

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

CƠ SỞ TẠI TP.HCM

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN

NĂM HỌC 2021

**NGHIÊN CỨU KỸ THUẬT HỒI QUY BÌNH PHƯƠNG TỐI THIỂU TRONG
NHÂN TIẾP TUYẾN THẦN KINH**

16-SV-2021-TH2

Thuộc nhóm ngành khoa học: Công nghệ thông tin

TP.Hồ Chí Minh, Tháng 10/2021

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

CƠ SỞ TẠI TP.HCM

BÁO CÁO TỔNG KẾT

ĐỀ TÀI NGHIÊN CỨU KHOA HỌC CỦA SINH VIÊN

NĂM HỌC 2021

**NGHIÊN CỨU KỸ THUẬT HỒI QUY BÌNH PHƯƠNG TỐI THIỂU TRONG
NHÂN TIẾP TUYẾN THÀNH KINH**

16-SV-2021-TH2

Thuộc nhóm ngành khoa học: Công nghệ thông tin

Sinh viên thực hiện: Phạm Đức Phú Phúc Nam, Nữ: Nam

Dân tộc: Kinh

Lớp, khoa: D18CQCP02-N

Năm thứ: 4 /Số năm đào tạo: 4,5

Ngành học: Công nghệ thông tin

Sinh viên tham gia : Nguyễn Văn Nhật

Người hướng dẫn: Th.S Nguyễn Thị Bích Nguyên

TP.Hồ Chí Minh, Tháng 10/2021

MỤC LỤC

MỞ ĐẦU.....	5
I. KỸ THUẬT HỒI QUY TUYẾN TÍNH	7
1. Mô hình hồi qui tuyến tính.....	7
2. Phương pháp đại số giải bài toán hồi quy tuyến tính.....	9
3. Phương trình hồi quy ước lượng. Ý nghĩa các hệ số hồi quy	9
4. Phương pháp đánh giá độ chính xác của mô hình hồi quy	10
4.1 Mối liên hệ giữa SST, SSR, SSE.....	10
4.2 Hệ số xác định.....	10
5. Biến độc lập định tính	11
6. Minh họa hồi quy tuyến tính qua ngôn ngữ lập trình Python	13
II. KỸ THUẬT HỒI QUY PHI TUYẾN TÍNH.....	20
1. Khái niệm	20
2. Một số loại hàm phi tuyến tính	20
2.1 Hàm bậc hai (Quadratic).....	20
2.2 Hàm mũ (Exponential)	21
2.3 Hàm logarit (Logarithm).....	21
2.4 Hàm Sigmoidal/Logistic	22
3. Ví dụ về bài toán hồi quy phi tuyến tính.....	22
3.1 Vẽ tập dữ liệu.....	22
3.2 Chọn một mô hình	23
3.3 Xây dựng mô hình	24
3.4 Huấn luyện mô hình.....	25
3.5 Đánh giá mô hình.....	26
III. CÁCH TÍNH HỒI QUY BÌNH PHƯƠNG TỐI THIỂU	28
1. Tổng quan về phương pháp bình phương tối thiểu	29

1.1 Diễn giải.....	29
1.2 Giải quyết.....	29
1.3 Bình phương tối thiểu có trọng số	30
1.4 Đường hồi qui bình phương tối thiểu	31
2. Cách tính hồi quy bình phương tối thiểu	31
2.1 Xấp xỉ bằng một hằng số	33
2.2 Xấp xỉ bằng một hàm tuyến tính.....	37
2.3 Xấp xỉ bởi một hàm bậc hai.....	41
2.4 Tổng kết	43
IV. SỬ DỤNG MỘT VÍ DỤ ĐƠN GIẢN ĐỂ GIÚP DỄ HIỂU HƠN Ý TƯỞNG VỀ GIÁ TRỊ HỒI QUY BÌNH PHƯƠNG TỐI THIỂU CỦA NHÂN TIẾP TUYẾN THẦN KINH.....	48
1. Ví dụ 1	48
2. Ví dụ 2.....	49
3. Ví dụ 3.....	50
4. Ví dụ 4.....	51
5. Ví dụ 5.....	53
KẾT LUẬN VÀ KIẾN NGHỊ.....	55
TÀI LIỆU THAM KHẢO.....	56

DANH MỤC HÌNH ẢNH

Hình I-1 Minh họa ứng dụng của kỹ thuật hồi quy tuyến tính.....	7
Hình II-1 Biểu diễn của mô hình hồi quy tuyến tính (trái) và hồi quy phi tuyến (phải) trong mặt phẳng tọa độ	20
Hình II-2 Minh họa đồ thị hàm bậc hai	21
Hình II-3 Minh họa đồ thị hàm mũ	21
Hình II-4 Minh họa đồ thị hàm logarit	22
Hình II-5 Minh họa đồ thị hàm Sigmoidal/Logistic	22

Hình II-6 Biểu diễn GDP của Trung Quốc từ năm 1960 đến năm 2014.....	23
Hình II-7 Đồ thị hàm Sigmoidal/Logistic	23
Hình II-8 Đồ thị biểu diễn mô hình ban đầu khi chưa huấn luyện.....	25
Hình II-9 Đồ thị biểu diễn mô hình sau khi huấn luyện.....	26
Hình III-1 Các phép đo độ lệch nhiệt độ trung bình hàng năm trên toàn cầu trong giai đoạn 1991-2000	32
Hình III-2 Đồ thị $F = F(\alpha)$	34
Hình III-3 Đồ thị $G = G(\alpha)$	36
Hình III-4 Hai giá trị xấp xỉ hằng số của các số đo độ lệch nhiệt độ trung bình hàng năm toàn cầu từ năm 1991 đến năm 2000.....	37
Hình III-5 Các xấp xỉ bình phương nhỏ nhất tuyến tính và xấp xỉ hằng của các số đo độ lệch nhiệt độ trung bình hàng năm toàn cầu từ năm 1991 đến năm 2000.....	40
Hình III-6 Các xấp xỉ hằng, tuyến tính và bậc hai của các số đo độ lệch nhiệt độ trung bình hàng năm toàn cầu từ năm 1991 đến năm 2000	43
Hình III-7 Các số đo độ lệch nhiệt độ trung bình hàng năm trên toàn cầu	45
Hình III-8 Các đường xấp xỉ bình phương nhỏ nhất không đổi, tuyến tính và bậc hai của các phép đo độ lệch nhiệt độ trung bình hàng năm trên toàn cầu; từ năm 1856 đến năm 2000	47
Hình IV-1 Minh họa sự thay đổi giá trị hàm khi thực hiện gradient descent trên hàm các giá trị riêng lẻ	48
Hình IV-2 Minh họa sự thay đổi giá trị hàm khi thực hiện gradient descent trên hàm tuyến tính	49
Hình IV-3 Minh họa sự thay đổi giá trị hàm khi thực hiện gradient descent trên hàm tuyến tính (sau khi tham số hóa lại hàm số).....	51
Hình IV-4 Minh họa sự thay đổi giá trị hàm khi thực hiện gradient descent trên hàm phi tuyến	52
Hình IV-5 Đồ thị biểu diễn độ thay đổi của giá trị hàm đối với lỗi dự đoán tại $x = 10$ (sau khi đã chuẩn hóa bằng cách chia cho độ lệch tại $x = 10$)	52
Hình IV-6 Minh họa sự thay đổi của hàm nhân qua 15 bước gradient descent.....	53
Hình IV-7 Minh họa sự thay đổi của hàm f_θ qua 15 bước gradient descent	53

DANH MỤC BẢNG BIỂU

Bảng III-1 Độ lệch nhiệt độ trung bình hàng năm toàn cầu đo được ở độ C trong những năm 1991-2000	32
---	----

MỞ ĐẦU

Trong những năm gần đây, lĩnh vực trí tuệ nhân tạo đã và đang phát triển mạnh mẽ. Trí tuệ nhân tạo len lỏi vào trong mọi lĩnh vực đời sống. Cùng với sự phát triển của nhân loại, các công nghệ cũng chạy theo với thời đại. Đặc biệt với thời đại 4.0 hiện nay, học máy (Machine Learning) và sử dụng trí tuệ nhân tạo áp dụng vào máy móc, các thiết bị hiện đại ra đời, giúp cho con người làm những việc tưởng chừng như khó có thể làm được, giúp tiết kiệm thời gian tiền bạc của con người. Những năm gần đây, khi mà khả năng tính toán của các máy tính được nâng lên một tầm cao mới và lượng dữ liệu khổng lồ được thu thập bởi các hãng công nghệ lớn, Machine Learning đã tiến thêm một bước dài và một lĩnh vực mới được ra đời gọi là Deep Learning (Học Sâu). Deep Learning đã giúp máy tính thực thi những việc tưởng chừng như không thể vào 10 năm trước: phân loại cả ngàn vật thể khác nhau trong các bức ảnh, tự tạo chú thích cho ảnh, bắt chước giọng nói và chữ viết của con người, giao tiếp với con người, hay thậm chí cả sáng tác văn hay âm nhạc.

Neural Network (mạng nơ-ron nhân tạo) là một chuỗi những thuật toán được đưa ra để tìm kiếm các mối quan hệ cơ bản trong tập hợp các dữ liệu. Thông qua việc bắt bước cách thức hoạt động từ não bộ con người.

Một loạt các bài báo gần đây về lý thuyết học sâu (Deep Learning) đề cập đến chủ đề chung là phân tích mạng nơ-ron trong giới hạn chiều rộng vô hạn (infinite-width). Lúc đầu, người ta thấy giới hạn này có vẻ không thực tế và thậm chí là vô nghĩa để nghiên cứu. Tuy nhiên, hóa ra các mạng nơ-ron trong chế độ này sẽ đơn giản hóa thành các mô hình tuyến tính (linear models) với một hạt nhân được gọi là nhân tiếp tuyến thần kinh (neural tangent kernel). Kỹ thuật gradient descent do đó sẽ rất đơn giản để nghiên cứu trong mạng này. Mặc dù điều này thoạt đầu nghe có vẻ hứa hẹn, nhưng kết quả thực nghiệm cho thấy rằng các mạng nơ-ron trong chế độ này hoạt động kém hơn so với các mạng được tham số hóa quá mức (over-parameterized) trong thực tế. Tuy nhiên, điều này vẫn cung cấp cái nhìn sâu sắc về mặt lý thuyết về một số khía cạnh nào đó của quá trình huấn luyện mạng nơ-ron, vì vậy nó rất đáng để nghiên cứu.

Qua bài báo cáo này, mục tiêu của chúng ta là nghiên cứu về một khía cạnh của nhân tiếp tuyến thần kinh, đó chính là hiểu và biết cách sử dụng *kỹ thuật hồi quy bình phương*

tối thiểu trong nhân tiếp tuyến thần kinh. Nội dung chính của báo cáo gồm có các phần sau đây :

- Kỹ thuật hồi quy tuyến tính
- Kỹ thuật hồi quy phi tuyến tính
- Cách tính hồi quy bình phương tối thiểu
- Sử dụng một ví dụ đơn giản để giúp dễ hiểu hơn ý tưởng về giá trị hồi quy bình phương tối thiểu của nhân tiếp tuyến thần kinh.

Phương pháp nghiên cứu cho bài báo cáo là thu thập thông tin khoa học trên cơ sở nghiên cứu các văn bản, tài liệu đã có và bằng các thao tác tư duy logic để rút ra kết luận khoa học cần thiết. Đối tượng nghiên cứu là kỹ thuật hồi quy bình phương tối thiểu, phạm vi nghiên cứu là nhân tiếp tuyến thần kinh (neural tangent kernel).

Vì đây là đề tài còn khá mới và không có nhiều tài liệu để nghiên cứu, ngoài ra đề tài đòi hỏi cần có nhiều kiến thức về Giải tích, Toán cao cấp,... cần thời gian trong việc tiếp cận ngôn ngữ, các thư viện, môi trường của ngôn ngữ lập trình..., trong khi thời gian nghiên cứu có hạn, nên trong quá trình nghiên cứu không thể tránh khỏi những sai sót. Nhóm em kính mong nhận được sự góp ý từ quý thầy cô để rút kinh nghiệm cho bài làm và các lần nghiên cứu khoa học sau.

Nhóm em xin chân thành cảm ơn cô Nguyễn Thị Bích Nguyên đã hướng dẫn chúng em trong quá trình thực hiện đề tài này. Chúng em cũng xin cảm ơn sự góp ý chân thành của các thầy cô khác để bổ sung cho bài nghiên cứu được tốt hơn.

I. KỸ THUẬT HỒI QUY TUYẾN TÍNH

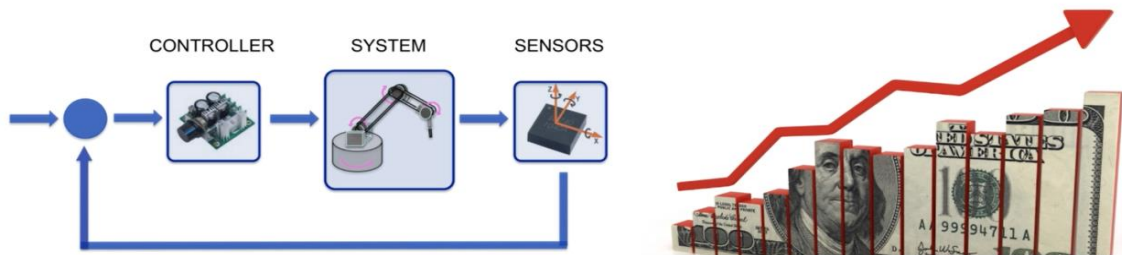
- Hồi quy tuyến tính (linear regression) là một phương pháp học máy có giám sát.
- Không chỉ trong học máy, nó còn được sử dụng trong nhiều lĩnh vực như: thống kê, kỹ thuật, kinh tế, tài chính, di truyền học, tâm lý học và nhiều hơn thế nữa...
- Nó cũng có thể là một phần quan trọng trong việc phân tích dữ liệu.

Not just for Machine Learning

Linear Regression can be found in many other fields:

Statistics, Engineering, Economics, Finance, Genetics, Psychology, +More!

Think of it like the swiss-army knife of data analysis



Hình I-1 Minh họa ứng dụng của kỹ thuật hồi quy tuyến tính

1. Mô hình hồi qui tuyến tính

Mô hình hồi qui tuyến tính là phương trình mô tả mối quan hệ giữa biến phụ thuộc y với các biến độc lập x_1, x_2, \dots, x_p và số hạng sai số ε :

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_p x_p + \varepsilon \quad (1)$$

Với:

$\beta_0, \beta_1, \beta_2, \dots, \beta_p$ là các tham số, và

ε là biến ngẫu nhiên gọi là số hạng sai số

Ví dụ 1-1: Khảo sát lương lập trình viên

Một công ty phần mềm thu thập dữ liệu của một mẫu gồm 20 lập trình viên. Người ta đề nghị sử dụng phân tích hồi qui để xác định xem lương có mối liên hệ với số năm kinh nghiệm và điểm thi năng khiếu về lập trình do công ty tổ chức hay không?

Số năm kinh nghiệm, điểm thi năng khiếu và mức lương hàng năm (\$1000s) của 20 lập trình viên được trình bày ở bảng sau:

<u>Exper.</u>	<u>Score</u>	<u>Salary</u>	<u>Exper.</u>	<u>Score</u>	<u>Salary</u>
4	78	24.0	9	88	38.0
7	100	43.0	2	73	26.6
1	86	23.7	10	75	36.2
5	82	34.3	5	81	31.6
8	86	35.8	6	74	29.0
10	84	38.0	8	87	34.0
0	75	22.2	4	79	30.1
1	80	23.1	6	94	33.9
6	83	30.0	3	70	28.2
6	91	33.0	3	89	30.0

Giả sử chúng ta tin rằng lương hàng năm (y) có mối liên hệ với số năm kinh nghiệm (x_1) và điểm thi năng khiếu (x_2) theo mô hình hồi qui sau:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \varepsilon \quad (2)$$

Với:

y : Lương hàng năm (\$1000)

x_1 : Số năm kinh nghiệm

x_2 : Điểm thi năng khiếu

Để tìm các hệ số của mô hình hồi quy ($\beta_0, \beta_1, \beta_2, \dots, \beta_p$) sao cho mô hình dự đoán kết quả khi biết đầu vào gần với các điểm dữ liệu, chúng ta có thể sử dụng kỹ thuật

gradient descent để tối ưu hóa loss function. Tuy nhiên để có thể đưa ra mô hình có độ chính xác cao nhất, chúng ta sẽ sử dụng phương pháp đại số.

2. Phương pháp đại số giải bài toán hồi quy tuyến tính

Gọi:

$w = [\beta_0, \beta_1, \beta_2, \dots, \beta_p]^T$ là vecto hệ số cần phải tối ưu

X là ma trận các tham số đầu vào (sau khi đã gắn thêm một cột gồm các số 1 vào bên trái)

y là ma trận các đầu ra tương ứng

Khi đó, người ta chứng minh được rằng vecto các hệ số của mô hình w sao cho mô hình hồi quy tối ưu nhất với dữ liệu được tính bởi công thức sau đây:

$$w = (X^T X)^{-1} X^T y \quad (3)$$

Ví dụ:

Trong bài toán về lương lập trình viên đã nêu, ta có:

$$X = \begin{bmatrix} 1 & 4 & 78 \\ 1 & 7 & 100 \\ 1 & 1 & 86 \\ \dots & \dots & \dots \\ 1 & 3 & 89 \end{bmatrix} \quad \text{và} \quad y = \begin{bmatrix} 24.0 \\ 43.0 \\ 23.7 \\ \dots \\ 30.0 \end{bmatrix}$$

Việc chúng ta cần làm chỉ là thay X và y vào công thức (3) và tính toán bằng máy tính, ta có được:

$$w = \begin{bmatrix} 3.174 \\ 1.404 \\ 0.251 \end{bmatrix}$$

3. Phương trình hồi quy ước lượng. Ý nghĩa các hệ số hồi quy

Thay các hệ số đã tìm được (trong vecto w) vào phương trình (2), ta có phương trình hồi quy ước lượng của bài toán lương lập trình viên:

$$\text{SALARY} = 3.174 + 1.404(\text{EXPER}) + 0.251(\text{SCORE})$$

Chúng ta có thể giải thích ý nghĩa của các hệ số đã tìm được như sau:

β_i là một ước lượng cho sự thay đổi của y ứng với sự gia tăng 1 đơn vị của x_i khi tất cả các biến độc lập được giữ không đổi.

Ví dụ:

$\beta_1 = 1.404$:

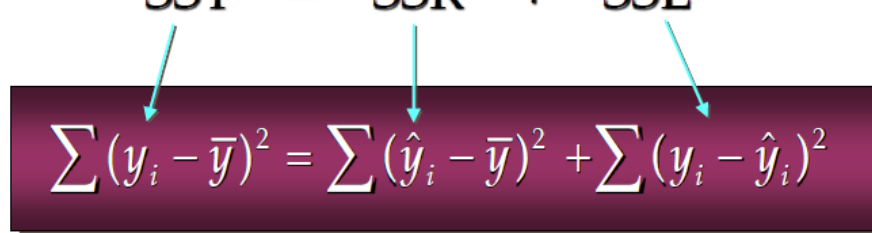
Lương được kỳ vọng tăng \$1,404 đối với mỗi 1 năm kinh nghiệm tăng thêm (khi điểm năng khiếu được giữ không đổi).

$\beta_2 = 0.251$:

Lương được kỳ vọng tăng \$251 đối với mỗi 1 điểm năng khiếu tăng thêm (khi số năm kinh nghiệm được giữ không đổi).

4. Phương pháp đánh giá độ chính xác của mô hình hồi quy

4.1 Mối liên hệ giữa SST, SSR, SSE

$$\text{SST} = \text{SSR} + \text{SSE}$$

$$\sum (y_i - \bar{y})^2 = \sum (\hat{y}_i - \bar{y})^2 + \sum (y_i - \hat{y}_i)^2$$

Trong đó:

SST : Tổng bình phương toàn phần

SSR : Tổng bình phương hồi qui

SSE : Tổng bình phương sai số

4.2 Hệ số xác định

$$R^2 = \text{SSR}/\text{SST} = 1 - \text{SSE}/\text{SST}$$

Lưu ý:

- Hệ số R^2 có giá trị tối đa là 1, thể hiện sự chính xác tuyệt đối của mô hình đối với dữ liệu đầu vào.

- Khác với khái niệm độ chính xác thông thường (giá trị chỉ trong đoạn 0 – 1), hệ số R^2 có thể có giá trị âm, lúc này ta có thể hiểu rằng mô hình hồi quy rất không chính xác.

Ví dụ: Trong bài toán tính lương lập trình viên ở trên:

$$R^2 = 500.3285 / 599.7855 = 0.83418$$

5. Biến độc lập định tính

Trong nhiều tình huống thực tiễn, ngoài các biến số có giá trị liên tục như số năm kinh nghiệm, số điểm năng khiếu như trong ví dụ trên, chúng ta có thể còn phải sử dụng các biến định tính như giới tính (Nam, Nữ), vùng miền (Bắc, Trung, Nam)

Ví dụ, ta có thể có một biến x đại diện cho giới tính với $x = 0$ để chỉ Nam và $x = 1$ để chỉ Nữ.

Trong trường hợp này x được gọi là biến giả, biến chỉ thị hay biến thuộc tính.

Ví dụ 1-2: Mở rộng vấn đề khảo sát lương lập trình viên trong ví dụ trước.

Giả sử về mặt quản lý, người ta tin rằng lương hằng năm có liên quan đến cá nhân có bằng tốt nghiệp về khoa học máy tính hay hệ thống thông tin.

Dữ liệu về Số năm kinh nghiệm, Điểm thi năng khiếu, Bằng cấp chuyên môn và lương hằng năm (\$1000) của mẫu gồm 20 lập trình viên được trình bày như sau:

<u>Exper.</u>	<u>Score</u>	<u>Degr.</u>	<u>Salary</u>	<u>Exper.</u>	<u>Score</u>	<u>Degr.</u>	<u>Salary</u>
4	78	No	24.0	9	88	Yes	38.0
7	100	Yes	43.0	2	73	No	26.6
1	86	No	23.7	10	75	Yes	36.2
5	82	Yes	34.3	5	81	No	31.6
8	86	Yes	35.8	6	74	No	29.0
10	84	Yes	38.0	8	87	Yes	34.0
0	75	No	22.2	4	79	No	30.1
1	80	No	23.1	6	94	Yes	33.9
6	83	No	30.0	3	70	No	28.2
6	91	Yes	33.0	3	89	No	30.0

Nhận xét: So với ví dụ trước, dữ liệu của chúng ta có thêm một cột cho biết lập trình viên đó có bằng cấp chuyên môn hay không (Yes: có, No: không)

Phương trình hồi quy của bài toán lúc này được viết lại thành:

$$y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3$$

Với:

y : Lương hàng năm (\$1000)

x_1 : Số năm kinh nghiệm

x_2 : Điểm thi năng khiếu

x_3 : **0** nếu **không có** bằng cấp chuyên môn, **1** nếu **có** bằng cấp chuyên môn

Biến x_3 của chúng ta trong trường hợp này chính là biến giả, biến chỉ thị hay biến thuộc tính.

Việc tìm hệ số tối ưu cho mô hình hồi quy hoàn toàn tương tự như ví dụ trước. Lưu ý là đối với cột Bằng cấp chuyên môn, ta cần chuyển các giá trị định tính về giá trị số để máy tính có thể tính toán được (Yes: 1, No: 0)

Ta có:

$$X = \begin{bmatrix} 1 & 4 & 78 & 0 \\ 1 & 7 & 100 & 1 \\ 1 & 1 & 86 & 0 \\ \dots & \dots & \dots & \dots \\ 1 & 3 & 89 & 0 \end{bmatrix} \quad \text{và} \quad y = \begin{bmatrix} 24.0 \\ 43.0 \\ 23.7 \\ \dots \\ 30.0 \end{bmatrix}$$

Thay X và y vào công thức (3) và tính toán bằng máy tính, ta có được:

$$w = \begin{bmatrix} 7.94485 \\ 1.1475 \\ 0.19694 \\ 2.28042 \end{bmatrix}$$

Ý nghĩa của các hệ số β_1 và β_2 (các hệ số tương ứng với đầu vào là số năm kinh nghiệm và số điểm năng khiếu) tương tự như trong ví dụ trước. Ta chỉ giải thích ý nghĩa của hệ số β_3 (hệ số tương ứng với đầu vào là có bằng cấp chuyên môn hay không)

$\beta_3 = 2.28042$:

Lương được kỳ vọng tăng \$2,280 nếu có bằng tốt nghiệp về khoa học máy tính hay hệ thống thông tin, ngược lại thì không tăng. (khi số năm kinh nghiệm và điểm năng khiếu được giữ không đổi).

Để đánh giá độ chính xác của mô hình với các hệ số trên, ta cũng tính hệ số xác định R^2 . Bằng máy tính ta tính được $R^2 = 0.846796$.

6. Minh họa hồi quy tuyến tính qua ngôn ngữ lập trình Python

- Xây dựng lớp biểu diễn mô hình hồi quy tuyến tính:

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

class LinearRegression:

    # Huấn Luyện mô hình bằng cách sử dụng kỹ thuật gradient descent
    def fit_grad(self, X, y, numOfIteration=None, learning_rate=1e-
6, threshold=1e-3, show_loss=False):
        dim = X.shape[1]
        self.w = np.ones(dim+1).reshape(-1, 1)
        N = X.shape[0]
        X = np.hstack((np.ones((N, 1)), X))

        # Trường hợp cho biết số lần lặp xác định
        if numOfIteration != None:
            cost = np.zeros((numOfIteration+1,1))
            r = np.dot(X, self.w) - y
            cost[0] = 0.5*np.sum(r*r)
            for epoch in range(1, numOfIteration+1):
                self.w[0] -= learning_rate*np.sum(r)
                for i in range(1, dim+1):
                    self.w[i] -= learning_rate *
np.sum(np.multiply(r, X[:,i].reshape(-1,1)))
                r = np.dot(X, self.w) - y
```

```

        cost[epoch] = 0.5*np.sum(r*r)
        # Trường hợp không cho biết số lần lặp: Lặp đến khi thay đổi kh
        ông đáng kể
    else:
        numOfIteration = 0
        cost = []
        r = np.dot(X, self.w) - y
        cost.append(0.5*np.sum(r*r))
        while True:
            self.w[0] -= learning_rate*np.sum(r)
            for i in range(1, dim+1):
                self.w[i] -= learning_rate *
np.sum(np.multiply(r, X[:,i].reshape(-1,1)))
            numOfIteration += 1
            r = np.dot(X, self.w) - y
            cost.append(0.5*np.sum(r*r))
            if cost[-2] - cost[-1] <= threshhold:
                break

        # Biểu diễn giá trị hàm loss trong quá trình huấn luyện
        if show_loss:
            plt.plot(np.arange(0, numOfIteration+1).reshape(-
1, 1), cost)
            plt.title('Giá trị hàm loss trong quá trình huấn luyện')
            plt.xlabel('Epoch #')
            plt.ylabel('Loss');
            plt.show()

        # Huấn Luyện mô hình bằng phương pháp đại số
    def fit(self, X, y):
        N = X.shape[0]
        X = np.hstack((np.ones((N, 1)), X))
        self.w = np.linalg.pinv(X.T @ X) @ (X.T @ y)

        # Dự đoán
    def predict(self, X):

```



```

N = X.shape[0]
X = np.hstack((np.ones((N, 1)), X))
return np.dot(X, self.w)

# Đánh giá mô hình bằng hệ số xác định R2
def score(self, X, y):
    #  $r^2 = SSR/SST = 1 - SSE/SST$ 
    y_pred = self.predict(X)
    sse = ((y - y_pred) ** 2).sum()
    sst = ((y - y.mean()) ** 2).sum()
    return 1 - sse/sst

# Biểu diễn mô hình lên đồ thị
def show(self, X, y, labels=['', '', '']):
    dim = X.shape[1]
    if dim == 1:
        plt.scatter(X, y)
        y_hat = self.predict(X)
        N = X.shape[0]
        plt.plot((X[0], X[N-1]), (y_hat[0], y_hat[N-1]), 'r')
        plt.xlabel(labels[0])
        plt.ylabel(labels[1])
        plt.show()
    elif dim == 2:
        m = int(max(X[:, 0].max(), X[:, 1].max()))
        # create x, y
        xx, yy = np.meshgrid(range(m), range(m))
        zz = np.hstack((xx.reshape(-1, 1), yy.reshape(-1, 1)))
        zz = self.predict(zz).reshape(m, m)
        # plot the surface
        plt3d = plt.figure().gca(projection='3d')
        plt3d.scatter(X[:,0], X[:,1], y, c='r', marker='o')
        plt3d.plot_surface(xx,yy,zz, color='blue')
        plt3d.set_xlabel(labels[0])
        plt3d.set_ylabel(labels[1])

```

```

plt3d.set_zlabel(labels[2])
plt.show()
else:
    print('Chỉ hỗ trợ vẽ mô hình có 1 hoặc 2 tham số đầu vào!')

# Lưu mô hình
def save(self, filename='LinearRegression.npy'):
    np.save(filename, self.w)

# Tải mô hình đã có sẵn
def load(self, filename='LinearRegression.npy'):
    self.w = np.load(filename)

```

- Các phương thức tiện ích hỗ trợ cho việc tách dữ liệu thành tham số đầu vào và nhãn, và chuyển list thành ma trận:

```

import numpy as np
import pandas as pd

def input_label_split(**kwargs):
    filepath = kwargs.get('filepath')
    df = kwargs.get('df')

    if filepath != None:
        df = pd.read_csv(filepath)

    headers = df.columns.values.tolist()
    data = df.values
    d = data.shape[1]
    x = data[:, :d-1].reshape(-1, d-1)

```

```
y = data[:, d-1].reshape(-1, 1)

return x, y, headers
```

```
def to_matrix(lst):

    return np.array(lst).reshape(-1, 1)
```

Bây giờ ta sẽ thử các module đã viết trên sổ tay.

- Thêm các thư viện đã viết ở trên

```
import pandas as pd

from model.LinearRegression import *
from utils.Functions import *
```

- Lấy thông tin lương nhân viên trong ví dụ 1-1

```
df_luong = pd.read_csv('dataset/luong.csv')
df_luong
```

- Xây dựng và huấn luyện mô hình theo 2 cách

```
# Tự xây dựng mô hình
model = LinearRegression()
X, y, headers = input_label_split(df=df_luong)
# model.fit_grad(X, y, numOfIteration=None, Learning_rate=1e-5, threshold=1e-3, show_loss=True)
model.fit(X, y)
```

```
# Sử dụng mô hình có sẵn trong thư viện scikit-learn
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(X, y);
```

- Hiển thị các giá trị như tham số mô hình, hệ số xác định theo 2 cách để so sánh:

```
model.w
lr.intercept_, lr.coef_
model.score(X, y)
lr.score(X, y)
```

Kết quả:

```
array([[3.17393627], [1.40390249], [0.25088545]])

(array([3.17393627]), array([[1.40390249, 0.25088545]]))

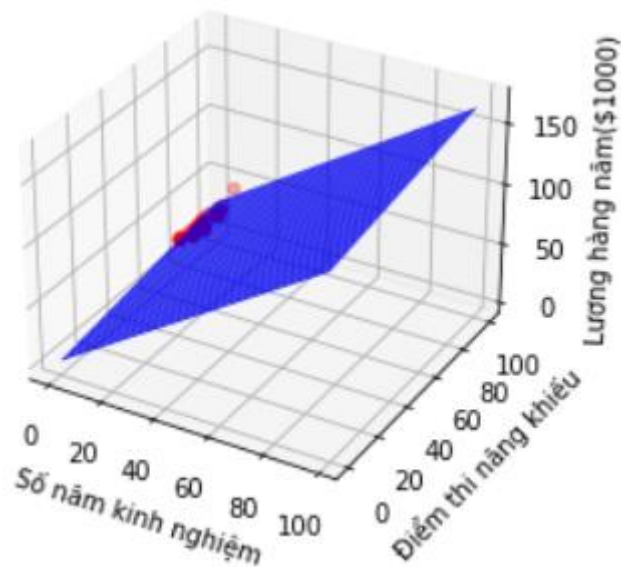
0.8341791028866659

0.8341791028866657
```

- Trực quan hóa mô hình bằng cách biểu diễn lên không gian 3 chiều:

```
model.show(X, y, labels=headers)
```

Kết quả:



- Lấy thông tin lương nhân viên trong ví dụ 1-2, đồng thời thay thế các giá trị định tính thành giá trị số tương ứng (No:0 / Yes:1) vì chúng ta chỉ làm việc với dữ liệu số:

```
df_luong2 = pd.read_csv('dataset/luong2.csv')
```

```
df_luong2['Bằng cấp chuyên môn'].replace({'No': 0, 'Yes': 1}, inplace =
True)

df_luong2
```

- Xây dựng và huấn luyện mô hình theo 2 cách

```
X, y, _ = input_label_split(df=df_luong2)
# model.fit_grad(X, y, numOfIteration=None, learning_rate=1e-5,
threshold=1e-3, show_loss=True)
model.fit(X, y)
lr.fit(X, y);
```

- Hiện thị các giá trị như ở trên để so sánh:

```
model.w
lr.intercept_, lr.coef_
model.score(X, y)
lr.score(X, y)
```

Kết quả:

```
array([[7.94484872], [1.14758173], [0.19693699], [2.28042384]])

(array([7.94484872]), array([1.14758173, 0.19693699, 2.28042384]))

0.8467960853151655

0.8467960853151653
```

Nhận xét:

- Các giá trị như tham số mô hình, hệ số xác định sau khi huấn luyện theo 2 cách đều giống nhau.

- Giá trị R^2 của mô hình khá cao (cả 2 ví dụ đều trên 0.8).

- Nếu huấn luyện mô hình bằng cách sử dụng kỹ thuật gradient descent sẽ cho kết quả xấp xỉ, nhưng với hệ số xác định không cao bằng. Điều đó chứng tỏ rằng phương pháp đại số cho kết quả chính xác hơn so với phương pháp sử dụng kỹ thuật gradient descent.

II. KỸ THUẬT HỒI QUY PHI TUYẾN TÍNH

Nếu dữ liệu cho thấy một xu hướng cong, thì hồi quy tuyến tính sẽ không tạo ra kết quả chính xác khi so sánh với hồi quy phi tuyến tính, bởi vì như tên của nó, hồi quy tuyến tính cho rằng dữ liệu là tuyến tính.

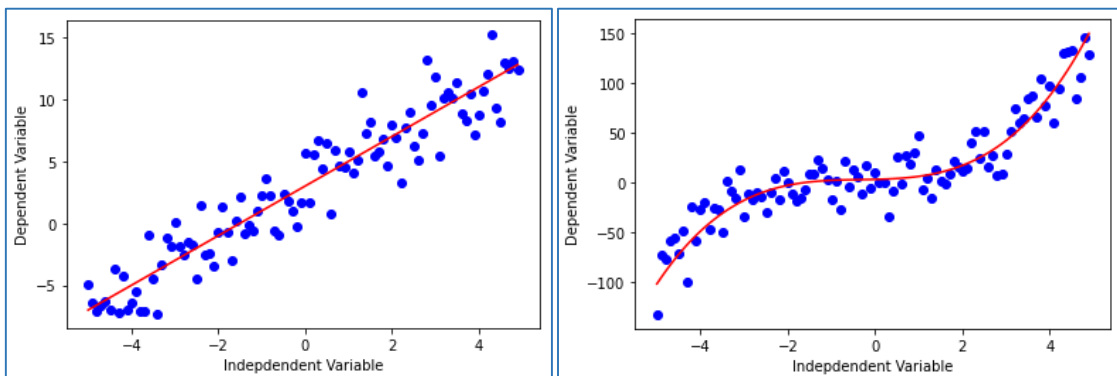
Bây giờ chúng ta hãy cùng tìm hiểu về hồi quy phi tuyến tính và áp dụng vào một ví dụ trên ngôn ngữ python.

1. Khái niệm

Hồi quy phi tuyến tính (non-linear regression) là mối quan hệ giữa các biến độc lập x và một biến phụ thuộc y , kết quả là dữ liệu được mô hình hóa bởi một hàm phi tuyến tính.

Về cơ bản, bất kỳ mối quan hệ nào không tuyến tính đều có thể được gọi là phi tuyến tính.

Giả sử nếu đầu vào của bài toán hồi quy chỉ có một đặc trưng, thì mô hình hồi quy có thể được biểu diễn trong mặt phẳng 2D. Nếu trong mặt phẳng tọa độ, mô hình hồi quy tuyến tính được biểu diễn bởi một đường thẳng thì mô hình hồi quy phi tuyến sẽ được biểu diễn bởi một đường cong.



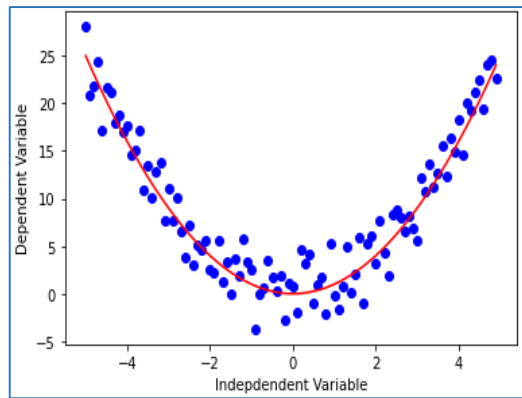
Hình II-1 Biểu diễn của mô hình hồi quy tuyến tính (trái) và hồi quy phi tuyến (phải) trong mặt phẳng tọa độ

2. Một số loại hàm phi tuyến tính

2.1 Hàm bậc hai (Quadratic)

Biểu thức:

$$Y = X^2$$



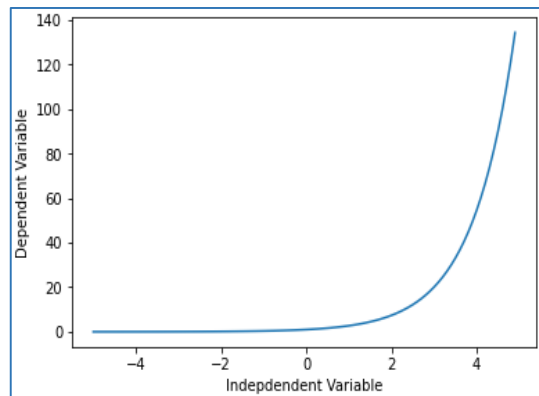
Hình II-2 Minh họa đồ thị hàm bậc hai

2.2 Hàm mũ (Exponential)

Biểu thức:

$$Y = a + bc^x$$

trong đó $b \neq 0$, $c > 0$, $c \neq 1$, và x là một số thực bất kỳ. Cơ số c là hằng số và số mũ x là một biến



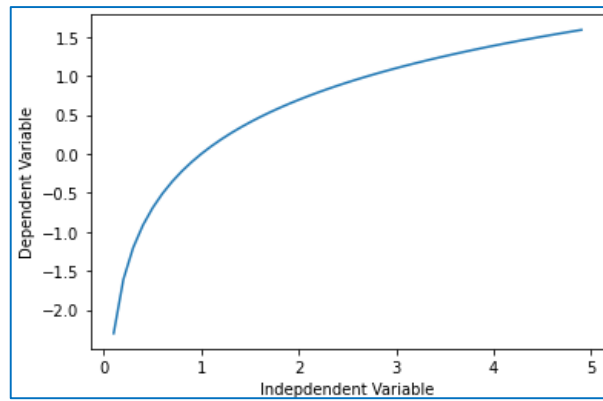
Hình II-3 Minh họa đồ thị hàm mũ

2.3 Hàm logarit (Logarithm)

Biểu thức:

$$Y = \log(X)$$

trong đó X là biểu diễn đa thức của x .

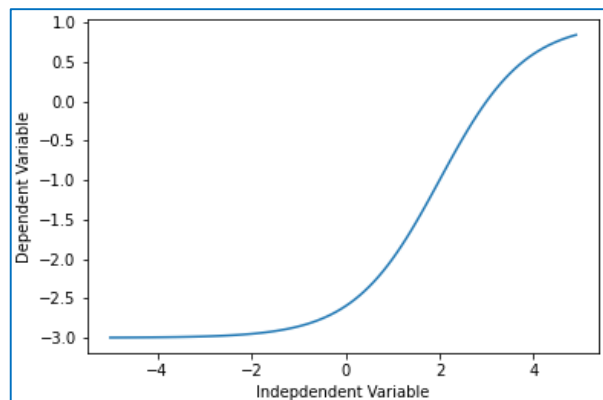


Hình II-4 Minh họa đồ thị hàm logarit

2.4 Hàm Sigmoidal/Logistic

Biểu thức:

$$Y = a + \frac{b}{1 + c^{(x-d)}}$$



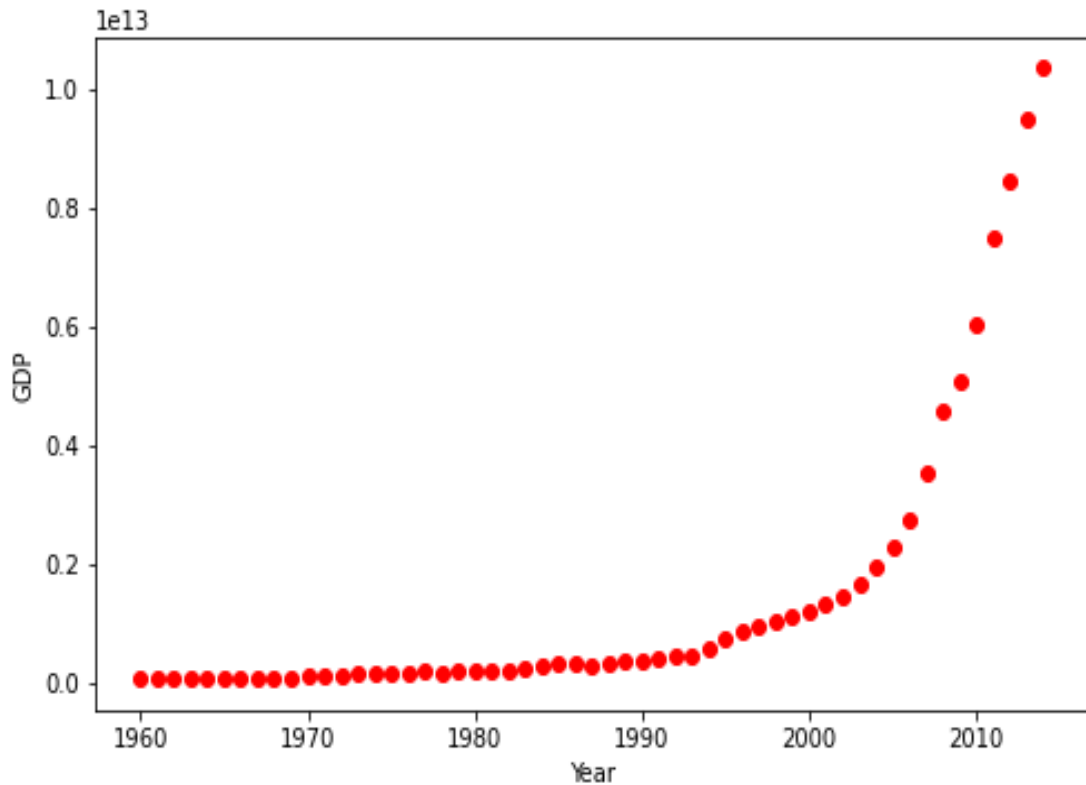
Hình II-5 Minh họa đồ thị hàm Sigmoidal/Logistic

3. Ví dụ về bài toán hồi quy phi tuyến tính

Chúng ta sẽ thử huấn luyện một mô hình phi tuyến tính với các điểm dữ liệu tương ứng với GDP của Trung Quốc từ năm 1960 đến năm 2014. Chúng ta tải xuống tập dữ liệu có hai cột: cột đầu tiên là một giá trị năm từ 1960 đến 2014, cột thứ hai là tổng thu nhập quốc nội hàng năm tương ứng của Trung Quốc tính bằng đô la Mỹ trong năm đó.

3.1 Vẽ tập dữ liệu

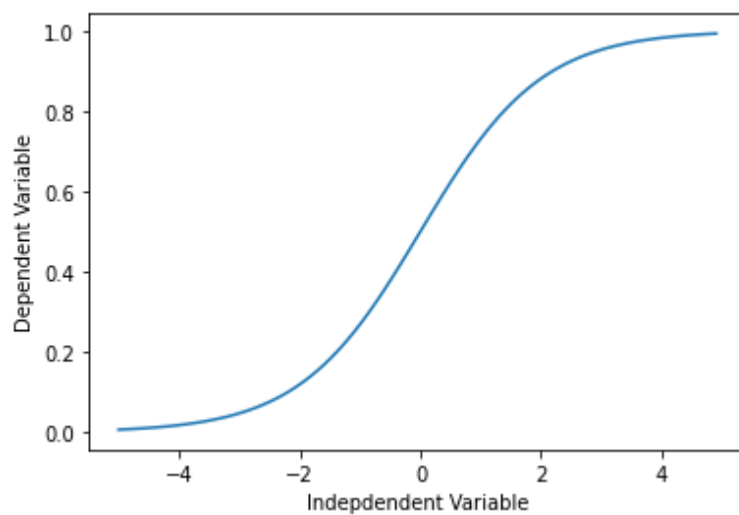
Đây là hình ảnh các điểm dữ liệu trông như thế nào. Nó giống như một hàm logistic hoặc hàm mũ. Sự tăng trưởng bắt đầu chậm, sau đó từ năm 2005 trở đi, sự tăng trưởng rất đáng kể. Và cuối cùng, nó giảm tốc một chút trong những năm 2010.



Hình II-6 Biểu diễn GDP của Trung Quốc từ năm 1960 đến năm 2014

3.2 Chọn một mô hình

Từ cái nhìn ban đầu về biểu đồ trên, chúng ta xác định rằng hàm logistic có thể là một xấp xỉ tốt, vì nó có đặc tính là bắt đầu với tốc độ tăng trưởng chậm, tăng trưởng tăng ở giữa và sau đó giảm lại ở cuối, như minh họa bên dưới:



Hình II-7 Đồ thị hàm Sigmoidal/Logistic

Công thức cho hàm logistic như sau:

$$Y = \frac{1}{1 + e^{\beta_1(X - \beta_2)}}$$

Trong đó:

β_1 : Kiểm soát độ dốc của đường cong

β_2 : Trượt đường cong trên trục x.

3.3 Xây dựng mô hình

Bây giờ, hãy xây dựng mô hình hồi quy của chúng ta và khởi tạo các tham số của nó.

```
def sigmoid(x, Beta_1, Beta_2):  
    y = 1 / (1 + np.exp(Beta_1*(x-Beta_2)))  
    return y
```

Hãy xem thử một đường sigmoid mẫu có thể phù hợp với dữ liệu:

```
beta_1 = -0.10
```

```
beta_2 = 1990.0
```

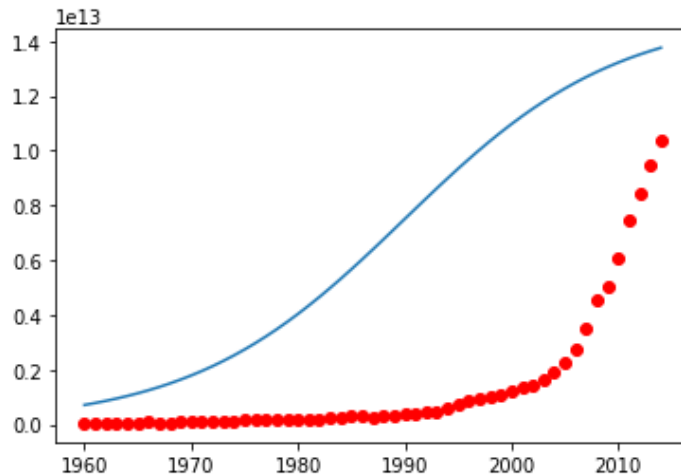
```
#logistic function
```

```
Y_pred = sigmoid(x_data, beta_1 , beta_2)
```

```
#plot initial prediction against datapoints
```

```
plt.plot(x_data, Y_pred*15e12)
```

```
plt.plot(x_data, y_data, 'ro')
```



Hình II-8 Đồ thị biểu diễn mô hình ban đầu khi chưa huấn luyện

Nhiệm vụ của chúng ta bây giờ là tìm ra các thông số tốt nhất cho mô hình của chúng ta. Nhưng trước tiên chúng ta cần chuẩn hóa x và y:

```
# Lets normalize our data
xdata = x_data/max(x_data)
ydata = y_data/max(y_data)
```

3.4 Huấn luyện mô hình

Làm thế nào để tìm ra các thông số tốt nhất cho đường huấn luyện của chúng ta?

Chúng ta có thể sử dụng **curve_fit**, hàm này sử dụng kỹ thuật bình phương nhỏ nhất phi tuyến tính để điều chỉnh hàm sigmoid của chúng ta gần với dữ liệu, tối ưu các giá trị cho các tham số để tổng các phần dư bình phương của sigmoid (xdata, *popt) - ydata được giảm thiểu.

```
from scipy.optimize import curve_fit
popt, pcov = curve_fit(sigmoid, xdata, ydata)
#print the final parameters
print(" beta_1 = %f, beta_2 = %f" % (popt[0], popt[1]))
```

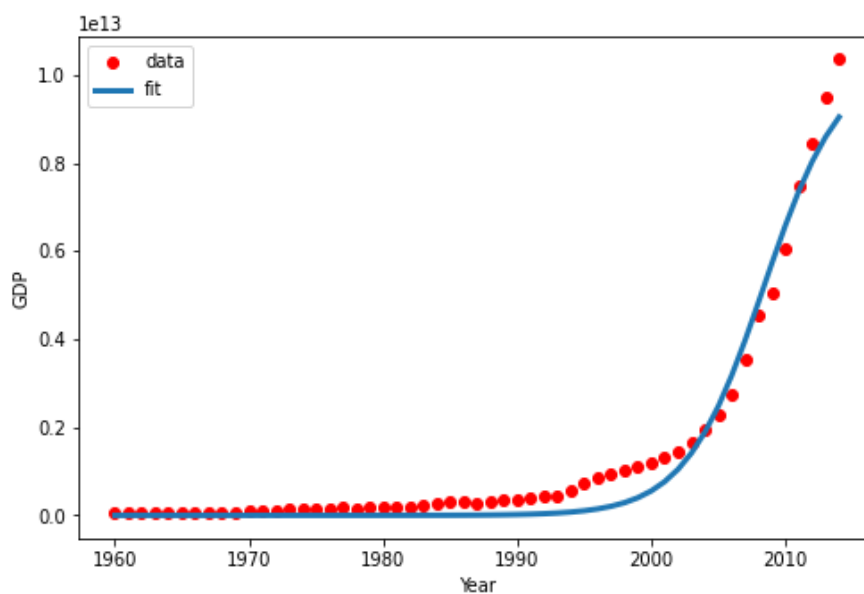
Chúng ta có thể in ra kết quả các tham số của mô hình sau khi huấn luyện:

```
beta_1 = -690.450283, beta_2 = 0.997207
```

Bây giờ chúng ta vẽ lại mô hình hồi quy và kết quả như hình bên dưới:

```
y_hat = sigmoid(xdata, *popt)
y_hat *= max(y_data)

plt.figure(figsize=(8,5))
plt.plot(x_data, y_data, 'ro', label='data')
plt.plot(x_data, y_hat, linewidth=3.0, label='fit')
plt.legend(loc='best')
plt.ylabel('GDP')
plt.xlabel('Year')
plt.show()
```



Hình II-9 Đồ thị biểu diễn mô hình sau khi huấn luyện

3.5 Đánh giá mô hình

Chúng ta có thể tính toán độ chính xác của mô hình là bao nhiêu không?

Chúng ta sẽ chia tập dữ liệu đầu vào thành 2 tập dữ liệu, một tập dùng cho việc huấn luyện (train), tập còn lại dùng để kiểm tra độ chính xác của mô hình (test)

Chúng ta sẽ đánh giá mô hình thông qua 3 tiêu chí, đó là:

- Trung bình sai số tuyệt đối (Mean Absolute Error - MAE):

MAE được tính bằng tổng các trị tuyệt đối giữa giá trị thực (y_i : target) và giá trị mà mô hình của chúng ta dự đoán (\hat{y}_i : predicted).

- Trung bình sai số bình phương (Mean Square Error – MSE)

MSE được tính bằng tổng các bình phương của hiệu giữa giá trị thực (y_i : target) và giá trị mà mô hình của chúng ta dự đoán (\hat{y}_i : predicted).

- Hệ số xác định R^2 (về cách tính đã được đề cập trong chương I – Kỹ thuật hồi quy tuyến tính)

```
from sklearn.model_selection import train_test_split
train_x, test_x, train_y, test_y = train_test_split(xdata, ydata, test_size=0.25, random_s
tate=42)
```

```
# build the model using train set
```

```
popt, pcov = curve_fit(sigmoid, train_x, train_y)
```

```
# predict using test set
```

```
y_hat = sigmoid(test_x, *popt)
```

```
# evaluation
```

```
print("Mean absolute error: %f" % np.mean(np.absolute(y_hat - test_y)))
```

```
print("Residual sum of squares (MSE): %f" % np.mean((y_hat - test_y) ** 2))
```

```
from sklearn.metrics import r2_score
```

```
print("R2-score: %f" % r2_score(y_hat, test_y))
```

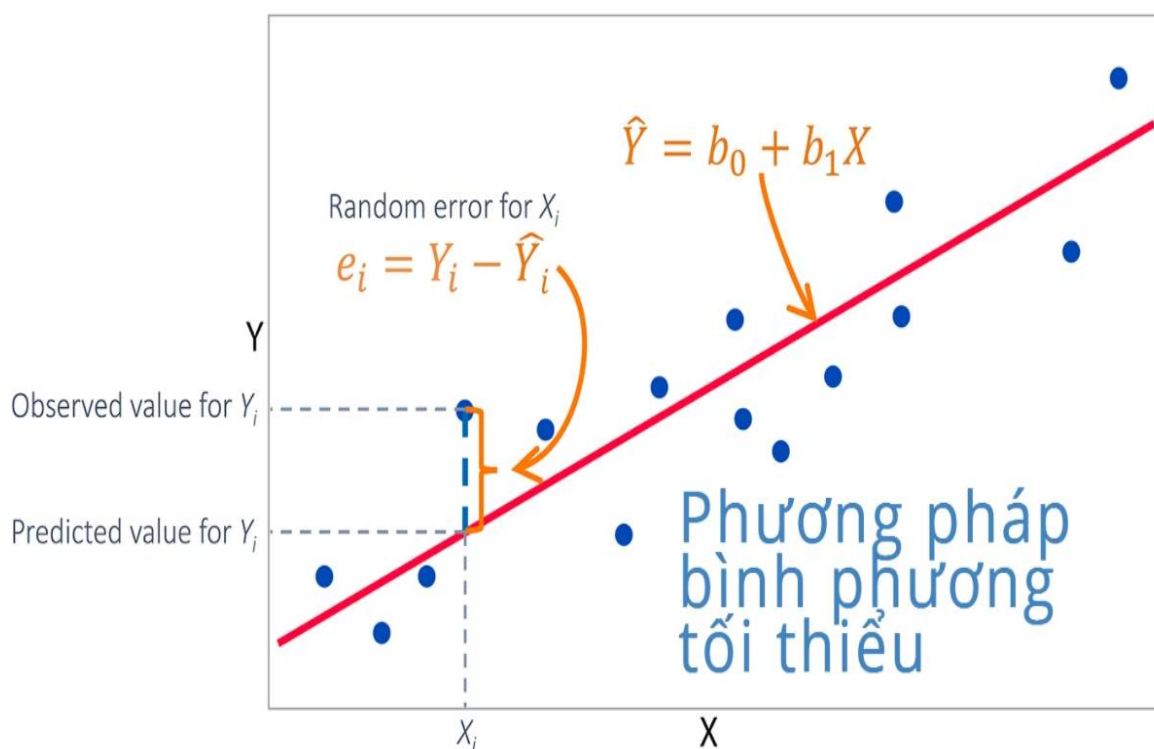
Kết quả:

Mean absolute error: 0.031038

Residual sum of squares (MSE): 0.001318

R2-score: 0.976240

III. CÁCH TÍNH HỒI QUY BÌNH PHƯƠNG TỐI THIỂU



Trong toán học, **phương pháp bình phương tối thiểu (Ordinary least square)**, còn gọi là **bình phương nhỏ nhất** hay **bình phương trung bình tối thiểu**, là một phương pháp tối ưu hóa để lựa chọn một đường khớp nhất cho một dải dữ liệu ứng với cực trị của tổng các sai số thống kê (error) giữa đường khớp và dữ liệu.

Phương pháp này giả định các sai số (error) của phép đo đặc dữ liệu phân phối ngẫu nhiên. Định lý Gauss-Markov chứng minh rằng kết quả thu được từ phương pháp bình phương tối thiểu không thiên vị và sai số của việc đo đặc dữ liệu không nhất thiết phải tuân theo, ví dụ, phân bố Gauss. Một phương pháp mở rộng từ phương pháp này là **bình phương tối thiểu có trọng số**.

Phương pháp bình phương tối thiểu thường được dùng trong khớp đường cong. Nhiều bài toán tối ưu hóa cũng được quy về việc tìm cực trị của dạng bình phương, ví dụ như tìm cực tiểu của năng lượng hay cực đại của entropy.

Khái niệm bình phương cực tiểu bắt nguồn từ công trình tiên phong của Gauss và Legendre trong khoảng đầu thế kỷ 19. Có nguồn khác cho rằng nhà toán học và nhà khoa học người Đức, Carl Friedrich Gauss là người đã phát hiện ra phương pháp bình

phương tối thiểu vào năm 1795. Bình phương cực tiểu được sử dụng nhiều trong thống kê hiện đại và mô hình toán học.

Các bài toán có nhu cầu sử dụng phương pháp bình phương cực tiểu: giải hệ phương trình, tìm đường cong phù hợp nhất ứng với dải dữ liệu cho trước (curve fitting), tìm phương trình hồi quy trong thống kê...

1. Tổng quan về phương pháp bình phương tối thiểu

1.1 Diễn giải

Giả sử dữ liệu gồm các điểm (x_i, y_i) với $i = 1, 2, \dots, n$. Chúng ta cần tìm một hàm số f thỏa mãn:

$$f(x_i) \approx y_i$$

Giả sử hàm f có thể thay đổi hình dạng, phụ thuộc vào các tham số p_j với $j = 1, 2, \dots, m$.

Nội dung của phương pháp là tìm giá trị của các tham số p_j sao cho biểu thức sau đạt cực tiểu:

$$\chi^2 = \sum_{i=1}^n (y_i - f(x_i))^2$$

Nội dung này giải thích tại sao tên gọi của phương pháp là *bình phương tối thiểu*.

Đôi khi thay vì tìm giá trị nhỏ nhất của tổng bình phương, người ta có thể tìm giá trị nhỏ nhất của bình phương trung bình:

$$\chi^2 = \frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$$

Lúc này phương pháp có tên gọi *bình phương trung bình tối thiểu*.

1.2 Giải quyết

Bài toán thường có lời giải đáng tin cậy khi số lượng các tham số p_j nhỏ hơn số lượng các dữ liệu ($m < n$).

Trong trường hợp, f là hàm tuyến tính của các tham số p_j , bài toán trở nên đơn giản hóa rất nhiều, rút gọn thành việc giải một hệ phương trình tuyến tính (bình phương tối thiểu tuyến tính).

Nếu f không là hàm tuyến tính của các tham số, bài toán trở thành một bài toán tối ưu hóa tổng quát. Bài toán tổng quát này có thể dùng các phương pháp như phương pháp tối ưu hóa Newton hay phương pháp trượt dốc (gradient descent). Đặc biệt thuật toán Gauss-Newton hay thuật toán Levenberg-Marquardt là thích hợp nhất cho bài toán bình phương tối thiểu tổng quát này.

1.3 Bình phương tối thiểu có trọng số

Trong các ngành thống kê và tối ưu hóa, *bình phương tối thiểu có trọng số* là một phương pháp mở rộng của bình phương tối thiểu.

Mục đích của phương pháp, tương tự như bình phương tối thiểu, là tìm hàm $f(x)$ khớp nhất với các dữ liệu đo đạc (x_i, y_i) với $i = 1, 2, \dots, n$:

$$f(x_i) \approx y_i$$

Hàm f phụ thuộc vào các tham số p_j với $j = 1, 2, \dots, m$. Tương tự như bình phương tối thiểu, nội dung của phương pháp là tìm các tham số p_j để thu được một biểu thức thể hiện sai khác nhỏ nhất giữa f và dữ liệu. Khác với bình phương tối thiểu, trong phương pháp bình phương tối thiểu có trọng số, thay vì dùng:

$$\chi^2 = \sum_{i=1}^n (y_i - f(x_i))^2$$

Người ta thêm vào biểu thức các "trọng số" thể hiện độ tin cậy của từng điểm đo đạc:

$$\chi^2 = \sum_{i=1}^n w_i (y_i - f(x_i))^2$$

Thông thường, w_i được đặt bằng nghịch đảo phương sai của từng điểm đo:

$$w_i = 1/\sigma_i^2$$

Nghĩa là điểm đo đạc càng chính xác (phương sai càng nhỏ) thì có trọng số càng cao trong biểu thức.

1.4 Đường hồi qui bình phương tối thiểu

Đường biểu diễn phù hợp nhất được xác định bởi phương pháp bình phương tối thiểu có dạng phương trình tổng quát để cho biết mối quan hệ giữa các điểm dữ liệu.

Nếu dữ liệu cho thấy mối quan hệ rõ ràng giữa hai biến nhất định, đường biểu diễn phù hợp nhất với mối quan hệ tuyến tính này được gọi là đường hồi qui bình phương tối thiểu.

Đường hồi qui bình phương tối thiểu có khoảng cách sao cho giữa các điểm dữ liệu đến đường này bình phương nhỏ nhất.

Nguyên nhân cần phải bình phương khoảng cách giữa các điểm dữ liệu và đường hồi qui là để ngăn các điểm dữ liệu trái dấu triệt tiêu cho nhau.

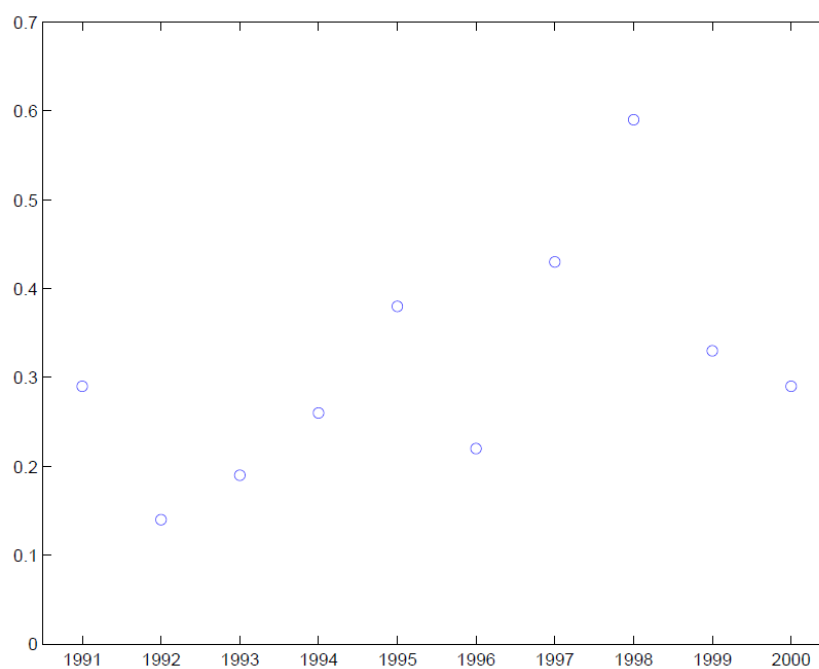
2. Cách tính hồi quy bình phương tối thiểu

Để có thể hiểu về cách thực hiện phương pháp bình phương cực tiểu một cách đơn giản nhất, chúng ta sẽ tiếp cận nó thông qua ví dụ đơn giản sau.

Ví dụ: Độ lệch nhiệt độ trung bình trên thế giới

Calendar year	Computational year t_i	Temperature deviation y_i
1991	1	0.29
1992	2	0.14
1993	3	0.19
1994	4	0.26
1995	5	0.28
1996	6	0.22
1997	7	0.43
1998	8	0.59
1999	9	0.33
2000	10	0.29

Bảng III-1 Độ lệch nhiệt độ trung bình hàng năm toàn cầu đo được ở độ C trong những năm 1991-2000



Hình III-1 Các phép đo độ lệch nhiệt độ trung bình hàng năm trên toàn cầu trong giai đoạn 1991-2000

2.1 Xấp xỉ bằng một hằng số

Chúng ta sẽ nghiên cứu xem tập dữ liệu này có thể được ước lượng như thế nào bởi các hàm đơn giản.

Đầu tiên, chúng ta hãy xem làm cách nào để tập dữ liệu này có thể được ước lượng gần đúng bằng một hàm không đổi:

$$p(t) \approx \alpha?$$

Dự đoán rõ ràng nhất sẽ là chọn α là trung bình số học

$$\alpha = \frac{1}{10} \sum_{i=1}^{10} y_i = 0.312$$

Chúng ta sẽ nghiên cứu phỏng đoán này chi tiết hơn.

Giả sử rằng chúng ta muốn giải pháp để giảm thiểu giá trị của hàm số

$$F(\alpha) = \sum_{i=1}^{10} (\alpha - y_i)^2$$

Hàm F đo lường một độ lệch từ α đến tập dữ liệu $(t_i, y_i)_{i=1}^{10}$

Chúng ta muốn tìm một giá trị α để làm cực tiểu giá trị của $F(\alpha)$, tức là chúng ta muốn tìm một α sao cho $F'(\alpha) = 0$

• Chúng ta có

$$F'(\alpha) = 2 \sum_{i=1}^{10} (\alpha - y_i)$$

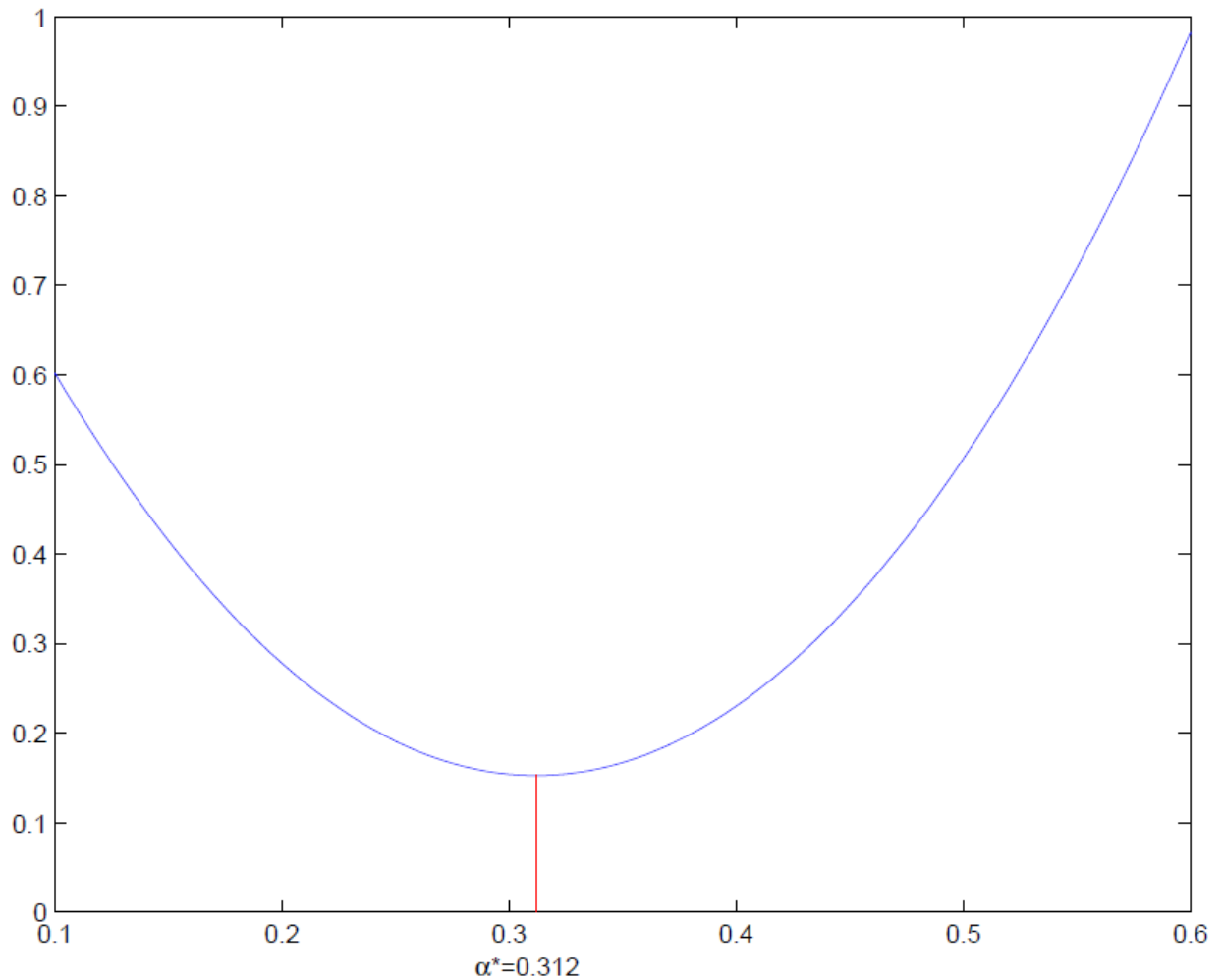
Điều này dẫn tới

$$2 \sum_{i=1}^{10} \alpha^* = 2 \sum_{i=1}^{10} y_i$$

Hay

$$\alpha^* = \frac{1}{10} \sum_{i=1}^{10} y_i$$

chính là trung bình số học



Hình III-2 Đồ thị $F = F(\alpha)$

Mặt khác, ta có:

$$F''(\alpha) = 2 \sum_{i=1}^{10} 1 = 20 > 0$$

Điều đó chứng minh rằng trung bình số học mà ta vừa tìm chính là điểm cực tiểu của F .

Chúng ta có thể nói rằng giá trị trung bình là giá trị tối ưu không đổi xấp xỉ cho nhiệt độ toàn cầu.

Cách để xác định một hằng số tối ưu, trong đó chúng ta giảm thiểu tổng bình phương của khoảng cách giữa ước tính và dữ liệu, được gọi là *phương pháp bình phương nhỏ nhất*.

Ngoài ra, có những cách khác để xác định một hằng số tối ưu.

Ta định nghĩa:

$$G(\alpha) = \sum_{i=1}^{10} (\alpha - y_i)^4$$

Hàm $G(\alpha)$ lúc này cũng đo lường độ lệch từ α đến dữ liệu

Ta có:

$$G'(\alpha) = 4 \sum_{i=1}^{10} (\alpha - y_i)^3$$

Và để tối thiểu hóa hàm G , chúng ta cần giải phương trình $G'(\alpha) = 0$ (và phải kiểm tra xem $G''(\alpha) > 0$ hay không)

Việc giải phương trình $G'(\alpha) = 0$ dẫn đến một phương trình phi tuyến có thể được giải quyết bằng phép lặp Newton.

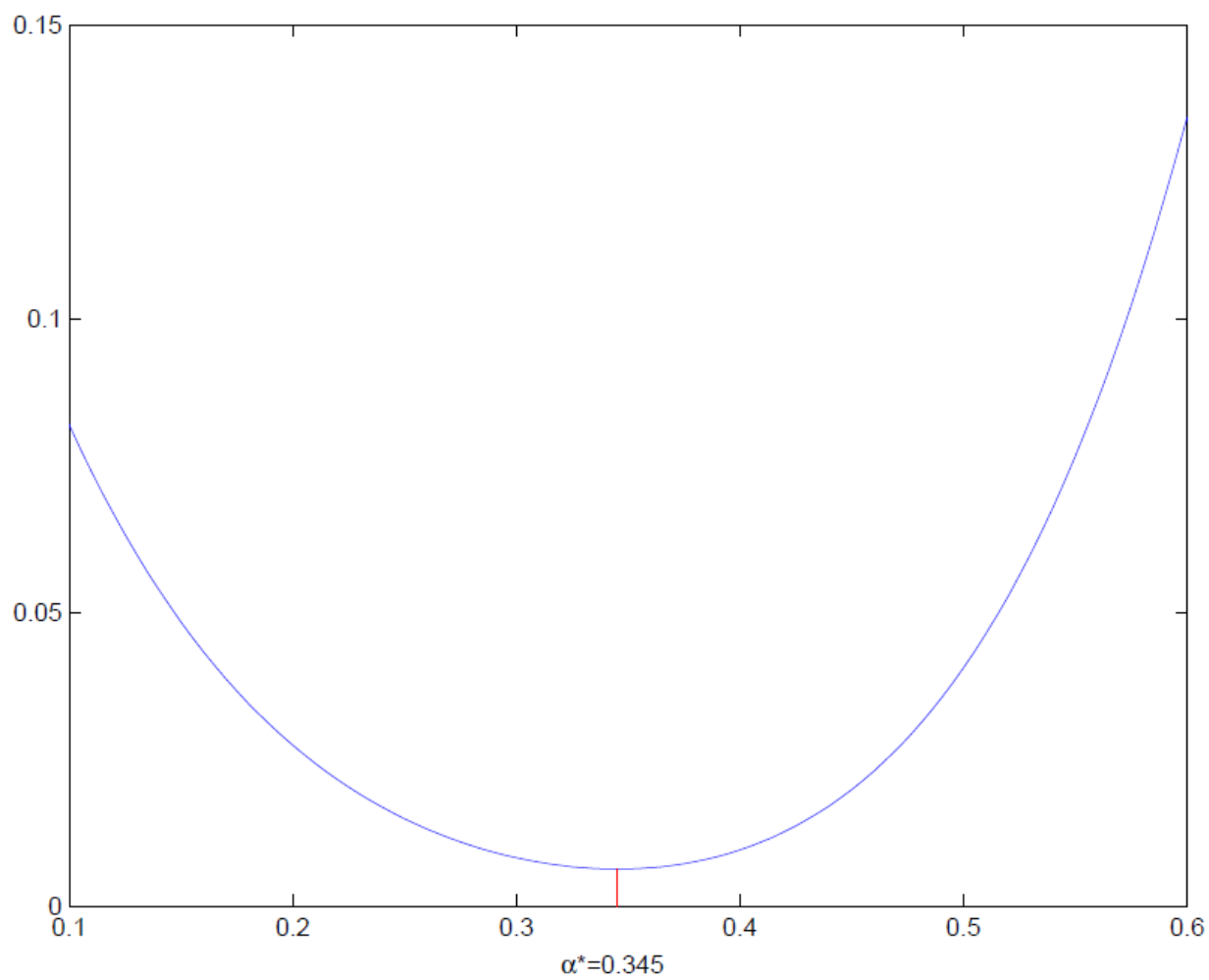
Chúng ta sử dụng phương pháp của Newton với

- Ước lượng ban đầu: $\alpha_0 = 0.312$
- Dung sai được chỉ định bởi: $\varepsilon = 10^{-8}$

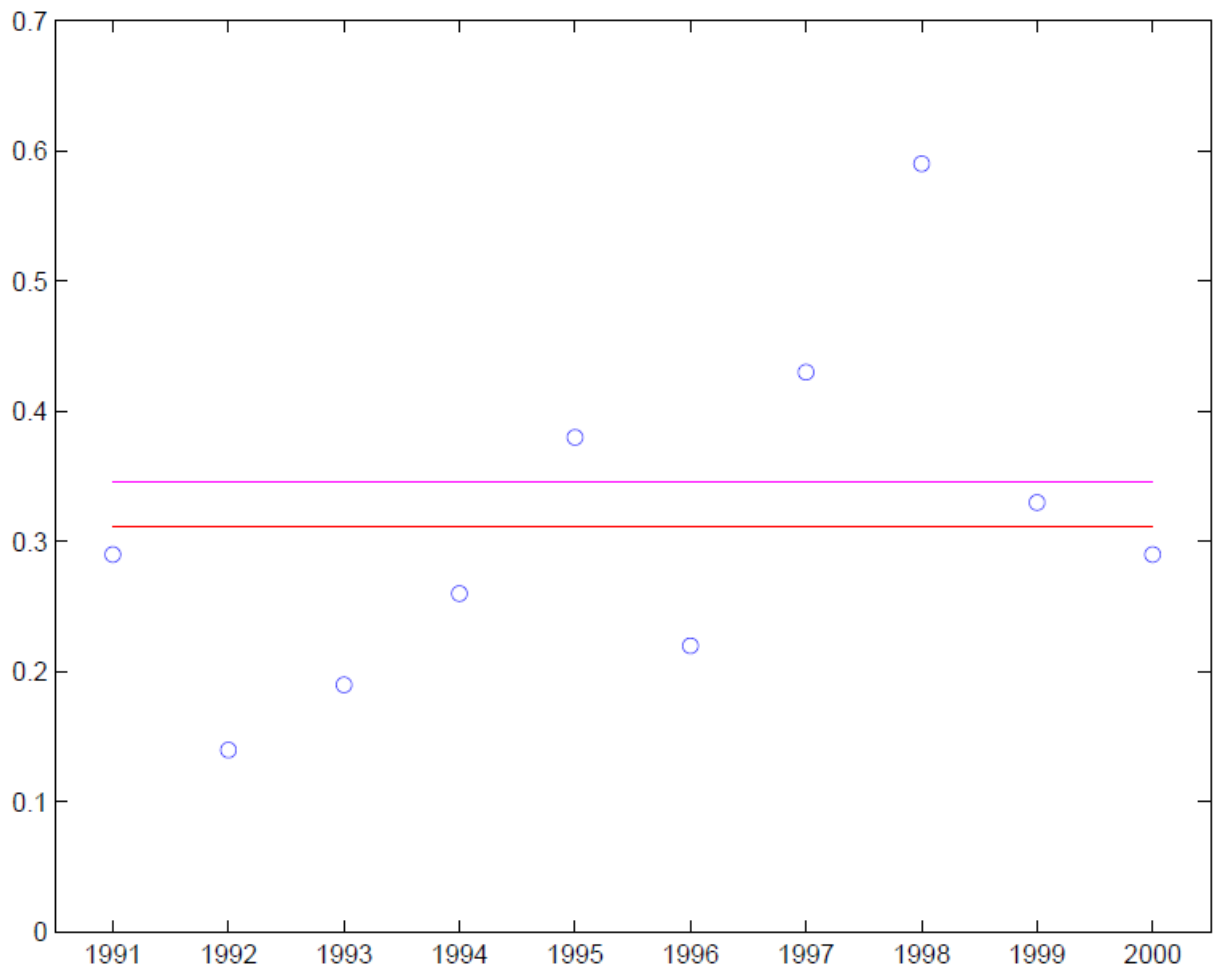
Điều này dẫn đến kết quả $\alpha^* \approx 0.345$ trong ba lần lặp.

α^* là một điểm cực tiểu của G vì

$$G''(\alpha^*) = 12 \sum_{i=1}^{10} (\alpha^* - y_i)^2 > 0$$



Hình III-3 Đồ thị $G = G(\alpha)$



Hình III-4 Hai giá trị xấp xỉ hằng số của các số đo độ lệch nhiệt độ trung bình hàng năm toàn cầu từ năm 1991 đến năm 2000

2.2 Xấp xỉ bằng một hàm tuyến tính

Bây giờ chúng ta sẽ nghiên cứu cách làm thế nào chúng ta có thể ước lượng độ lệch nhiệt độ trung bình trên thế giới với một hàm tuyến tính.

Chúng ta muốn xác định hai hằng số α và β sao cho

$$p(t) = \alpha + \beta t$$

phù hợp với dữ liệu tốt nhất có thể theo định nghĩa của bình phương cực tiểu.

Ta định nghĩa

$$F(\alpha, \beta) = \sum_{i=1}^{10} (\alpha + \beta t_i - y_i)^2$$

Để tối thiểu hóa giá trị của hàm F đối với hai tham số α và β , chúng ta có thể giải phương trình sau:

$$\frac{\partial F}{\partial \alpha} = \frac{\partial F}{\partial \beta} = 0$$

Ta có:

$$\frac{\partial F}{\partial \alpha} = 2 \sum_{i=1}^{10} (\alpha + \beta t_i - y_i)$$

Và do đó điều kiện $\frac{\partial F}{\partial \alpha} = 0$ dẫn đến phương trình sau:

$$10\alpha + \left(\sum_{i=1}^{10} t_i \right) \beta = \sum_{i=1}^{10} y_i$$

Trong đó:

$$\sum_{i=1}^{10} t_i = 1 + 2 + 3 + \dots + 10 = 55$$

Và:

$$\sum_{i=1}^{10} y_i = 0.29 + 0.14 + 0.19 + \dots + 0.29 = 3.12$$

Do đó chúng ta có:

$$10\alpha + 55\beta = 3.12 \quad (*)$$

Hơn nữa, chúng ta có:

$$\frac{\partial F}{\partial \beta} = 2 \sum_{i=1}^{10} (\alpha + \beta t_i - y_i) t_i$$

Và do đó điều kiện $\frac{\partial F}{\partial \beta} = 0$ cho ta phương trình sau:

$$\left(\sum_{i=1}^{10} t_i \right) \alpha + \left(\sum_{i=1}^{10} t_i^2 \right) \beta = \sum_{i=1}^{10} y_i t_i$$

Chúng ta có thể tính được:

$$\sum_{i=1}^{10} t_i^2 = 1 + 2^2 + 3^2 + \dots + 10^2 = 385$$

Và

$$\sum_{i=1}^{10} t_i y_i = 1 \cdot 0.29 + 2 \cdot 0.14 + 3 \cdot 0.19 + \dots + 10 \cdot 0.29 = 20$$

Vì vậy chúng ta đi đến phương trình:

$$55\alpha + 385\beta = 20 \quad (**)$$

Bây giờ chúng ta có một hệ phương trình tuyến tính 2×2 để xác định α và β :

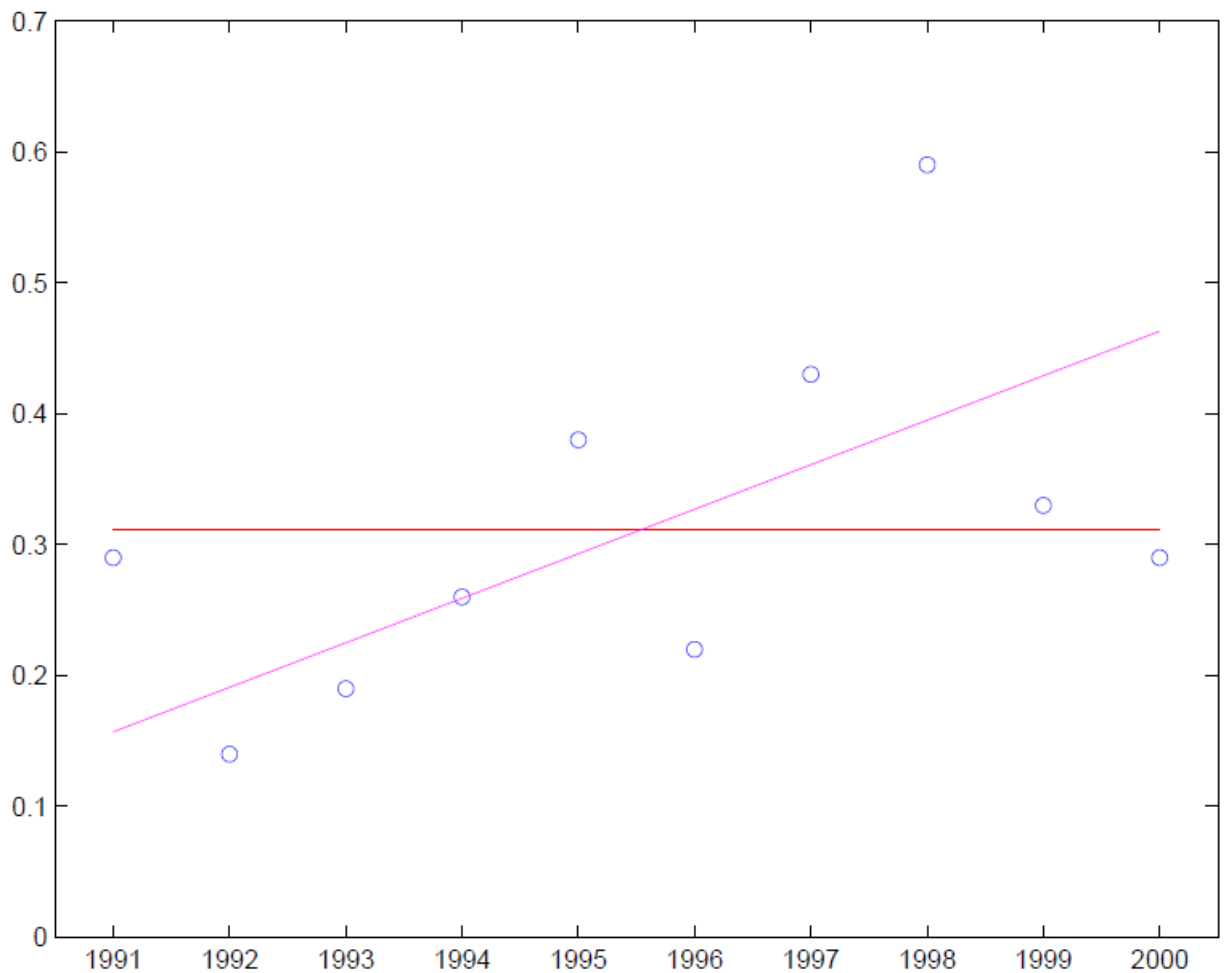
$$\begin{pmatrix} 10 & 55 \\ 55 & 385 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 3.12 \\ 20 \end{pmatrix}$$

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} 10 & 55 \\ 55 & 385 \end{pmatrix}^{-1} \begin{pmatrix} 3.12 \\ 20 \end{pmatrix} \\ = \frac{1}{825} \begin{pmatrix} 385 & -55 \\ -55 & 10 \end{pmatrix} \begin{pmatrix} 3.12 \\ 20 \end{pmatrix} \approx \begin{pmatrix} 0.123 \\ 0.034 \end{pmatrix}$$

Chúng ta kết luận rằng mô hình tuyến tính

$$p(t) = 0.123 + 0.034t$$

xấp xỉ dữ liệu tốt nhất theo định nghĩa của bình phương cực tiểu.



Hình III-5 Các xấp xỉ bình phương nhỏ nhất tuyến tính và xấp xỉ hằng của các số đo độ lệch nhiệt độ trung bình hàng năm toàn cầu từ năm 1991 đến năm 2000

2.3 Xấp xỉ bởi một hàm bậc hai

Bây giờ chúng ta muốn xác định các hằng số α , β và γ trong đa thức bậc hai sau:

$$p(t) = \alpha + \beta t + \gamma t^2$$

phù hợp với dữ liệu một cách tối ưu nhất theo nghĩa là bình phương nhỏ nhất

Để tìm điểm cực tiểu của hàm

$$F(\alpha, \beta, \gamma) = \sum_{i=1}^{10} (\alpha + \beta t_i + \gamma t_i^2 - y_i)^2$$

Chúng ta cần phải giải phương trình:

$$\frac{\partial F}{\partial \alpha} = \frac{\partial F}{\partial \beta} = \frac{\partial F}{\partial \gamma} = 0$$

Ta có:

$$\frac{\partial F}{\partial \alpha} = 2 \sum_{i=1}^{10} (\alpha + \beta t_i + \gamma t_i^2 - y_i) = 0$$

$$\frac{\partial F}{\partial \beta} = 2 \sum_{i=1}^{10} (\alpha + \beta t_i + \gamma t_i^2 - y_i) t_i = 0$$

$$\frac{\partial F}{\partial \gamma} = 2 \sum_{i=1}^{10} (\alpha + \beta t_i + \gamma t_i^2 - y_i) t_i^2 = 0$$

Suy ra:

$$10\alpha + \left(\sum_{i=1}^{10} t_i \right) \beta + \left(\sum_{i=1}^{10} t_i^2 \right) \gamma = \sum_{i=1}^{10} y_i$$

$$\left(\sum_{i=1}^{10} t_i\right) \alpha + \left(\sum_{i=1}^{10} t_i^2\right) \beta + \left(\sum_{i=1}^{10} t_i^3\right) \gamma = \sum_{i=1}^{10} y_i t_i$$

$$\left(\sum_{i=1}^{10} t_i^2\right) \alpha + \left(\sum_{i=1}^{10} t_i^3\right) \beta + \left(\sum_{i=1}^{10} t_i^4\right) \gamma = \sum_{i=1}^{10} y_i t_i^2$$

Trong đó:

$$\begin{aligned} \sum_{i=1}^{10} t_i &= 55, & \sum_{i=1}^{10} t_i^2 &= 385, & \sum_{i=1}^{10} t_i^3 &= 3025, \\ \sum_{i=1}^{10} t_i^4 &= 25330, & \sum_{i=1}^{10} y_i &= 3.12, & \sum_{i=1}^{10} t_i y_i &= 20, \\ \sum_{i=1}^{10} t_i^2 y_i &= 138.7, \end{aligned}$$

Thay các kết quả trên vào, chúng ta có hệ tuyến tính sau:

$$\begin{pmatrix} 10 & 55 & 385 \\ 55 & 385 & 3025 \\ 385 & 3025 & 25330 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} = \begin{pmatrix} 3.12 \\ 20 \\ 138.7 \end{pmatrix}$$

Giải hệ tuyến tính trên bằng máy tính, chúng ta nhận được:

$$\begin{aligned} \alpha &\approx -0.4078, \\ \beta &\approx 0.2997, \\ \gamma &\approx -0.0241. \end{aligned}$$

Bây giờ chúng ta đã tìm được ba hàm xấp xỉ với dữ liệu đã cho:

- Hằng số

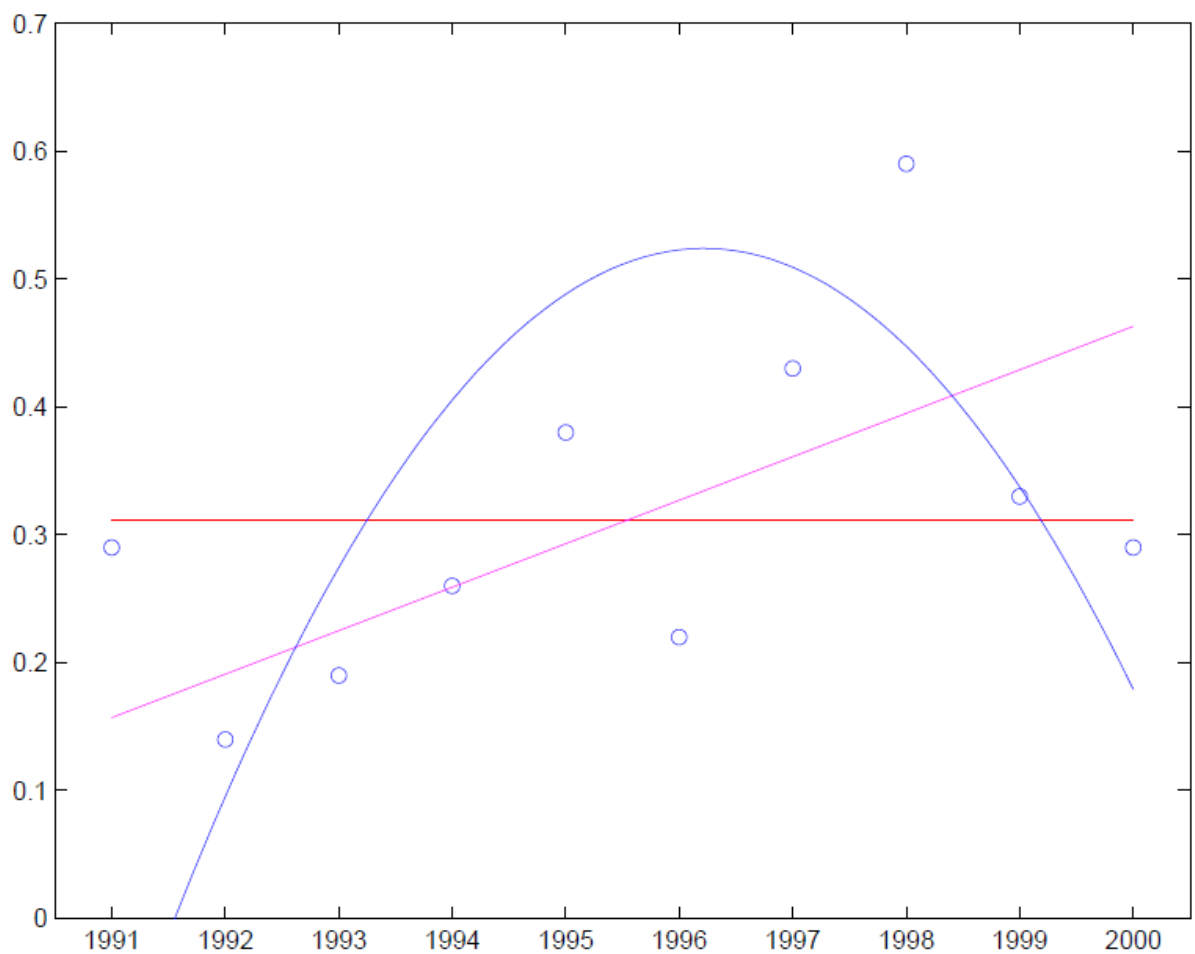
$$p_0(t) = 0.312$$

- Tuyến tính

$$p_1(t) = 0.123 + 0.034t$$

- Hàm bậc hai

$$p_2(t) = -0.4078 + 0.2997t - 0.0241t^2$$



Hình III-6 Các xấp xỉ hằng, tuyến tính và bậc hai của các số đo độ lệch nhiệt độ trung bình hàng năm toàn cầu từ năm 1991 đến năm 2000

2.4 Tổng kết

Xấp xỉ gần đúng một tập dữ liệu

$$(t_i, y_i) \text{ với } i = 1, 2, \dots, n$$

bằng một hàm không đổi

$$p_0(t) = \alpha$$

sử dụng phương pháp bình phương nhỏ nhất sẽ cho kết quả là

$$\alpha = \frac{1}{n} \sum_{i=1}^n y_i$$

chính là trung bình số học.

Xấp xỉ tập dữ liệu với một hàm tuyến tính

$$p_1(t) = \alpha + \beta t$$

có thể được thực hiện bằng cách tìm cực tiểu

$$\min_{\alpha, \beta} F(\alpha, \beta) = \min_{\alpha, \beta} \sum_{i=1}^n (p_1(t_i) - y_i)^2$$

dẫn đến hệ tuyến tính 2×2 sau

$$\begin{pmatrix} n & \sum_{i=1}^n t_i \\ \sum_{i=1}^n t_i & \sum_{i=1}^n t_i^2 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n y_i \\ \sum_{i=1}^n t_i y_i \end{pmatrix}$$

Xấp xỉ tập dữ liệu với một hàm bậc hai

$$p_2(t) = \alpha + \beta t + \gamma t^2$$

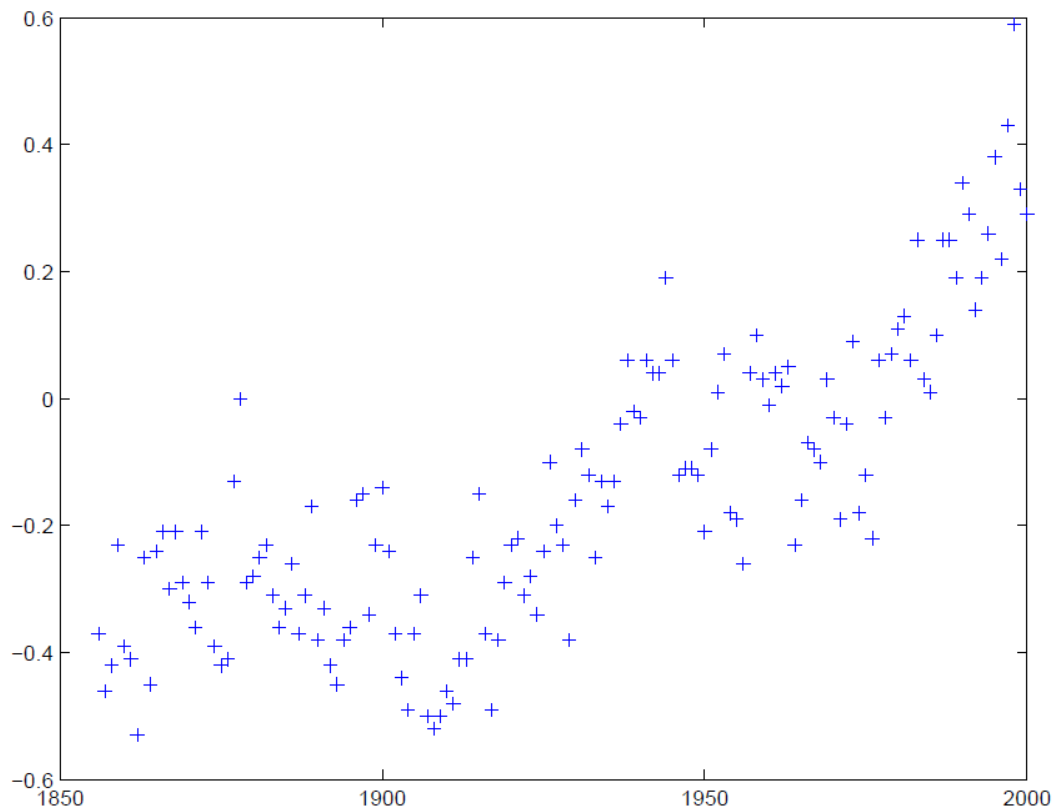
có thể được thực hiện bằng cách tìm cực tiểu

$$\min_{\alpha, \beta, \gamma} F(\alpha, \beta, \gamma) = \min_{\alpha, \beta, \gamma} \sum_{i=1}^n (p_2(t_i) - y_i)^2$$

dẫn đến hệ tuyến tính 3×3 sau:

$$\begin{pmatrix} n & \sum_{i=1}^n t_i & \sum_{i=1}^n t_i^2 \\ \sum_{i=1}^n t_i & \sum_{i=1}^n t_i^2 & \sum_{i=1}^n t_i^3 \\ \sum_{i=1}^n t_i^2 & \sum_{i=1}^n t_i^3 & \sum_{i=1}^n t_i^4 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^n y_i \\ \sum_{i=1}^n y_i t_i \\ \sum_{i=1}^n y_i t_i^2 \end{pmatrix}$$

Ở trong các phần trước, chúng ta chỉ mới xét trên tập dữ liệu nhỏ trong khoảng từ năm 1991 đến năm 2000, bây giờ chúng ta áp dụng công thức trên vào tập dữ liệu lớn hơn (từ năm 1856 đến năm 2000).



*Hình III-7 Các số đo độ lệch nhiệt độ trung bình hàng năm trên toàn cầu
từ năm 1856 đến năm 2000*

Chúng ta sử dụng các phương pháp đã biết để lập mô hình cho độ lệch nhiệt độ toàn cầu từ năm 1856 đến 2000. Trong đó

$$\begin{aligned}
n &= 145, & \sum_{i=1}^{145} t_i &= 10585, \\
\sum_{i=1}^{145} t_i^2 &= 1026745, & \sum_{i=1}^{145} t_i^3 &= 1.12042 \cdot 10^8, \\
\sum_{i=1}^{145} t_i^4 &= 1.30415 \cdot 10^{10}, & \sum_{i=1}^{145} y_i &= -21.82, \\
\sum_{i=1}^{145} t_i y_i &= -502.43, & \sum_{i=1}^{145} t_i^2 y_i &= 19649.8,
\end{aligned}$$

trong đó chúng ta đã cho $t_i = i$, tức là, $t_1 = 1$ tương ứng với năm 1856, $t_2 = 2$ tương ứng với năm 1857 v.v.

Đầu tiên, chúng ta nhận được mô hình không đổi sau:

$$p_0(t) \approx -0.1505$$

Các hệ số α và β của mô hình tuyến tính thu được bằng cách giải hệ tuyến tính trong trường hợp xấp xỉ tập dữ liệu với một hàm tuyến tính, tức là

$$\begin{pmatrix} 145 & 10585 \\ 10585 & 1026745 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} -21.82 \\ -502.43 \end{pmatrix}$$

Kết quả là:

$$\alpha \approx -0.4638 \text{ và } \beta \approx 0.0043$$

vì vậy mô hình tuyến tính được cho bởi công thức sau:

$$p_I(t) = -0.4638 + 0.0043t$$

Tương tự, các hệ số α , β và γ của mô hình bậc hai thu được bằng cách giải hệ tuyến tính trong trường hợp xấp xỉ tập dữ liệu với hàm bậc hai, tức là

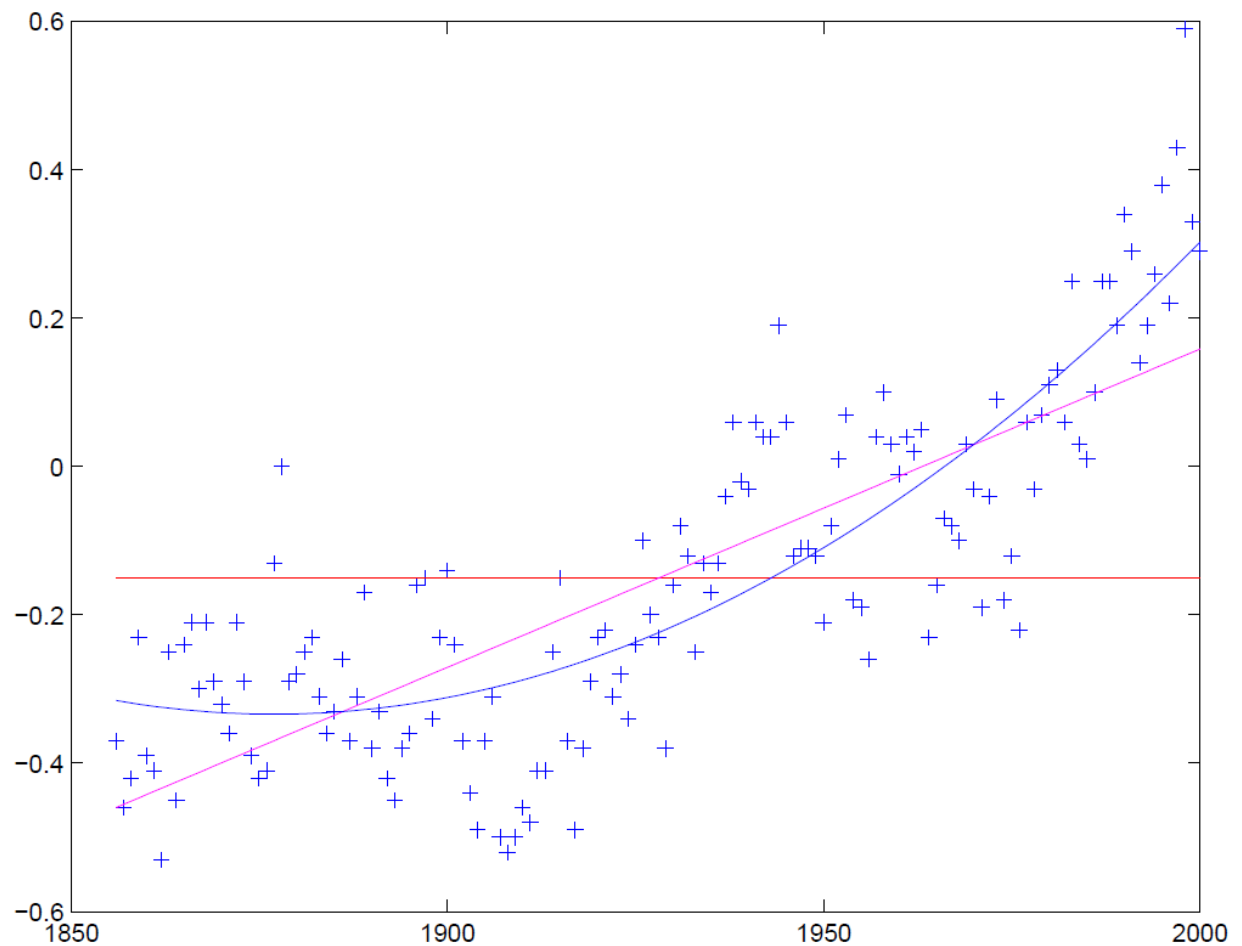
$$\begin{pmatrix} 145 & 10585 & 1026745 \cdot 10^6 \\ 10585 & 1026745 \cdot 10^6 & 1.12042 \cdot 10^8 \\ 1026745 \cdot 10^6 & 1.12042 \cdot 10^8 & 1.30415 \cdot 10^{10} \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} = \begin{pmatrix} -21.82 \\ -502.43 \\ 19649.8 \end{pmatrix}$$

Đáp án của hệ này là:

$$\alpha \approx -0.3136, \beta \approx -1.8404 \times 10^{-3} \text{ và } \gamma \approx 4.2005 \times 10^{-5},$$

vì vậy mô hình bậc hai được đưa ra bởi

$$p_2(t) \approx -0.3136 - 1.8404 \cdot 10^{-3}t + 4.2005 \cdot 10^{-5}t^2$$

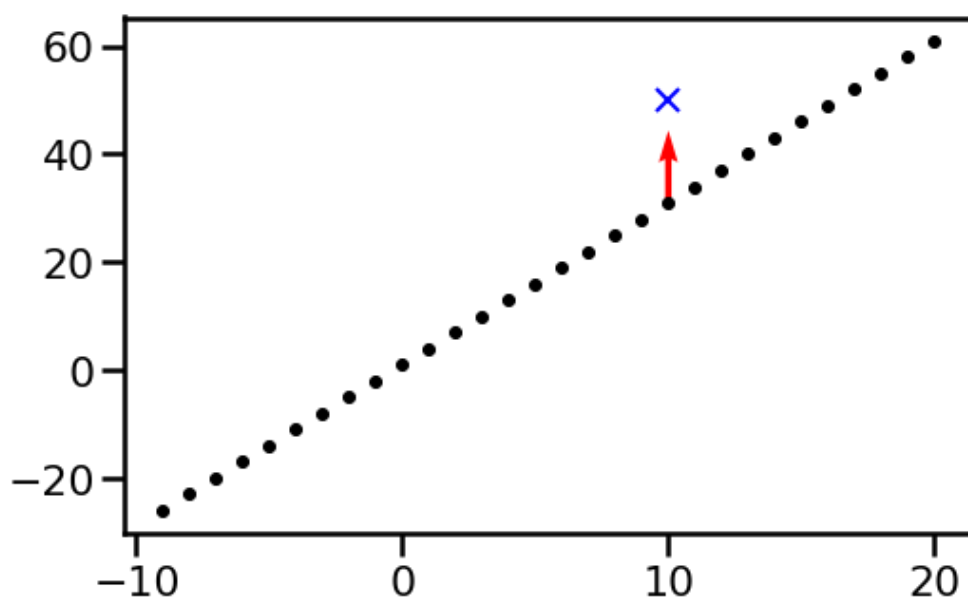


Hình III-8 Các đường xấp xỉ bình phương nhỏ nhất không đổi, tuyến tính và bậc hai của các phép đo độ lệch nhiệt độ trung bình hàng năm trên toàn cầu; từ năm 1856 đến năm 2000

IV. SỬ DỤNG MỘT VÍ DỤ ĐƠN GIẢN ĐỂ GIÚP ĐỂ HIỂU HƠN Ý TƯỞNG VỀ GIÁ TRỊ HỒI QUY BÌNH PHƯƠNG TỐI THIỂU CỦA NHÂN TIẾP TUYẾN THẦN KINH

1. Ví dụ 1

- Giả sử chúng ta có một hàm được xác định trên các số nguyên từ -10 đến 20.
- Chúng ta tham số hóa hàm này dưới dạng các giá trị hàm riêng lẻ: giá trị của hàm $f(i)$ tại mỗi số nguyên i phụ thuộc vào tham số $\theta_i = f(i)$.
- Ta khởi tạo các tham số của hàm này là $\theta_i = 3i+2$. Hàm này được biểu diễn bằng các chấm đen như hình bên.
- Xét một điểm dữ liệu $(x, y) = (10, 50)$ được biểu diễn bằng dấu gạch chéo màu xanh lam.
- Thực hiện cập nhật θ bằng gradient descent, sử dụng hàm loss sai số bình phương $(f(10; \theta) - 50)^2$ và tốc độ học (learning rate) $\eta = 0.1$



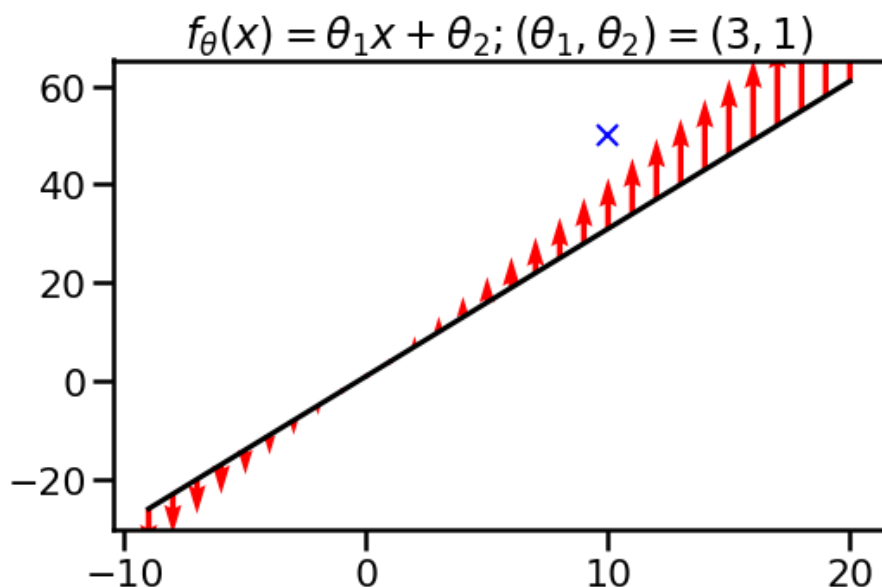
Hình IV-1 Minh họa sự thay đổi giá trị hàm khi thực hiện gradient descent trên hàm các giá trị riêng lẻ

Nhận xét:

Vì giá trị của hàm tại $x = 10$ chỉ phụ thuộc vào tham số θ_{10} , nên chỉ tham số này sẽ được cập nhật. Các tham số còn lại, và do đó các giá trị hàm tại điểm khác không thay đổi.

2. Ví dụ 2

- Xét hàm tuyến tính $f(x, \theta) = \theta_1 x + \theta_2$
- Khởi tạo các tham số là $\theta_1 = 3$ và $\theta_2 = 1 \rightarrow$ khi khởi tạo, ta có chính xác hàm trên số nguyên như đã có trong ví dụ 1.
- Cập nhật θ bằng cách thực hiện một bước gradient descent theo điểm dữ liệu $(x, y) = (10, 50)$ như trước.
- Các mũi tên màu đỏ hiển thị cách giá trị hàm di chuyển



Hình IV-2 Minh họa sự thay đổi giá trị hàm khi thực hiện gradient descent trên hàm tuyến tính

Nhận xét:

- Vì các giá trị hàm không còn được tham số độc lập như trong ví dụ 1, nên chúng ta không thể di chuyển chúng một cách độc lập.
- Mô hình liên kết chúng với nhau thông qua các tham số tổng thể θ_1 và θ_2

- Nếu chúng ta muốn di chuyển hàm đến gần đầu ra mong muốn $y = 50$ tại vị trí $x = 10$ thì các giá trị hàm ở nơi khác cũng phải thay đổi

Định nghĩa neural tangent kernel

- Cho một hàm $f_\theta(x)$ được tham số hóa bởi θ , nhân tiếp tuyến thần kinh (neural tangent kernel - NTK) của nó, ký hiệu là $k_\theta(x, x')$ xác định giá trị của hàm tại x thay đổi bao nhiêu khi ta thực hiện một bước gradient nhỏ trong thời gian ngắn theo θ đối với một sai số tại x' .
- Nói cách khác: $k(x, x')$ đo độ nhạy của giá trị hàm tại x đối với các lỗi dự đoán tại x' .
- Ký hiệu kích thước của các mũi tên màu đỏ tại mỗi vị trí x trong các ví dụ trước là $\tilde{k}_\theta(x, x')$

$$\eta \tilde{k}_\theta(x, x') = f\left(x, \theta + \eta \frac{df_\theta(x')}{d\theta}\right) - f(x, \theta)$$

$$k(x, x') = \lim_{\eta \rightarrow 0} \frac{f\left(x, \theta + \eta \frac{df_\theta(x')}{d\theta}\right) - f(x, \theta)}{\eta}$$

Sử dụng khai triển Taylor bậc 1 của $f_\theta(x)$:

$$k_\theta(x, x') = \left\langle \frac{df_\theta(x)}{d\theta}, \frac{df_\theta(x')}{d\theta} \right\rangle$$

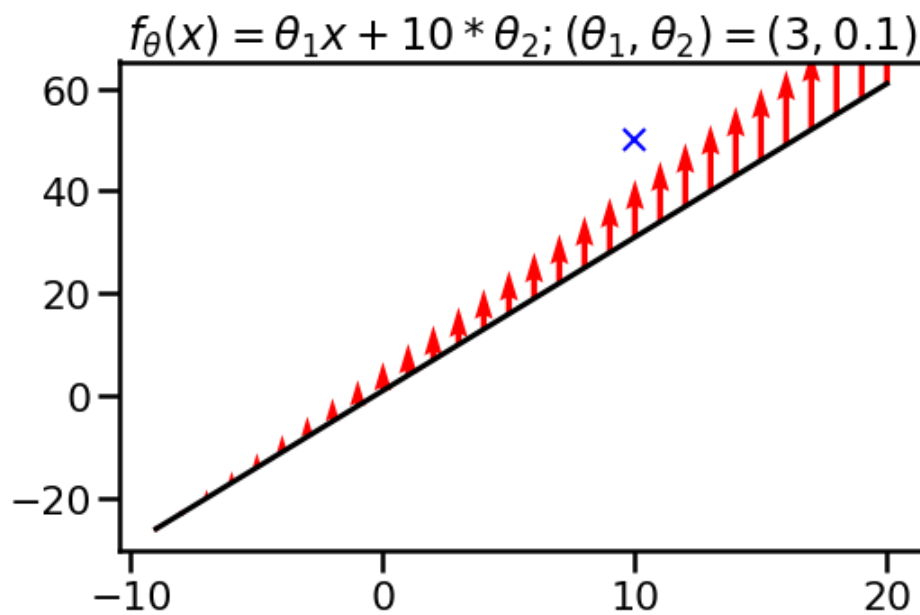
3. Ví dụ 3

Tham số hóa lại hàm tuyến tính đã sử dụng trong ví dụ 2 như sau:

$$f(x, \theta) = \theta_1 x + 10\theta_2$$

nhưng bây giờ với các tham số $\theta_1 = 3$ và $\theta_2 = 0.1$

Cập nhật θ bằng cách thực hiện một bước gradient descent theo điểm dữ liệu $(x, y) = (10, 50)$ như trước.



Hình IV-3 Minh họa sự thay đổi giá trị hàm khi thực hiện gradient descent trên hàm tuyến tính (sau khi tham số hóa lại hàm số)

Nhận xét:

- Cách giá trị hàm thay đổi khác với trong ví dụ 2.

→ NTK nhạy cảm với sự tham số hóa lại hàm số

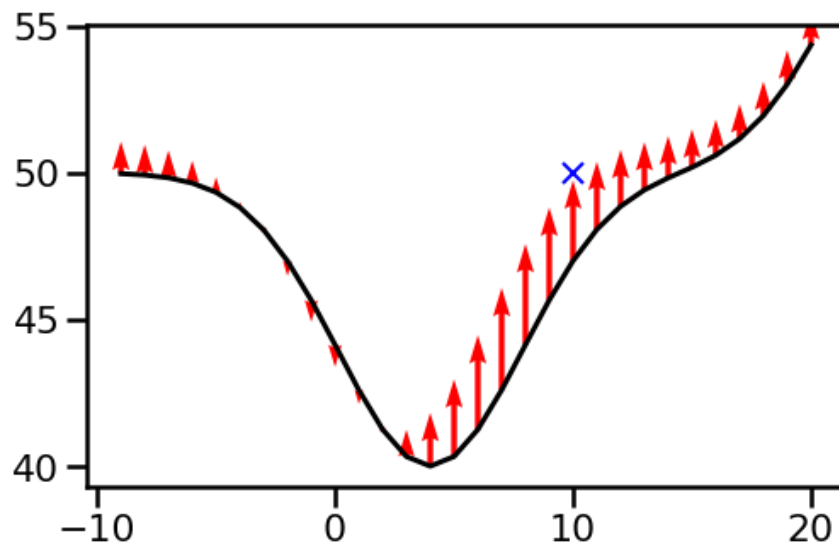
4. Ví dụ 4

Xét một mô hình phi tuyến có hai hàm cơ sở là mũ và bình phương:

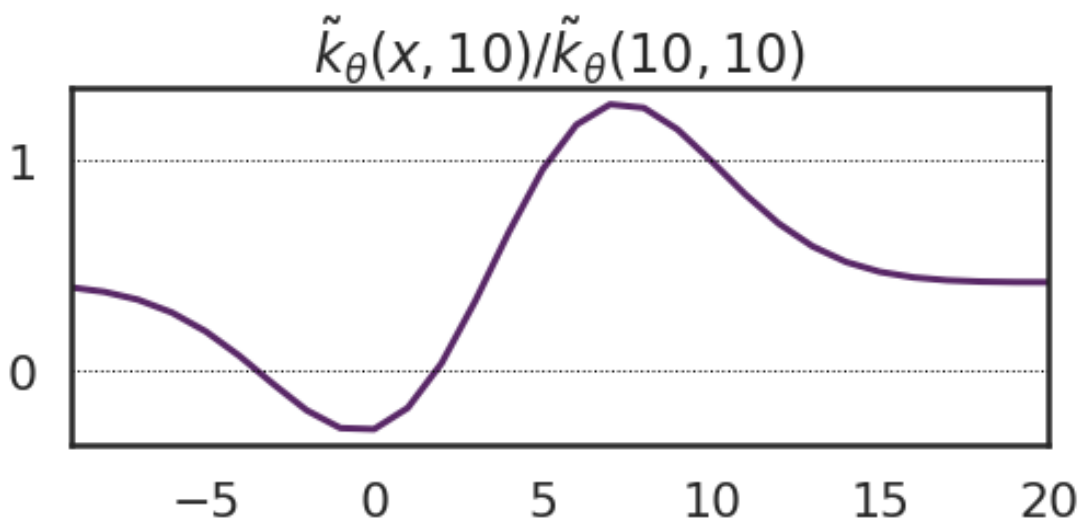
$$f_{\theta}(x) = \theta_1 \exp\left(-\frac{(x - \theta_2)^2}{30}\right) + \theta_3 \exp\left(-\frac{(x - \theta_4)^2}{30}\right) + \theta_5$$

$$(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) = (4.0, -10.0, 25.0, 10.0, 50.0)$$

Cập nhật θ bằng cách thực hiện một bước gradient descent theo điểm dữ liệu $(x, y) = (10, 50)$ như trước.



Hình IV-4 Minh họa sự thay đổi giá trị hàm khi thực hiện gradient descent trên hàm phi tuyến



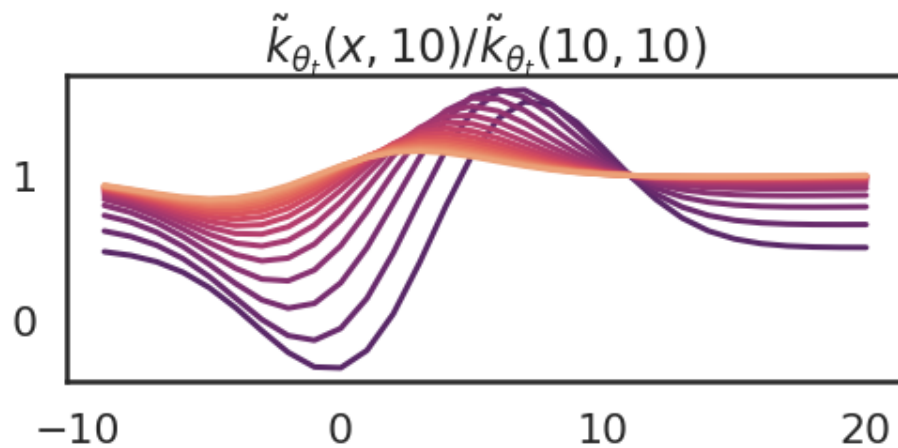
Hình IV-5 Đồ thị biểu diễn độ thay đổi của giá trị hàm đối với lỗi dự đoán tại $x = 10$ (sau khi đã chuẩn hóa bằng cách chia cho độ lệch tại $x = 10$)

Nhận xét:

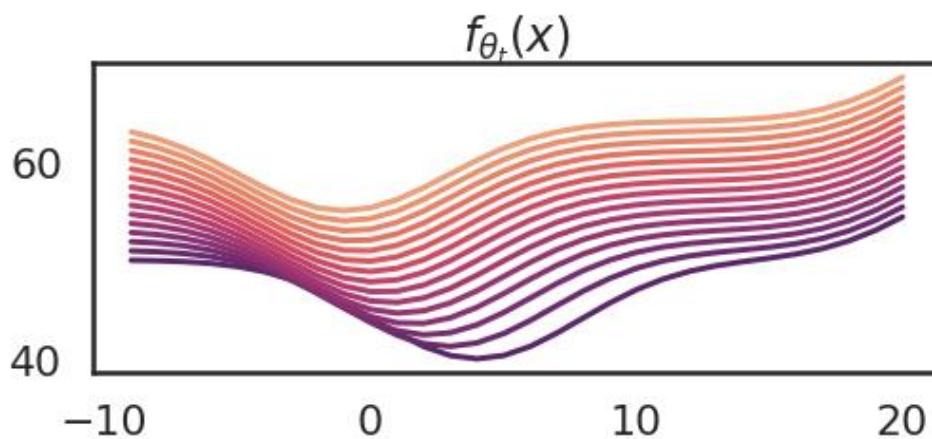
- Cực đại của hàm nhân này không phải ở $x = 10$ mà ở $x = 7$, nghĩa là giá trị hàm $f(7)$ thay đổi nhiều hơn so với giá trị $f(10)$, mặc dù ta cập nhật θ theo điểm dữ liệu tại $x = 10$.
- Tại một số điểm hàm có giá trị âm.
- Hàm nhân hội tụ về một hằng số dương ở các đuôi của nó.

5. Ví dụ 5

- Thực hiện cập nhật θ qua 15 bước gradient ascent nhằm cố gắng tăng $f(10)$ để gần với điểm dữ liệu $(x, y) = (10, 50)$.
- Đường cong màu tím là đường ta có lúc khởi tạo (ở trên) và đường cong màu cam hiển thị hàm nhân ở bước gradient cuối cùng.
- Các thay đổi tương ứng của hàm f_θ được hiển thị ở hình dưới.



Hình IV-6 Minh họa sự thay đổi của hàm nhân qua 15 bước gradient descent



Hình IV-7 Minh họa sự thay đổi của hàm f_θ qua 15 bước gradient descent

Nhận xét:

- Hàm nhân thay đổi trong quá trình huấn luyện.
 - Nhân trở nên phẳng hơn.
- Do hàm nhân phụ thuộc vào tham số của mô hình, trong quá trình huấn luyện, các tham số thay đổi, do đó hàm nhân thay đổi theo.

KẾT LUẬN VÀ KIẾN NGHỊ

Toàn bộ bài báo cáo nghiên cứu khoa học với đề tài *Nghiên cứu kỹ thuật hồi quy bình phương tối thiểu trong nhân tiếp tuyến thần kinh* đã làm rõ các nội dung về kỹ thuật hồi quy tuyến tính, hồi quy phi tuyến và kỹ thuật bình phương tối thiểu nói chung cũng như cách tính giá trị của nó. Đồng thời ở trong mỗi nội dung đều có các ví dụ minh họa giúp làm rõ và dễ hiểu hơn về ý tưởng và cách tính toán của các kỹ thuật hồi quy.

Ý nghĩa quan trọng nhất của đề tài nghiên cứu khoa học này là giúp chúng ta hiểu được kỹ thuật hồi quy bình phương tối thiểu, từ đó áp dụng vào trong mô hình mạng.

Tuy nhiên, vì đây là một đề tài còn khá mới và không có nhiều tài liệu, bài viết học thuật có liên quan để nghiên cứu, nên bài nghiên cứu khoa học này chưa thể đưa ra được ví dụ để làm sáng tỏ về mối liên hệ giữa kỹ thuật hồi quy bình phương tối thiểu và nhân tiếp tuyến thần kinh, đồng thời chưa thể nêu được ứng dụng cụ thể của vấn đề đã nghiên cứu vào trong đời sống.

Vì vậy, nội dung bài báo cáo này chỉ nên dùng vào mục đích tham khảo hoặc tìm hiểu về các kỹ thuật hồi quy như tuyến tính, phi tuyến và bình phương tối thiểu nói riêng. Bài nghiên cứu vẫn chưa có đủ nội dung cần thiết để nghiên cứu sâu về nhân tiếp tuyến thần kinh.

TÀI LIỆU THAM KHẢO

1. Ferenc Huszár, *Some Intuition on the Neural Tangent Kernel*, November 20, 2020.
2. Nhiều tác giả, *Bình phương tối thiểu*, wikipedia.org, 2021.
3. Saeed Aghabozorgi, *Non Linear Regression Analysis*, 2018.