HỒI QUY TUYẾN TÍNH

Giới thiệu

Trải qua 04 tuần học, chúng ta đã tìm hiểu được rất nhiều nội dung liên quan đến việc phân tích và xử lý dữ liệu với tập dữ liệu "California Housing Prices". Ngày hôm nay, chúng ta sẽ bắt đầu một hành trình mới với máy học. Và bài toán cơ bản nhất tôi muốn nhắc đến liên quan đến bài toán "Hồi quy tuyến tính", nó là một trong những bài toán quan thuộc nhất khi chúng ta bắt đầu tìm hiểu về "Học có giám sát" trong máy học. Nhiều nhà nghiên cứu vẫn thường gọi bài toán này với cái tên quen thuộc là Linear fitting trong bộ môn xác xuất thống kê hoặc Linear Least Square.

Trong bài học ngày hôm nay, tôi sẽ lần lượt hướng dẫn các bạn đi qua 03 nội dung cơ bản nhất của bài toán này, bao gồm:

- Phân tích toán học
- Biểu diễn bài toán với Python
- Ưu điểm và nhược điểm của bài toán

Ý nghĩa về mặt toán học

Quay trở lại tập dữ liệu "California Housing Prices", hãy tưởng tượng rằng chúng ta đang là một nhà môi giới bất động sản ở bang California - Mỹ và một ngày nọ có một khách hàng đến văn phòng và đưa ra giải thuyết rằng:

- Có diện tích x_1 m^2,
- Có x_2 phòng ngủ,
- Cách trung tâm thành phố x_3 km

Và rồi, khách hàng này muốn chúng ta đưa ra một mức giá đối với ngôi nhà mà họ đưa ra yêu cầu với những số liệu bên trên. Quả thật đây là một câu trả lời khó nếu như chúng ta hoàn toàn không có số liệu gì để tham khảo. Nhưng nếu trong trường hợp chúng ta có một bộ dữ liệu gồm khoảng vài nghìn ngôi nhà ở California với đầy đủ thông tin thống kê, bao gồm diện tích, số lượng phòng ngủ và khoảng cách đến trung tâm thành phố, liệu rằng chúng ta có thể đưa ra được dự đoán về giá của ngôi nhà mà khách hàng yêu cầu hay không? Nếu có thì hàm dự đoán của chúng ta sẽ như thế nào? và liệu rằng số liệu thống kê bên trên sẽ được chúng ta sử dụng như thế nào?

Thông thường các bài toán liên quan đến mô hình dự đoán “Linear regression” sẽ có dạng $y = f(x)$, trong đó:

- y sẽ là một hàm vô hướng (scalar) biểu diễn giá trị đầu ra (output) của bài toán (trong trường hợp này sẽ là giá của căn nhà mà khách hàng yêu cầu).
- x sẽ bao gồm x_1, x_2, và x_3 sẽ được biểu diễn dưới dạng một danh sách $x = [x_1, x_2, x_3]$ và nó là một vector chứa các giá trị đầu vào (input) của bài toán.

Nếu chúng ta để ý thì sẽ thấy một số đặc điểm quan trọng trong lĩnh vực bất động sản như sau:

- Giá của ngôi nhà càng cao thì ngôi nhà ấy có diện tích càng lớn, số phòng ngủ càng nhiều và gần với trung tâm thành phố.

- Ngược lại, nếu ngôi nhà có diện tích nhỏ, số phòng ngủ ít và xa trung tâm thành phố thì có giá càng thấp.

Do đó, để thể hiện mối quan hệ giữa giá ngôi nhà và 03 đại lượng đầu vào thì chúng ta dùng hàm như sau:

$y \approx f(x) = \hat{y}$, và $f(x) = w_1x_1 + w_2x_2 + w_3x_3 + w_0$, trong đó:

- w_1, w_2, w_3, w_0 là các hằng số, và w_0 còn được gọi là thiên vị (bias). Mối quan hệ của $y \approx f(x)$ là một mối quan hệ tuyến tính. Và bái toán dự đoán giá nhà California mà chúng ta đang làm thuộc bài toán hồi quy (regression). Mục tiêu của chúng ta là đi tìm cách hiệu chỉnh các hệ số $\{w_1, w_2, w_3, w_0\}$ để tối ưu hoá nó được gọi là bài toán hồi quy tuyến tính.

- y là giá trị thực của kết quả (outcome) và dựa trên số liệu mà chúng ta đã thống kê có trong tập dữ liệu đào tạo (training).

- $\hat{y}$ là giá trị mà mô hình hồi quy tuyến tính của chúng ta dự đoán

Chú ý: y và $\hat{y}$ là hai giá trị khác nhau do có sự sai số của mô hình đào tạo và việc chúng ta cần làm là phải tìm ra cách nào để hiệu chỉnh các hệ số $\{w_1, w_2, w_3, w_0\}$ để sai số này ở mức thấp nhất có thể.

Vì sao chúng ta lại gọi là tuyến tính?

Khi nhắc đến một bài toán tuyến tính, chúng ta cần tưởng tượng rằng một hàm số tuyến tính sẽ có dạng đồ thị là một đường thẳng trong không gian hai chiều, và nó sẽ có dạng đồ thị là một mặt phẳng trong không gian ba chiều. Trong không gian đa chiều, "khái niệm mặt phẳng không còn phù hợp nữa, thay vào đó, một khái niệm khác ra đời được gọi là siêu mặt phẳng (hyperplane). Các hàm số tuyến tính là các hàm đơn giản nhất, vì chúng thuận tiện trong việc hình dung và tính toán. Chúng ta sẽ được thấy trong các bài viết sau, tuyến tính rất quan trọng và hữu ích trong các bài toán Machine Learning. Kinh nghiệm cá nhân tôi cho thấy, trước khi hiểu được các thuật toán phi tuyến (non-linear, không phẳng), chúng ta cần nắm vững các kỹ thuật cho các mô hình tuyến tính."

Phân tích toán học

<u>Dạng của bài toán Hồi quy tuyến tính</u>

Giả sử chúng ta có $f(x) = w_1x_1 + w_2x_2 + w_3x_3 + w_0$ (1) là một vector hàng như đã nhắc đến bên trên, bâu giờ chúng ta đặt $w = [w_1, w_2, w_3, w_0]^T$ là một vector cột, và hệ số mà chúng ta cần phải tối ưu là $\bar{x} = [w_1, w_2, w_3, 1]$ (nó có dạng là một vector hàng mở rộng). Giá trị 1 ở đây có nghĩa là chúng ta sẽ mặc định đặc $w_0 = 1$ để cho bài toán đơn giản và dễ dàng tính toán hơn. Như thế, chúng ta sẽ dễ dàng viết lại phương trình (1) thành: $y \approx \bar{x}w = \hat{y}$

Sai số dự đoán

Giả sử chúng ta gọi e là sai số giữa giá trị thực y và giá trị dự đoán $\hat{y}$. Và mục tiêu của chúng ta sẽ làm sao để cho sai số e là nhỏ nhất trong bài toán hồi quy tuyến tính. Như thế, chúng ta có thể biểu diễn dưới dạng:

$\frac{1}{2}e^2 = \frac{1}{2}(y - \hat{y})^2 = \frac{1}{2}(y - \bar{x}w)^2$, trong đó:

- $\frac{1}{2}$ là hệ số được thêm vào để thuận tiện cho việc tính toán sau này, vì khi tính đạo hàm thì $\frac{1}{2}$ sẽ bị triệt tiêu.
- e^2 được sử dụng bởi vì trong một số trường hợp $e = y - \hat{y}$ có thể là một số âm.

Chú ý: Việc nói $e = y - \hat{y}$ là số nhỏ nhất hoàn toàn không chính xác vì nếu $e = -\infty$ là một số rất nhỏ nhưng sai số của chúng ta là rất lớn.

Vậy tại sao chúng ta không dùng giá trị tuyệt đối của e, tức $|e|$ mà chúng ta lại sử dụng là e^2?

Hàm mất mát

Trong một tập hợp dữ liệu, chúng ta luôn luôn xác định được các cặp giá trị đầu vào (input) và giá trị đầu ra (outcome) và chúng được ký kiệu là (x_i, y_i), với $i = 1,2,3,4...N$. N được gọi là số lượng dữ liệu mà chúng ta có thể quan sát được. Và mục tiêu của chúng ta sẽ phải tìm tổng sai số là nhỏ nhất, việc này cũng tương tự như việc chúng ta tìm w để hàm số sau đạt giá trị nhỏ nhất:

$L(w) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \bar{x_i}w)^2$ (2), trong đó:

$L(w)$ được gọi là hàm mất mát (Loss function). Chúng ta luôn mong muốn rằng sự mất mát (sai số) là nhỏ nhất, điều đó đồng nghĩa với việc tìm vector hệ số w sao cho giá trị của hàm mất mát này càng nhỏ càng tốt. Giá trị của w làm cho hàm mất mát đạt giá trị nhỏ nhất được gọi là điểm tối ưu (optimal point), ký hiệu: $w^* = argmin L(w)$.

Bây giờ, chúng ta hãy đơn giản phương trình (2) bên trên bằng cách đặt $y = [y_1, y_2, y_3 \ldots, y_N]$ là một vector cột chưa tất cả giá trị output của tập dữ liệu training và $\bar{X} = [\bar{x_1}, \bar{x_2}, \bar{x_3}, \ldots, \bar{x_N}]$ là một ma trận dữ liệu đầu vào mở rộng mà mỗi hàng của nó là một điểm dữ liệu. Khi đó, hàm mất mát $L(w)$ được viết dưới dạng một ma trận đơn giản: $(2) = \frac{1}{2}||y - \bar{X}x||_2^2$ (3), trong đó:

$||y - \bar{X}x||_2$ sẽ là chuẩn Euclid hay còn được gọi là khoảng cách Euclid, và chúng ta cũng có thể đơn giản hoá công thức này bắt cách đặt $z = y - \bar{X}x$. Như thế $||z||_2^2$ được gọi là tổng của bình phương mỗi phần tử của vector z. Như thế phương trình (3) là một dạng đơn giản của hàm mất mát.

Tìm nghiệm của bài toán Hồi quy tuyến tính

Một cách đơn giản để tìm nghiệm của của bài toán tối ưu hay bài toán hồi quy tuyến tính là giải phương trình đạo hàm (gradient) bằng 0. Và việc này sẽ không quá phức tạp đối với chúng ta.

Để tính đạo hàm của hàm mất mát chúng ta sẽ áp dụng công thức:

$$\frac{\partial L(w)}{\partial w} = \bar{X}^T(\bar{X}w - y)$$

Đến đây tôi xin quay lại câu hỏi ở phần Sai số dự đoán phía trên về việc tại sao không dùng trị tuyệt đối mà lại dùng bình phương. Câu trả lời là hàm bình phương có đạo hàm tại mọi nơi, trong khi hàm trị tuyệt đối thì không (đạo hàm không xác định tại 0).

- Phương trình đạo hàm bằng 0 sẽ tương đương với: $0 = \bar{X}^T(\bar{X}w - y) \Rightarrow \bar{X}^T\bar{X}w = \bar{X}^Ty \triangleq b$ (4), trong đó, $\bar{X}^Ty \triangleq b$ có nghĩa là đặt $\bar{X}^Ty$ bằng b.

- Nếu ma trận vuông $A \triangleq \bar{X}^Ty$ khả nghịch (non-singular hay invertible) thì phương trình (4) có nghiệm duy nhất: $w = A^{-1}b$.

- Trong trường hợp nếu ma trận A không khả nghịch (có định thức bằng 0) thì phương trình (4) sẽ vô nghiệm hoặc là nó sẽ có vô số nghiệm. Khi đó, chúng ta sử dụng khái niệm liên quan đến giả nghịch đảo (pseudo inverse) $A\dagger$, tức nó là trường hợp tổng quát của nghịch đảo khi ma trận không khả nghịch hoặc thậm chí không vuông.

Với khái niệm giả nghịch đảo, điểm tối ưu của bài toán Hồi quy tuyến tính có dạng:

$$w = A\dagger b = \bar{X}^T\bar{X}\dagger\bar{X}y \text{ (5)}$$

Thực nghiệm bằng chương trình Python

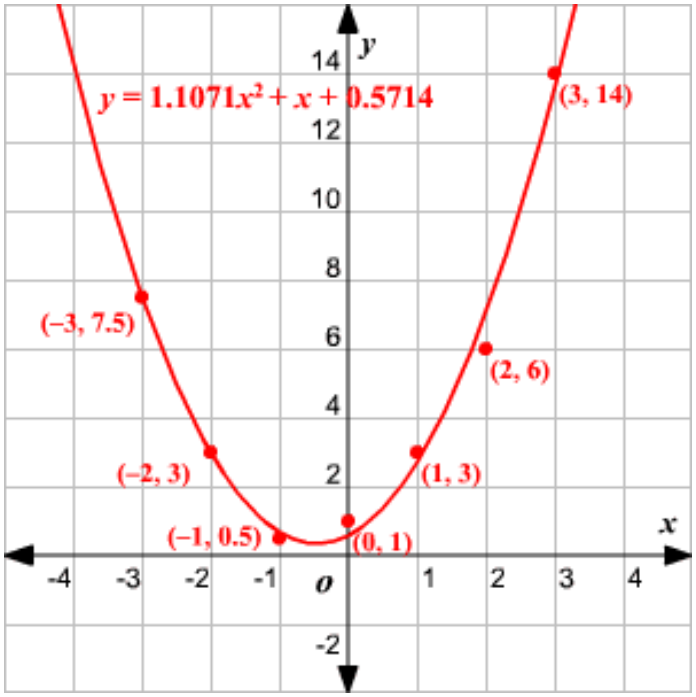
Các bạn hãy chuyển qua file Google Colab để xem thực nghiệm bằng chương trình Python.

Các vấn đề mở rộng

Ứng dụng

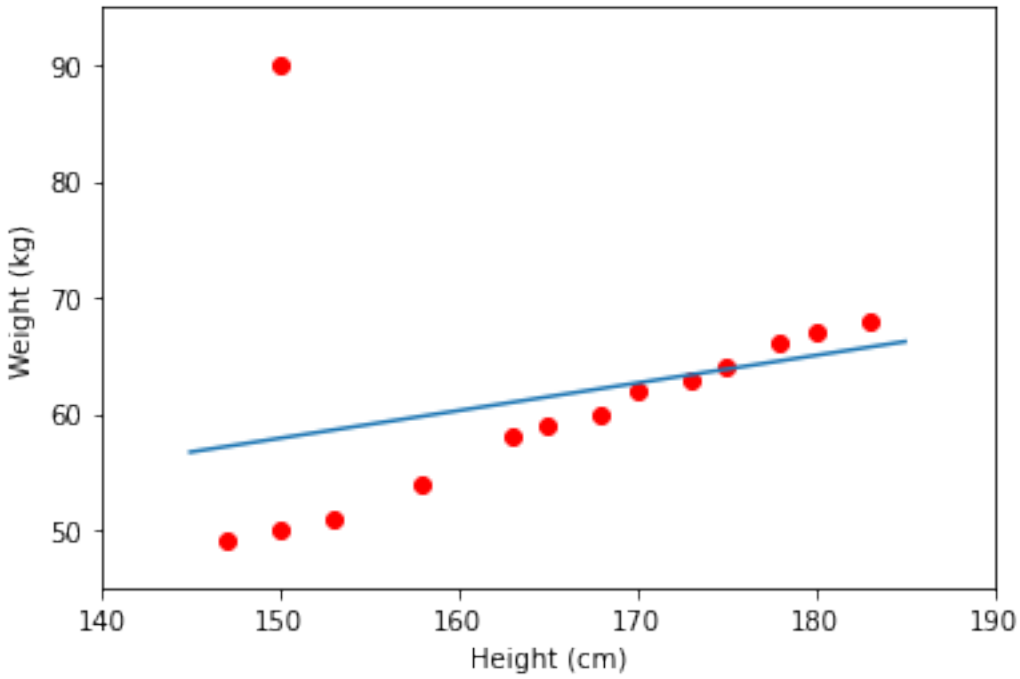
Như các bạn đã biết thì $y \approx f(x) = w^T x$ là một hàm tuyến tính phụ thuộc vào cả w và x. Trong thực tế thì chúng ta có thể áp dụng các mô hình tuyến tính chỉ phụ thuộc vào w.

Ví dụ: $y \approx w_1x_1 + w_2x_2 + w_3x_3 + w_4sin(x_2) + w_5x_1x_2 + w_0$ là một hàm tuyến tính phụ thuộc w, do đó, chúng ta có thể áp dụng hồi quy tuyến tính để giải bài toán này. Với mỗi dữ liệu đầu vào $x = [x_1, x_2, x_3]$, chúng ta tính toán dữ liệu mới $\bar{x} = [x_1, x_2, x_3, sin(x_2), x_1x_2]$



Nhược điểm

Hồi quy tuyến tính rất nhạy cảm với nhiễu (sensitive to noise).



Vì vậy, trước khi thực hiện Linear Regression, các nhiễu (outlier) cần phải được loại bỏ. Bước này được gọi là tiền xử lý (pre-processing).

Hồi quy tuyến tính không thể biểu diễn được với các mô hình phức tạp

Mặc dù trong phần trên, chúng ta thấy rằng phương pháp này có thể được áp dụng nếu quan hệ giữa outcome và input không nhất thiết phải là tuyến tính, nhưng mối quan hệ này vẫn đơn giản nhiều so với các mô hình thực tế.

Các phương pháp tối ưu

Linear Regression là một mô hình đơn giản, lời giải cho phương trình đạo hàm bằng 0 cũng khá đơn giản. Trong hầu hết các trường hợp, chúng ta không thể giải được phương trình đạo hàm bằng 0.

Nhưng có một điều chúng ta nên nhớ, còn tính được đạo hàm là còn có hy vọng.

Tài liệu tham khảo:

1. Brownlee, J. (2016, Mar 28). Simple Linear Regression Tutorial for Machine Learning. Machine Learning Mastery. https://machinelearningmastery.com/simple-linear-regression-tutorial-for-machine-learning/

2. Chapter 11: Least Squares, Pseudo-Inverses, PCA & SVD. (n.d). The University of Utah. http://www.sci.utah.edu/~gerig/CS6640-F2012/Materials/pseudoinverse-cis61009sl10.pdf

3. Quadratic Regression. (n.d). Varsity Tutors. https://www.varsitytutors.com/hotmath/hotmath_help/topics/quadratic-regression

4. Bài 3: Linear Regression. (2016, Dec 28). Machine Learning cơ bản blog. https://machinelearningcoban.com/2016/12/28/linearregression/

5. Framing: Key ML Terminology. (n.d). Machine Learning Crash Course. https://developers.google.com/machine-learning/crash-course/framing/ml-terminology

6. Descending into ML: Linear Regression. (n.d). Machine Learning Crash Course. https://developers.google.com/machine-learning/crash-course/descending-into-ml/linear-regression