thuật toán ML có giám sát, có nghĩa là trong các thuật toán học máy không được giám sát, chúng tôi không có bất kỳ người giám sát nào để cung cấp bất kỳ loại hướng dẫn nào. Các thuật toán học tập không giám sát rất hữu ích trong trường hợp chúng ta không có quyền tự do, giống như trong các thuật toán học tập có giám sát, có dữ liệu đào tạo được gắn nhãn trước và chúng ta muốn trích xuất mẫu hữu ích từ dữ liệu đầu vào.

Ví dụ, có thể hiểu như sau:

Giả sử chúng ta có -

x: Input variables, khi đó sẽ không có biến đầu ra tương ứng và các thuật toán cần khám phá ra mô hình thú vị trong dữ liệu để học.

Ví dụ về các thuật toán học máy không được giám sát bao gồm phân cụm K-mean, **K-nearest neighbors** Vân vân.

Dựa trên các nhiệm vụ ML, các thuật toán học tập không giám sát có thể được chia thành các lớp rộng sau:

- Clustering
- Association
- Giảm kích thước

Clustering

Các phương pháp phân cụm là một trong những phương pháp ML không giám sát hữu ích nhất. Các thuật toán này được sử dụng để tìm sự giống nhau cũng như các mẫu mối quan hệ giữa các mẫu dữ liệu và sau đó phân cụm các mẫu đó thành các nhóm có sự tương đồng dựa trên các tính năng. Ví dụ thực tế về phân nhóm là nhóm các khách hàng theo hành vi mua hàng của họ.

Association

Một phương pháp ML không giám sát hữu ích khác là **Association** được sử dụng để phân tích tập dữ liệu lớn để tìm ra các mẫu biểu thị thêm các mối quan hệ thú vị giữa các mục khác nhau. Nó cũng được gọi là **Association Rule Mining** hoặc là **Market basket analysis** được sử dụng chủ yếu để phân tích các mô hình mua sắm của khách hàng.

Dimensionality Reduction

Phương pháp ML không giám sát này được sử dụng để giảm số lượng biến đặc trưng cho mỗi mẫu dữ liệu bằng cách chọn tập hợp các đặc trưng chính hoặc đại diện. Một câu hỏi được đặt ra ở đây là tại sao chúng ta cần giảm kích thước? Lý do đằng sau là vấn đề phức tạp về không gian đối tượng nảy sinh khi chúng tôi bắt đầu phân tích và trích xuất hàng triệu đối tượng từ các mẫu dữ liệu. Vấn đề này thường đề cập đến "lời nguyền về chiều". PCA (Phân tích thành phần

chính), K-láng giếng gần nhất và phân tích phân biệt là một số thuật toán phổ biến cho mục đích này.

Anomaly Detection

Phương pháp ML không giám sát này được sử dụng để tìm ra sự xuất hiện của các sự kiện hiếm gặp hoặc quan sát thường không xảy ra. Bằng cách sử dụng kiến thức đã học, các phương pháp phát hiện điểm bất thường sẽ có thể phân biệt điểm dữ liệu bất thường hay điểm bình thường. Một số thuật toán không được giám sát như phân cụm, KNN có thể phát hiện sự bất thường dựa trên dữ liệu và các tính năng của nó.

2.6.1.2 Học tập bán giám sát

Các loại thuật toán hoặc phương pháp như vậy không được giám sát đầy đủ hoặc không được giám sát hoàn toàn. Về cơ bản chúng nằm giữa hai phương pháp học có giám sát và không giám sát. Các loại thuật toán này thường sử dụng thành phần học tập có giám sát nhỏ, tức là một lượng nhỏ dữ liệu được chú thích được gắn nhãn trước và thành phần học tập không được giám sát lớn tức là rất nhiều dữ liệu không được gắn nhãn để huấn luyện. Chúng ta có thể làm theo bất kỳ phương pháp nào sau đây để triển khai phương pháp học bán giám sát -

- Cách tiếp cận đầu tiên và đơn giản là xây dựng mô hình được giám sát dựa trên một lượng nhỏ dữ liệu được gắn nhãn và chú thích, sau đó xây dựng mô hình không được giám sát bằng cách áp dụng điều tương tự cho một lượng lớn dữ liệu không được gắn nhãn để có được nhiều mẫu được gắn nhãn hơn. Bây giờ, đào tạo mô hình trên chúng và lặp lại quy trình.
- Cách tiếp cận thứ hai cần thêm một số nỗ lực. Trong cách tiếp cận này, trước tiên chúng ta có thể sử dụng các phương pháp không được giám sát để phân cụm các mẫu dữ liệu tương tự, chú thích các nhóm này và sau đó sử dụng kết hợp thông tin này để đào tạo mô hình.

2.6.1.3 Học tăng cường

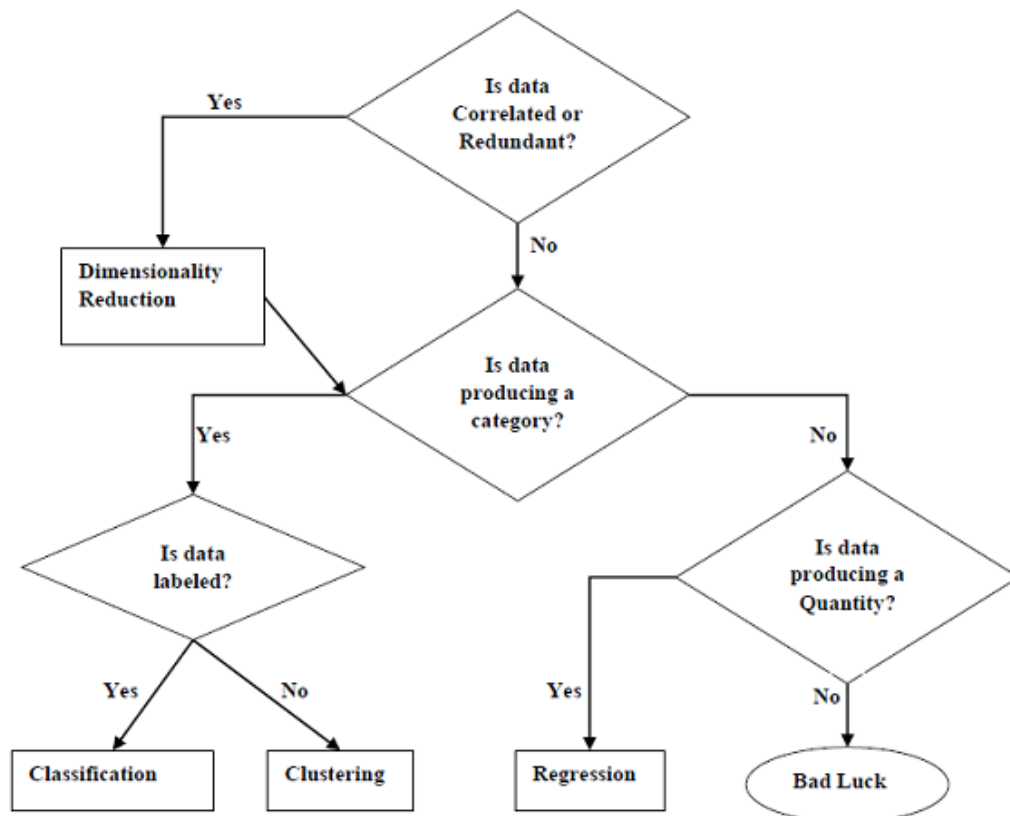
Các phương pháp này khác với các phương pháp đã nghiên cứu trước đây và cũng rất hiếm khi được sử dụng. Trong loại thuật toán học tập này, sẽ có một tác nhân mà chúng tôi muốn đào tạo trong một khoảng thời gian để nó có thể tương tác với một môi trường cụ thể. Tác nhân sẽ tuân theo một tập hợp các chiến lược để tương tác với môi trường và sau đó sau khi quan sát môi trường, tác nhân sẽ thực hiện các hành động liên quan đến tình trạng hiện tại của môi trường. Sau đây là các bước chính của phương pháp học củng cố -

- **Step 1** - Đầu tiên, chúng ta cần chuẩn bị cho đại lý một số chiến lược đặt ra ban đầu.
- **Step 2** - Sau đó quan sát môi trường và hiện trạng của nó.

- **Step 3** - Tiếp theo, chọn chính sách tối ưu liên quan đến tình trạng hiện tại của môi trường và thực hiện hành động quan trọng.
- **Step 4** - Bây giờ, đại lý có thể nhận được phần thưởng hoặc hình phạt tương ứng theo hành động của nó ở bước trước.
- **Step 5** - Bây giờ, chúng tôi có thể cập nhật các chiến lược nếu nó được yêu cầu.
- **Step 6** - Cuối cùng, lặp lại các bước 2-5 cho đến khi đại lý tìm hiểu và áp dụng các chính sách tối ưu.

Các nhiệm vụ phù hợp với học máy

Sơ đồ sau đây cho thấy loại nhiệm vụ nào thích hợp cho các vấn đề ML khác nhau -



2.6.2 Dựa trên khả năng học tập

Trong quá trình học, sau đây là một số phương pháp dựa trên năng lực học tập -

Batch Learning

Trong nhiều trường hợp, chúng tôi có các hệ thống Học máy end-to-end trong đó chúng tôi cần đào tạo mô hình trong một lần bằng cách sử dụng toàn bộ dữ liệu đào tạo có sẵn. Loại phương pháp học hoặc thuật toán như vậy được gọi là **Batch or Offline learning**. Nó được gọi là học Batch hoặc Offline vì nó là một thủ tục một lần và mô hình sẽ được đào tạo với dữ liệu trong một đợt duy nhất. Sau đây là các bước chính của phương pháp học theo lô -

- **Step 1** - Đầu tiên, chúng ta cần thu thập tất cả dữ liệu huấn luyện để bắt đầu huấn luyện mô hình.

- **Step 2** - Bây giờ, hãy bắt đầu đào tạo mô hình bằng cách cung cấp toàn bộ dữ liệu đào tạo trong một lần.
- **Step 3** - Tiếp theo, dừng quá trình học tập / đào tạo khi bạn đã đạt được kết quả / hiệu suất làm việc đạt yêu cầu.
- **Step 4**- Cuối cùng là triển khai mô hình đã được đào tạo này vào sản xuất. Tại đây, nó sẽ dự đoán đầu ra cho mẫu dữ liệu mới.

2.6.2.1 Học trực tuyến

Nó hoàn toàn ngược lại với phương pháp học theo lô hay học offline. Trong các phương pháp học tập này, dữ liệu huấn luyện được cung cấp theo nhiều lô tăng dần, được gọi là lô nhỏ, cho thuật toán. Sau đây là các bước chính của phương pháp học Online -

- **Step 1** - Đầu tiên, chúng ta cần thu thập tất cả các dữ liệu huấn luyện để bắt đầu huấn luyện mô hình.
- **Step 2** - Bây giờ, hãy bắt đầu đào tạo mô hình bằng cách cung cấp một lô nhỏ dữ liệu đào tạo cho thuật toán.
- **Step 3** - Tiếp theo, chúng ta cần cung cấp các lô nhỏ dữ liệu huấn luyện theo nhiều bước cho thuật toán.
- **Step 4** - Vì nó sẽ không dừng lại giống như học theo lô, do đó sau khi cung cấp toàn bộ dữ liệu đào tạo trong các lô nhỏ, hãy cung cấp các mẫu dữ liệu mới cho nó.
- **Step 5** - Cuối cùng, nó sẽ tiếp tục học hỏi trong một khoảng thời gian dựa trên các mẫu dữ liệu mới.

2.6.2.2 Dựa trên Cách tiếp cận Tổng quát

Trong quá trình học, sau đây là một số phương pháp dựa trên cách tiếp cận tổng quát hóa:

Học tập dựa trên phiên bản

Phương pháp học tập dựa trên phiên bản là một trong những phương pháp hữu ích để xây dựng các mô hình ML bằng cách tổng quát hóa dựa trên dữ liệu đầu vào. Nó đối lập với các phương pháp học tập đã nghiên cứu trước đây ở chỗ loại học tập này liên quan đến các hệ thống ML cũng như các phương pháp sử dụng chính các điểm dữ liệu thô để rút ra kết quả cho các mẫu dữ liệu mới hơn mà không cần xây dựng một mô hình rõ ràng về dữ liệu đào tạo.

Nói một cách dễ hiểu, học dựa trên phiên bản về cơ bản bắt đầu hoạt động bằng cách xem xét các điểm dữ liệu đầu vào và sau đó sử dụng một số liệu tương tự, nó sẽ khái quát hóa và dự đoán các điểm dữ liệu mới.

Học tập dựa trên mô hình

Trong phương pháp học dựa trên mô hình, một quá trình lặp lại diễn ra trên các mô hình ML được xây dựng dựa trên các tham số mô hình khác nhau, được gọi là siêu tham số và trong đó

dữ liệu đầu vào được sử dụng để trích xuất các tính năng. Trong bài học này, các siêu tham số được tối ưu hóa dựa trên các kỹ thuật xác nhận mô hình khác nhau. Đó là lý do tại sao chúng ta có thể nói rằng phương pháp học tập dựa trên Mô hình sử dụng cách tiếp cận ML truyền thống hơn theo hướng tổng quát hóa.

2.7 Học máy với Python - Chuẩn bị dữ liệu

2.7.1 Giới thiệu

Các thuật toán Học máy hoàn toàn phụ thuộc vào dữ liệu vì nó là khía cạnh quan trọng nhất giúp đào tạo mô hình có thể thực hiện được. Mặt khác, nếu chúng ta không thể hiểu được dữ liệu đó, trước khi đưa nó vào các thuật toán ML, một cỗ máy sẽ trở nên vô dụng. Nói một cách đơn giản, chúng ta luôn cần cung cấp dữ liệu phù hợp, tức là dữ liệu có tỷ lệ, định dạng chính xác và chứa các tính năng có ý nghĩa, cho vấn đề chúng ta muốn máy giải quyết.

Điều này làm cho việc chuẩn bị dữ liệu trở thành bước quan trọng nhất trong quy trình ML. Chuẩn bị dữ liệu có thể được định nghĩa là quy trình làm cho tập dữ liệu của chúng tôi phù hợp hơn với quy trình ML.

2.7.2 Tại sao phải xử lý trước dữ liệu?

Sau khi chọn dữ liệu thô để huấn luyện ML, công việc quan trọng nhất là xử lý trước dữ liệu. Theo nghĩa rộng, tiền xử lý dữ liệu sẽ chuyển đổi dữ liệu đã chọn thành một dạng mà chúng ta có thể làm việc hoặc có thể cung cấp cho các thuật toán ML. Chúng tôi luôn cần xử lý trước dữ liệu của mình để dữ liệu có thể giống như kỳ vọng của thuật toán học máy.

2.7.3 Kỹ thuật xử lý trước dữ liệu

Chúng tôi có các kỹ thuật tiền xử lý dữ liệu sau đây có thể được áp dụng trên tập dữ liệu để tạo dữ liệu cho các thuật toán ML -

Mở rộng quy mô

Hầu hết có lẽ tập dữ liệu của chúng tôi bao gồm các thuộc tính với tỷ lệ khác nhau, nhưng chúng tôi không thể cung cấp dữ liệu đó cho thuật toán ML do đó nó yêu cầu thay đổi tỷ lệ. Thay đổi tỷ lệ dữ liệu đảm bảo rằng các thuộc tính có cùng tỷ lệ. Nói chung, các thuộc tính được thay đổi tỷ lệ thành phạm vi 0 và 1. Các thuật toán ML như gradient descent và k-Nearest Neighbors yêu cầu dữ liệu được chia tỷ lệ. Chúng ta có thể bán lại dữ liệu với sự trợ giúp của lớp MinMaxScaler của thư viện Python scikit-learning.

Thí dụ

Trong ví dụ này, chúng tôi sẽ bán lại dữ liệu của bộ dữ liệu bệnh tiểu đường người da đỏ Pima mà chúng tôi đã sử dụng trước đó. Đầu tiên, dữ liệu CSV sẽ được tải (như đã thực hiện trong các chương trước) và sau đó với sự trợ giúp của lớp MinMaxScaler, nó sẽ được thay đổi tỷ lệ trong phạm vi 0 và 1.

Vài dòng đầu tiên của tập lệnh sau giống như chúng ta đã viết trong các chương trước khi tải dữ liệu CSV.

```
from pandas import read_csv
from numpy import set_printoptions
from sklearn import preprocessing
path = r'C:\pima-indians-diabetes.csv'
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
dataframe = read_csv(path, names=names)
array = dataframe.values
```

Bây giờ, chúng ta có thể sử dụng lớp MinMaxScaler để chia tỷ lệ dữ liệu trong phạm vi 0 và 1.

```
data_scaler = preprocessing.MinMaxScaler(feature_range=(0,1))
data_rescaled = data_scaler.fit_transform(array)
```

Chúng tôi cũng có thể tóm tắt dữ liệu cho đầu ra theo lựa chọn của chúng tôi. Ở đây, chúng tôi đang đặt độ chính xác thành 1 và hiển thị 10 hàng đầu tiên trong đầu ra.

```
set_printoptions(precision=1)
print ("\nScaled data:\n", data_rescaled[0:10])
```

Đầu ra

```
Scaled data:
[
  [0.4 0.7 0.6 0.4 0.  0.5 0.2 0.5 1. ]
  [0.1 0.4 0.5 0.3 0.  0.4 0.1 0.2 0. ]
  [0.5 0.9 0.5 0.  0.  0.3 0.3 0.2 1. ]
  [0.1 0.4 0.5 0.2 0.1 0.4 0.  0.  0. ]
  [0.  0.7 0.3 0.4 0.2 0.6 0.9 0.2 1. ]
  [0.3 0.6 0.6 0.  0.  0.4 0.1 0.2 0. ]
  [0.2 0.4 0.4 0.3 0.1 0.5 0.1 0.1 1. ]
  [0.6 0.6 0.  0.  0.  0.5 0.  0.1 0. ]
  [0.1 1.  0.6 0.5 0.6 0.5 0.  0.5 1. ]
  [0.5 0.6 0.8 0.  0.  0.  0.1 0.6 1. ]
]
```

Từ đầu ra ở trên, tất cả dữ liệu đã được thay đổi tỷ lệ thành phạm vi 0 và 1.

Bình thường hóa

Một kỹ thuật tiền xử lý dữ liệu hữu ích khác là Chuẩn hóa. Điều này được sử dụng để bán lại mỗi hàng dữ liệu có độ dài là 1. Nó chủ yếu hữu ích trong tập dữ liệu thưa thớt nơi chúng ta có rất nhiều số không. Chúng tôi có thể chia tỷ lệ dữ liệu với sự trợ giúp của lớp Normalizer của thư viện Python scikit-learning.

2.7.4 Các loại chuẩn hóa

Trong học máy, có hai loại kỹ thuật tiền xử lý chuẩn hóa như sau:

Chuẩn hóa L1

Nó có thể được định nghĩa là kỹ thuật chuẩn hóa sửa đổi các giá trị của tập dữ liệu theo cách mà trong mỗi hàng, tổng các giá trị tuyệt đối sẽ luôn lên đến 1. Nó còn được gọi là Sai lệch tuyệt đối ít nhất.

Example

Trong ví dụ này, chúng tôi sử dụng kỹ thuật L1 Normalize để chuẩn hóa dữ liệu của bộ dữ liệu bệnh tiểu đường người da đỏ Pima mà chúng tôi đã sử dụng trước đó. Đầu tiên, dữ liệu CSV sẽ được tải và sau đó với sự trợ giúp của lớp Normalizer, nó sẽ được chuẩn hóa.

Vài dòng đầu tiên của tập lệnh sau giống như chúng ta đã viết trong các chương trước khi tải dữ liệu CSV.

```
from pandas import read_csv
from numpy import set_printoptions
from sklearn.preprocessing import Normalizer
path = r'C:\pima-indians-diabetes.csv'
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
dataframe = read_csv(path, names=names)
array = dataframe.values
```

Bây giờ, chúng ta có thể sử dụng lớp Normalizer với L1 để chuẩn hóa dữ liệu.

```
Data_normalizer = Normalizer(norm='l1').fit(array)
Data_normalized = Data_normalizer.transform(array)
```

Chúng tôi cũng có thể tóm tắt dữ liệu cho đầu ra theo lựa chọn của chúng tôi. Ở đây, chúng tôi đang đặt độ chính xác thành 2 và hiển thị 3 hàng đầu tiên trong đầu ra.

```
set_printoptions(precision=2)
print ("\nNormalized data:\n", Data_normalized [0:3])
```

Output

Normalized data:

```
[
  [0.02 0.43 0.21 0.1  0. 0.1  0. 0.14 0. ]
  [0.   0.36 0.28 0.12 0. 0.11 0. 0.13 0. ]
  [0.03 0.59 0.21 0.   0. 0.07 0. 0.1  0. ]
]
```

Chuẩn hóa L2

Nó có thể được định nghĩa là kỹ thuật chuẩn hóa sửa đổi các giá trị của tập dữ liệu theo cách mà trong mỗi hàng, tổng các bình phương sẽ luôn lên đến 1. Nó còn được gọi là bình phương nhỏ nhất.

Example

Trong ví dụ này, chúng tôi sử dụng kỹ thuật Chuẩn hóa L2 để chuẩn hóa dữ liệu của bộ dữ liệu bệnh tiểu đường người da đỏ Pima mà chúng tôi đã sử dụng trước đó. Đầu tiên, dữ liệu CSV sẽ được tải (như đã thực hiện trong các chương trước) và sau đó với sự trợ giúp của lớp Normalizer, nó sẽ được chuẩn hóa.

Vài dòng đầu tiên của tập lệnh sau giống như chúng ta đã viết trong các chương trước khi tải dữ liệu CSV.

```
from pandas import read_csv
from numpy import set_printoptions
from sklearn.preprocessing import Normalizer
path = r'C:\pima-indians-diabetes.csv'
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']
dataframe = read_csv (path, names=names)
array = dataframe.values
```

Bây giờ, chúng ta có thể sử dụng lớp Normalizer với L1 để chuẩn hóa dữ liệu.

```
Data_normalizer = Normalizer(norm='l2').fit(array)
Data_normalized = Data_normalizer.transform(array)
```

Chúng tôi cũng có thể tóm tắt dữ liệu cho đầu ra theo lựa chọn của chúng tôi. Ở đây, chúng tôi đang đặt độ chính xác thành 2 và hiển thị 3 hàng đầu tiên trong đầu ra.

```
set_printoptions(precision=2)
print ("\nNormalized data:\n", Data_normalized [0:3])
```

Output

```
Normalized data:
[
  [0.03 0.83 0.4  0.2  0. 0.19 0. 0.28 0.01]
```

```
[0.01 0.72 0.56 0.24 0. 0.22 0. 0.26 0. ]  
[0.04 0.92 0.32 0. 0. 0.12 0. 0.16 0.01]  
]
```

2.7.5 Binarization

Như tên cho thấy, đây là kỹ thuật với sự trợ giúp của chúng tôi có thể tạo dữ liệu nhị phân. Chúng ta có thể sử dụng ngưỡng nhị phân để tạo dữ liệu của chúng ta ở dạng nhị phân. Các giá trị trên giá trị ngưỡng đó sẽ được chuyển đổi thành 1 và dưới ngưỡng đó sẽ được chuyển đổi thành 0. Ví dụ: nếu chúng ta chọn giá trị ngưỡng = 0,5, thì giá trị tập dữ liệu ở trên nó sẽ trở thành 1 và dưới ngưỡng này sẽ trở thành 0. Nghĩa là tại sao chúng ta có thể gọi nó **binarizing** dữ liệu hoặc **thresholding** dữ liệu. Kỹ thuật này hữu ích khi chúng ta có xác suất trong tập dữ liệu của mình và muốn chuyển chúng thành các giá trị rõ nét.

Chúng ta có thể mã hóa dữ liệu với sự trợ giúp của lớp Binarizer của thư viện Python scikit-learning.

Thí dụ

Trong ví dụ này, chúng tôi sẽ bán lại dữ liệu của bộ dữ liệu bệnh tiểu đường người da đỏ Pima mà chúng tôi đã sử dụng trước đó. Đầu tiên, dữ liệu CSV sẽ được tải và sau đó với sự trợ giúp của lớp Binarizer, nó sẽ được chuyển đổi thành các giá trị nhị phân, tức là 0 và 1 tùy thuộc vào giá trị ngưỡng. Chúng tôi đang lấy 0,5 làm giá trị ngưỡng.

Vài dòng đầu tiên của tập lệnh sau giống như chúng ta đã viết trong các chương trước khi tải dữ liệu CSV.

```
from pandas import read_csv  
  
from sklearn.preprocessing import Binarizer  
path = r'C:\pima-indians-diabetes.csv'  
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']  
dataframe = read_csv(path, names=names)  
array = dataframe.values
```

Bây giờ, chúng ta có thể sử dụng lớp Binarize để chuyển đổi dữ liệu thành các giá trị nhị phân.

```
binarizer = Binarizer(threshold=0.5).fit(array)  
Data_binarized = binarizer.transform(array)
```

Ở đây, chúng tôi đang hiển thị 5 hàng đầu tiên trong đầu ra.

```
print ("\nBinary data:\n", Data_binarized [0:5])
```

Đầu ra

Binary data:

```
[  
  [1. 1. 1. 1. 0. 1. 1. 1. 1.]  
  [1. 1. 1. 1. 0. 1. 0. 1. 0.]  
  [1. 1. 1. 0. 0. 1. 1. 1. 1.]  
  [1. 1. 1. 1. 1. 1. 0. 1. 0.]  
  [0. 1. 1. 1. 1. 1. 1. 1. 1.]  
]
```

2.7.6 Tiêu chuẩn hóa

Một kỹ thuật tiền xử lý dữ liệu hữu ích khác về cơ bản được sử dụng để biến đổi các thuộc tính dữ liệu với phân phối Gaussian. Nó khác giá trị trung bình và SD (Độ lệch chuẩn) với phân phối Gaussian chuẩn với giá trị trung bình là 0 và SD là 1. Kỹ thuật này hữu ích trong các thuật toán ML như hồi quy tuyến tính, hồi quy logistic giả định phân phối Gaussian trong tập dữ liệu đầu vào và tạo ra tốt hơn kết quả với dữ liệu được thay đổi tỷ lệ. Chúng ta có thể chuẩn hóa dữ liệu (mean = 0 và SD = 1) với sự trợ giúp của lớp StandardScaler của thư viện Scikit-learning Python.

Thí dụ

Trong ví dụ này, chúng tôi sẽ bán lại dữ liệu của bộ dữ liệu bệnh tiểu đường người da đỏ Pima mà chúng tôi đã sử dụng trước đó. Đầu tiên, dữ liệu CSV sẽ được tải và sau đó với sự trợ giúp của lớp StandardScaler, nó sẽ được chuyển đổi thành Phân phối Gaussian với giá trị trung bình = 0 và SD = 1.

Vài dòng đầu tiên của tập lệnh sau giống như chúng ta đã viết trong các chương trước khi tải dữ liệu CSV.

```
from sklearn.preprocessing import StandardScaler  
from pandas import read_csv  
from numpy import set_printoptions  
path = r'C:\pima-indians-diabetes.csv'  
names = ['preg', 'plas', 'pres', 'skin', 'test', 'mass', 'pedi', 'age', 'class']  
dataframe = read_csv(path, names=names)  
array = dataframe.values
```

Bây giờ, chúng ta có thể sử dụng lớp StandardScaler để bán lại dữ liệu.

```
data_scaler = StandardScaler().fit(array)
data_rescaled = data_scaler.transform(array)
```

Chúng tôi cũng có thể tóm tắt dữ liệu cho đầu ra theo lựa chọn của chúng tôi. Ở đây, chúng tôi đang đặt độ chính xác thành 2 và hiển thị 5 hàng đầu tiên trong đầu ra.

```
set_printoptions(precision=2)
print ("\nRescaled data:\n", data_rescaled [0:5])
```

Đầu ra

Rescaled data:

```
[
  [ 0.64  0.85  0.15  0.91 -0.69  0.2   0.47  1.43  1.37]
  [-0.84 -1.12 -0.16  0.53 -0.69 -0.68 -0.37 -0.19 -0.73]
  [ 1.23  1.94 -0.26 -1.29 -0.69 -1.1   0.6  -0.11  1.37]
  [-0.84 -1.   -0.16  0.15  0.12 -0.49 -0.92 -1.04 -0.73]
  [-1.14  0.5  -1.5   0.91  0.77  1.41  5.48 -0.02  1.37]
]
```

Ghi nhãn dữ liệu

Chúng tôi đã thảo luận về tầm quan trọng của dữ liệu tốt đối với các thuật toán ML cũng như một số kỹ thuật xử lý trước dữ liệu trước khi gửi nó đến các thuật toán ML. Một khía cạnh nữa trong vấn đề này là ghi nhãn dữ liệu. Nó cũng rất quan trọng để gửi dữ liệu đến các thuật toán ML có nhãn thích hợp. Ví dụ, trong trường hợp các vấn đề phân loại, rất nhiều nhãn ở dạng từ, số, v.v. có trên dữ liệu.

Mã hóa nhãn là gì?

Hầu hết các hàm sklearn đều mong đợi rằng dữ liệu có nhãn số hơn là nhãn từ. Do đó, chúng ta cần chuyển đổi các nhãn đó thành nhãn số. Quá trình này được gọi là mã hóa nhãn. Chúng ta có thể thực hiện mã hóa nhãn của dữ liệu với sự trợ giúp của hàm LabelEncoder () của thư viện Python scikit-learning.

2.7.7 ML với Python - Lựa chọn tính năng dữ liệu

Trong chương trước, chúng ta đã xem chi tiết cách xử lý trước và chuẩn bị dữ liệu cho học máy. Trong chương này, chúng ta hãy hiểu chi tiết lựa chọn tính năng dữ liệu và các khía cạnh khác nhau liên quan đến nó.

Tầm quan trọng của việc lựa chọn tính năng dữ liệu

Hiệu suất của mô hình học máy tỷ lệ thuận với các tính năng dữ liệu được sử dụng để đào tạo nó. Hiệu suất của mô hình ML sẽ bị ảnh hưởng tiêu cực nếu các tính năng dữ liệu được cung cấp cho nó không liên quan. Mặt khác, việc sử dụng các tính năng dữ liệu có liên quan có thể làm tăng độ chính xác của mô hình ML của bạn, đặc biệt là hồi quy tuyến tính và logistic.

Bây giờ câu hỏi nảy sinh rằng lựa chọn tính năng tự động là gì? Nó có thể được định nghĩa là quá trình với sự trợ giúp của chúng tôi chọn các tính năng đó trong dữ liệu của chúng tôi có liên quan nhất đến đầu ra hoặc biến dự đoán mà chúng tôi quan tâm. Nó còn được gọi là lựa chọn thuộc tính.

Sau đây là một số lợi ích của việc lựa chọn tính năng tự động trước khi lập mô hình dữ liệu:

- Thực hiện lựa chọn tính năng trước khi lập mô hình dữ liệu sẽ làm giảm sự trang bị quá mức.
- Thực hiện lựa chọn tính năng trước khi lập mô hình dữ liệu sẽ làm tăng độ chính xác của mô hình ML.
- Thực hiện lựa chọn tính năng trước khi lập mô hình dữ liệu sẽ giảm thời gian đào tạo

Kỹ thuật lựa chọn tính năng

Sau đây là các kỹ thuật chọn tính năng tự động mà chúng ta có thể sử dụng để lập mô hình dữ liệu ML bằng Python:

Lựa chọn đơn biến

Kỹ thuật lựa chọn tính năng này rất hữu ích trong việc lựa chọn các tính năng đó, với sự trợ giúp của kiểm tra thống kê, có mối quan hệ chặt chẽ nhất với các biến dự đoán. Chúng ta có thể triển khai kỹ thuật lựa chọn tính năng đơn biến với sự trợ giúp của `SelectKBest` class của thư viện Python `scikit-learning`.

Loại bỏ tính năng đệ quy

Như tên cho thấy, kỹ thuật lựa chọn tính năng RFE (Loại bỏ tính năng đệ quy) loại bỏ các thuộc tính một cách đệ quy và xây dựng mô hình với các thuộc tính còn lại. Chúng ta có thể triển khai kỹ thuật lựa chọn tính năng RFE với sự trợ giúp của lớp RFE của thư viện Python `scikit-learning`.

Phân tích thành phần chính (PCA)

PCA, thường được gọi là kỹ thuật giảm dữ liệu, là kỹ thuật lựa chọn tính năng rất hữu ích vì nó sử dụng đại số tuyến tính để chuyển đổi tập dữ liệu thành dạng nén. Chúng ta có thể triển khai kỹ thuật lựa chọn tính năng PCA với sự trợ giúp của lớp PCA của thư viện Python `scikit-learning`. Chúng ta có thể chọn số lượng các thành phần chính trong đầu ra.

Tầm quan trọng của tính năng

Như tên cho thấy, kỹ thuật quan trọng của đối tượng địa lý được sử dụng để chọn các đối tượng địa lý quan trọng. Về cơ bản, nó sử dụng bộ phân loại có giám sát được đào tạo để chọn các tính năng. Chúng ta có thể thực hiện kỹ thuật lựa chọn tính năng này với sự trợ giúp của lớp ExtraTreeClassifier của thư viện Python scikit-learning.

CHƯƠNG III: ỨNG DỤNG MACHINE LEARNING TRONG PHÂN TÍCH VÀ DỰ BÁO LÚN CÔNG TRÌNH

3.1 Các mô hình ML

3.1.1 Thuật toán ML - Phân loại

- [Phân loại - Giới thiệu](#)
- [Giải thuật phân loại - Hồi quy logistic](#)
- [Máy vectơ hỗ trợ \(SVM\)](#)
- [Thuật toán phân loại - Cây quyết định](#)
- [Thuật toán phân loại - Naïve Bayes](#)
- [Thuật toán phân loại - Rừng ngẫu nhiên](#)

3.1.2 Thuật toán ML - Phân cụm

- [Thuật toán phân cụm - Tổng quan](#)
- [Thuật toán phân cụm - Thuật toán K-mean](#)
- [Thuật toán phân cụm - Thuật toán dịch chuyển trung bình](#)
- [Thuật toán phân cụm - Phân cụm theo thứ bậc](#)

3.1.3 Thuật toán ML - Thuật toán KNN

- [Thuật toán KNN - Tìm những người hàng xóm gần nhất](#)
- [Học máy - Chỉ số hiệu suất](#)
- [Học máy - Quy trình làm việc tự động](#)
- [Cải thiện Hiệu suất của Mô hình ML](#)
- [Cải thiện Hiệu suất của Mô hình ML \(Tiếp theo...\)](#)

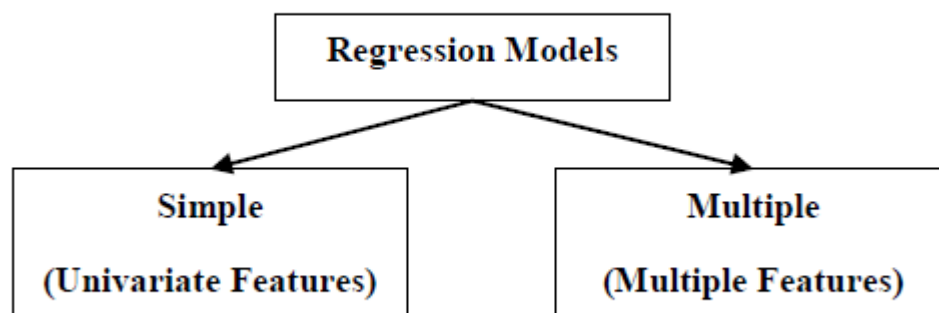
3.1.4 Thuật toán ML - Hồi quy

- [Thuật toán hồi quy - Tổng quan](#)
- [Thuật toán hồi quy - Hồi quy tuyến tính](#)

3.2 Lựa chọn mô hình ML trong phân tích kế quả dự báo lún

Hồi quy là một công cụ thống kê và máy học quan trọng và được sử dụng rộng rãi. Mục tiêu chính của các nhiệm vụ dựa trên hồi quy là dự đoán các nhãn hoặc phản hồi đầu ra là các giá trị số tiếp tục, cho dữ liệu đầu vào đã cho. Đầu ra sẽ dựa trên những gì mô hình đã học được trong giai đoạn đào tạo. Về cơ bản, mô hình hồi quy sử dụng các đặc trưng dữ liệu đầu vào (các biến độc lập) và các giá trị đầu ra số liên tục tương ứng của chúng (biến phụ thuộc hoặc biến kết quả) để tìm hiểu mối liên hệ cụ thể giữa đầu vào và đầu ra tương ứng.

Các loại mô hình hồi quy



Mô hình hồi quy có hai loại sau:

Simple regression model - Đây là mô hình hồi quy cơ bản nhất, trong đó các dự đoán được hình thành từ một đặc trưng duy nhất, đơn biến của dữ liệu.

Multiple regression model - Như tên của nó, trong mô hình hồi quy này, các dự đoán được hình thành từ nhiều đặc điểm của dữ liệu.

Xây dựng Regressor bằng Python

Mô hình bộ hồi quy trong Python có thể được xây dựng giống như chúng ta đã xây dựng bộ phân loại. Scikit-learning, một thư viện Python dành cho học máy cũng có thể được sử dụng để xây dựng một bộ hồi quy bằng Python.

Trong ví dụ sau, chúng tôi sẽ xây dựng mô hình hồi quy cơ bản sẽ phù hợp với một dòng với dữ liệu tức là hồi quy tuyến tính. Các bước cần thiết để xây dựng một bộ hồi quy trong Python như sau:

Bước 1: Nhập gói python cần thiết

Để xây dựng một bộ hồi quy bằng scikit-learning, chúng ta cần nhập nó cùng với các gói cần thiết khác. Chúng tôi có thể nhập bằng cách sử dụng tập lệnh sau:

```
import numpy as np
from sklearn import linear_model
import sklearn.metrics as sm
```

```
import matplotlib.pyplot as plt
```

Bước 2: Nhập tập dữ liệu

Sau khi nhập gói cần thiết, chúng ta cần một tập dữ liệu để xây dựng mô hình dự đoán hồi quy. Chúng tôi có thể nhập nó từ tập dữ liệu sklearn hoặc có thể sử dụng tập dữ liệu khác theo yêu cầu của chúng tôi. Chúng tôi sẽ sử dụng dữ liệu đầu vào đã lưu của chúng tôi. Chúng tôi có thể nhập nó với sự trợ giúp của tập lệnh sau:

```
input = r'C:\linear.txt'
```

Tiếp theo, chúng ta cần tải dữ liệu này. Chúng tôi đang sử dụng hàm `np.loadtxt` để tải nó.

```
input_data = np.loadtxt(input, delimiter=',')  
X, y = input_data[:, :-1], input_data[:, -1]
```

Bước 3: Tổ chức dữ liệu thành các bộ đào tạo và kiểm tra

Do đó, khi chúng ta cần kiểm tra mô hình của mình trên dữ liệu không nhìn thấy, chúng ta sẽ chia tập dữ liệu của mình thành hai phần: tập huấn luyện và tập thử nghiệm. Lệnh sau sẽ thực hiện nó:

```
training_samples = int(0.6 * len(X))  
testing_samples = len(X) - num_training  
  
X_train, y_train = X[:training_samples], y[:training_samples]  
  
X_test, y_test = X[training_samples:], y[training_samples:]
```

Bước 4: Đánh giá và dự đoán mô hình

Sau khi phân chia dữ liệu thành đào tạo và thử nghiệm, chúng ta cần xây dựng mô hình. Chúng tôi sẽ sử dụng hàm `LinearRegression()` của Scikit-learning cho mục đích này. Lệnh sau sẽ tạo một đối tượng hồi quy tuyến tính.

```
reg_linear = linear_model.LinearRegression()
```

Tiếp theo, đào tạo mô hình này với các mẫu đào tạo như sau:

```
reg_linear.fit(X_train, y_train)
```

Bây giờ, cuối cùng chúng ta cần thực hiện dự đoán với dữ liệu thử nghiệm.

```
y_test_pred = reg_linear.predict(X_test)
```

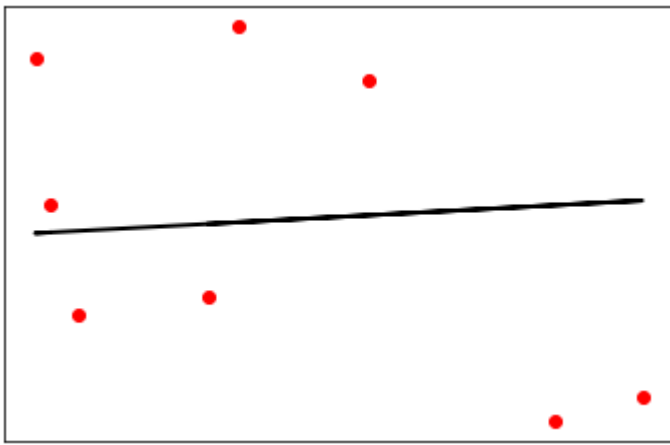
Bước 5: Vẽ và hình dung

Sau khi dự đoán, chúng ta có thể vẽ và hình dung nó với sự trợ giúp của tập lệnh sau:

Example

```
plt.scatter(X_test, y_test, color='red')
plt.plot(X_test, y_test_pred, color='black', linewidth=2)
plt.xticks()
plt.yticks()
plt.show()
```

Output



Trong kết quả trên, chúng ta có thể thấy đường hồi quy giữa các điểm dữ liệu.

Bước 6: Tính toán hiệu suất

Chúng tôi cũng có thể tính toán hiệu suất của mô hình hồi quy với sự trợ giúp của các số liệu hiệu suất khác nhau như sau:

Example

```
print("Regressor model performance:")
print("Mean absolute error(MAE) =", round(sm.mean_absolute_error(y_test, y_test_pred),
2))
print("Mean squared error(MSE) =", round(sm.mean_squared_error(y_test, y_test_pred),
2))
print("Median absolute error =", round(sm.median_absolute_error(y_test, y_test_pred), 2))
print("Explain variance score =", round(sm.explained_variance_score(y_test, y_test_pred),
2))
print("R2 score =", round(sm.r2_score(y_test, y_test_pred), 2))
```

Output

Regressor model performance:

Mean absolute error(MAE) = 1.78

Mean squared error(MSE) = 3.89

Median absolute error = 2.01

Explain variance score = -0.09

R2 score = -0.09

3.3 Các ứng dụng của thuật toán hồi quy ML như sau:

Forecasting or Predictive analysis- Một trong những ứng dụng quan trọng của hồi quy là dự báo hoặc phân tích dự báo. Ví dụ, chúng ta có thể dự báo GDP, giá dầu hay nói cách đơn giản là dữ liệu định lượng thay đổi theo thời gian và có thể dự báo độ lún công trình theo thời gian.

Optimization- Có thể tối ưu hóa các quy trình mô tả độ lún với sự trợ giúp của hồi quy.

Error correction- Trong quan trắc lún, việc đưa ra quyết định đúng đắn cũng quan trọng không kém việc tối ưu hóa quy trình mô tả độ lún. Hồi quy có thể giúp chúng ta đưa ra quyết định chính xác cũng như điều chỉnh quyết định đã được thực hiện.

Economics- Dự báo độ lún trong tương lai cho công trình hết sức quan trọng. Chúng ta có thể sử dụng hồi quy để dự đoán độ lún công trình theo các mô hình lựa chọn

Finance- Một công ty tài chính luôn quan tâm đến việc giảm thiểu danh mục rủi ro và muốn biết các yếu tố ảnh hưởng đến công trình. Tất cả những điều này có thể được dự đoán với sự trợ giúp của mô hình hồi quy.

3.4 Thực nghiệm trên số liệu thực tế:

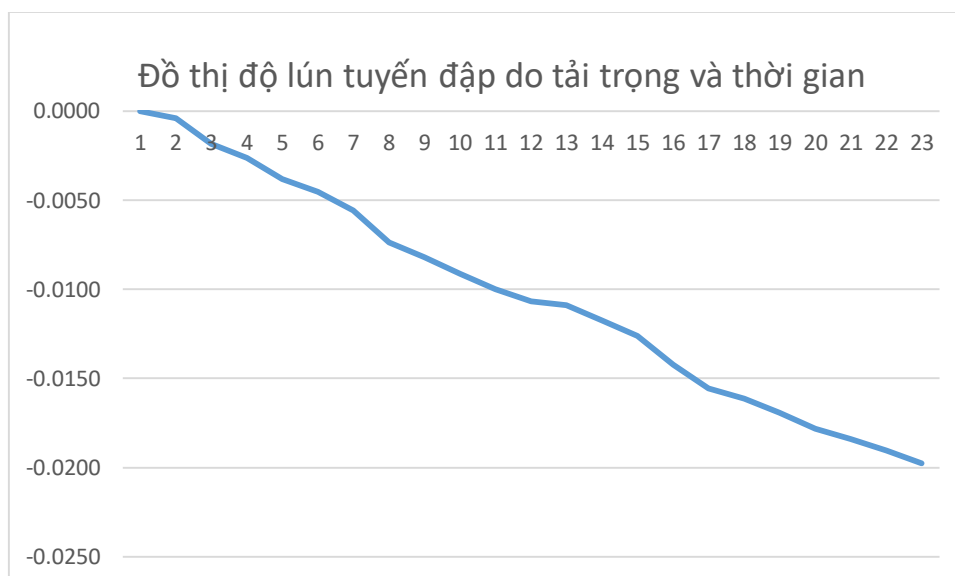
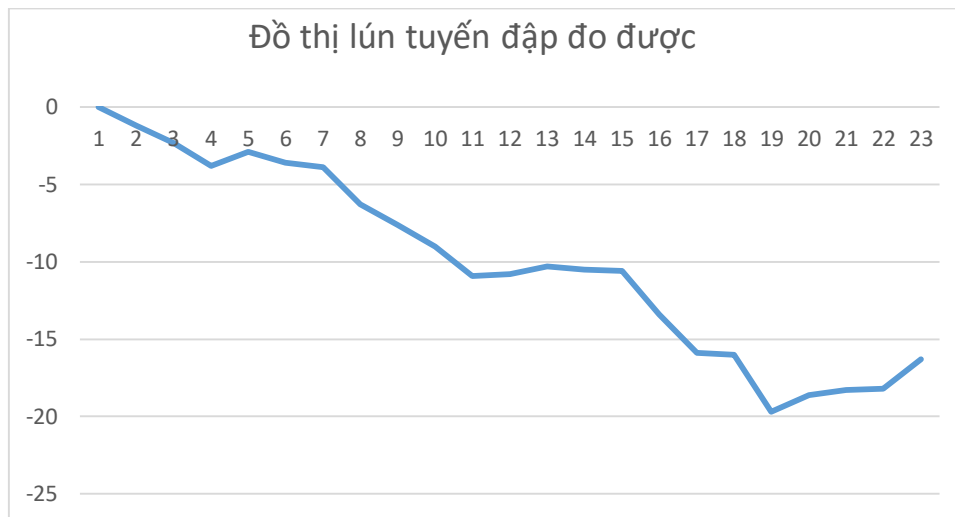
Do lun_mm	Do lun_m	hs	thoi gian	Muc nuoc_so với trung bình	
0	0	1	0	4.47	111.51
-1.2	-0.0012	1	0.06	-4.33	102.71
-2.3	-0.0023	1	0.26	-5.22	101.82
-3.8	-0.0038	1	0.37	-17.42	89.62
-2.9	-0.0029	1	0.54	2.76	109.8
-3.6	-0.0036	1	0.64	10.16	117.2
-3.9	-0.0039	1	0.79	9.37	116.41
-6.3	-0.0063	1	1.04	-5.18	101.86
-7.6	-0.0076	1	1.16	2.87	109.91
-9	-0.009	1	1.29	-5.19	101.85
-10.9	-0.0109	1	1.41	-20.2	86.84

-10.8	-0.0108	1	1.51	-2.95	104.09
-10.3	-0.0103	1	1.54	0.55	107.59
-10.5	-0.0105	1	1.66	8.36	115.4
-10.6	-0.0106	1	1.78	9.68	116.72
-13.4	-0.0134	1	2.01	7.28	114.32
-15.9	-0.0159	1	2.2	4.56	111.6
-16	-0.016	1	2.28	2.07	109.11
-19.7	-0.0197	1	2.39	-21.21	85.83
-18.6	-0.0186	1	2.52	-2.36	104.68
-18.3	-0.0183	1	2.6	3.78	110.82
-18.2	-0.0182	1	2.69	8.81	115.85
-16.3	-0.0163	1	2.79	9.34	116.38

Kết quả tính hồi quy trên số liệu thực tế:

SUMMARY OUTPUT								
<i>Regression Statistics</i>								
Multiple R	0.997966							
R Square	0.995936							
Adjusted R Square	0.945529							
Standard Error	0.000801							
Observations	23							
ANOVA								
	<i>df</i>	<i>SS</i>	<i>MS</i>	<i>F</i>	<i>Significance F</i>			
Regression	3	0.003142	0.001047	1633.651	4.28E-23			
Residual	20	1.28E-05	6.41E-07					
Total	23	0.003155						
	<i>Coefficients</i>	<i>Standard Error</i>	<i>t Stat</i>	<i>P-value</i>	<i>Lower 95%</i>	<i>Upper 95%</i>	<i>Lower 95.0%</i>	<i>Upper 95.0%</i>
Intercept	0	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
hs	0.000314	0.00033	0.949803	0.35355	-0.00038	0.001003	-0.00038	0.001003
thời gian	-0.00708	0.000196	-36.1931	1.05E-19	-0.00749	-0.00667	-0.00749	-0.00667
Mức nước_so với trung bình	0.000106	1.85E-05	5.742486	1.28E-05	6.77E-05	0.000145	6.77E-05	0.000145

Đồ thị hồi quy trên số liệu thực tế:



KẾT LUẬN

- Ngôn ngữ Python là ngôn ngữ đa dụng, dễ hiểu có thể dễ dàng lập chương trình ứng dụng Machine learning trong phân tích và dự báo độ lún công trình.
- Machine learning có thể ứng dụng trong nhiều lĩnh vực của phân tích dữ liệu và có thể phân tích dữ liệu trong quan trắc lún công trình.
- Machine learning có thể ứng dụng nhiều thuật toán để phân tích, tuy nhiên ứng dụng hồi quy trong phân tích dữ liệu quan trắc lún công trình bước đầu là phù hợp.