HCMUS
Viet Nam National University
Ho Chi Minh City
University of Science

Khoa Toán - Tin học
Fac. of Math. & Computer Science

Chương 3

Mô hình Tuyến Tính cho Hồi Quy

ThS. Lưu Giang Nam

Bộ môn Ứng dụng Tin học
Khoa Toán - Tin học
Trường Đại học KHTN, ĐHQG TPHCM

10/2025

Thông tin thành viên

- Nguyễn Phước Thịnh
- Nguyễn Minh Khang
- Phạm Xuân Huyên
- Trần Hữu Bách Tùng

Phần 1

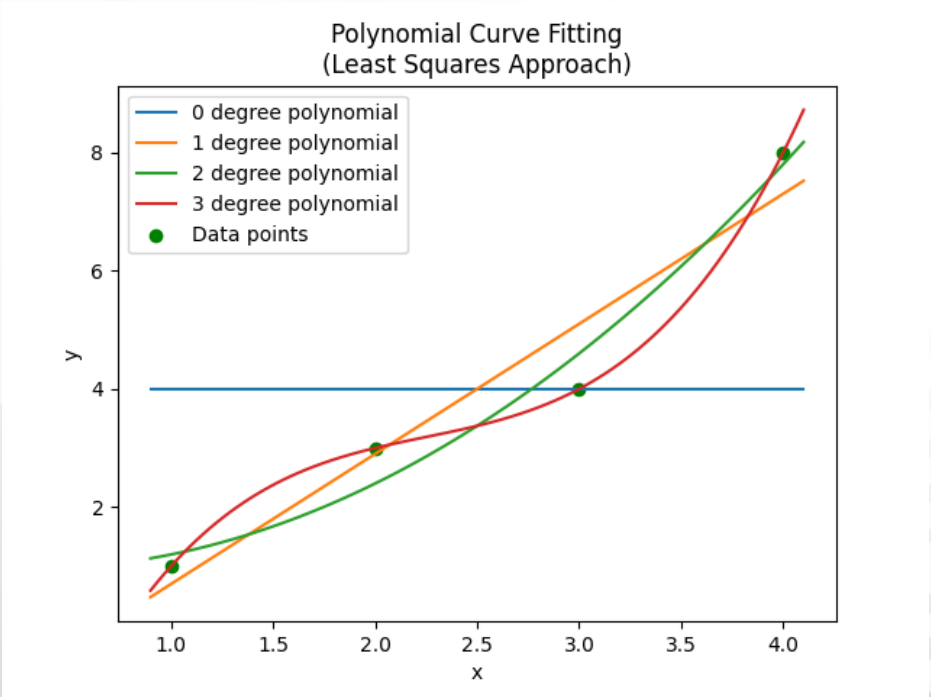
Tổng quát và ý nghĩa ra đời của Hồi Quy

Hồi quy trong Supervised Learning

Khái niệm cơ bản:

- **Hồi quy (Regression)** là bài toán dự đoán giá trị của *một biến liên tục*.
- Dựa trên một hoặc nhiều vector đầu vào $x \in \mathbb{R}^D$.

Ví dụ điển hình: Polynomial Curve Fitting



Mối liên hệ Polynomial với Linear Regression

- **Polynomial Regression** là một trường hợp đặc biệt của lớp mô hình **Linear Regression Models**.
- Mặc dù mô hình *phi tuyến theo biến đầu vào* x, nhưng vẫn *tuyến tính theo tham số* $\mathbf{w}$.
- Ví dụ:

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 = \mathbf{w}^\top [1, x, x^2, x^3]$$

- $\Rightarrow$ Polynomial Regression có thể được xem như một **Linear Model in the Parameters**, trong đó các hàm cơ sở được chọn là $\phi_j(x) = x^j$.

Mở rộng từ mô hình tuyến tính

Bayesian Linear Regression

- Mở rộng trực tiếp của Linear Regression, vẫn **tuyến tính theo tham số** nhưng có cách nhìn **xác suất**.
- Xem trọng số **w** là biến ngẫu nhiên $\Rightarrow$ mô hình cho **phân phối dự đoán** thay vì một giá trị duy nhất.
- **Công thức:**

$$p(t|x, t_{train}) = \mathcal{N}(t \mid \mathbf{m}_N^\top \phi(x),\ \beta^{-1} + \phi(x)^\top S_N \phi(x))$$

- **Ví dụ:** Dự đoán giá nhà theo diện tích kèm khoảng tin cậy 192 ± 15.

Phần 2

Mô hình hàm cơ sở tuyến tính

Linear Regression cơ bản

- Mô hình tuyến tính trực tiếp theo biến đầu vào:

$$y(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + \cdots + w_D x_D \tag{3.1}$$

Trong đó:

- x_j: đặc trưng đầu vào.
- w_j: trọng số mô hình.
- w_0: hệ số sai lệch
- y : giá trị đầu ra.

Mở rộng mô hình với Basis function

Thay vì chỉ dùng các biến đầu vào x trực tiếp, ta sẽ xây dựng các hàm phi tuyến $\phi(x)$ của các biến này.

Mô hình thành:

$$y(\mathbf{x}, \mathbf{w}) = \sum_{j=0}^{M-1} w_j \, \phi_j(\mathbf{x}) = \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}) \qquad (3.3)$$

- w_j: tham số đầu vào (có thể điều chỉnh).
- $\phi_j(x)$: Basis function.
- w_0: Tham số sai lệch **Bias**.

$\phi_0(x) = 1$ trong Linear Regression

$\phi_0(x)$ là một "dummy" basis function cố định, không phụ thuộc vào input x.

Mục đích của $\phi_0(x) = 1$ là cho phép dịch đồ thị lên/xuống:

- Nếu không có Bias w_0 thì mô hình sẽ buộc đi qua gốc tọa độ, có thể không khớp với dữ liệu.
- Với $\phi_0(x) = 1$ mô hình có thể tăng hoặc giảm toàn bộ giá trị y theo w_0.

Basis function thông dụng

- **Polynomial** :

$$\phi_j(x) = x^j$$

- **Gaussian**:

$$\phi_j(x) = \exp\left(-\frac{(x-\mu_j)^2}{2s^2}\right) \tag{3.4}$$

- **Sigmoidal**:

$$\phi_j(x) = \sigma\left(\frac{x-\mu_j}{s}\right) \tag{3.5}$$

Với

$$\sigma(a) = \frac{1}{1+\exp(-a)} \tag{3.6}$$

- **splines, Fourier, wavelets, etc**

Mô hình cơ sở tuyến tính

Trong trường hợp đơn giản, ta giả định mối quan hệ tuyến tính giữa đầu vào $\mathbf{x}$ và đầu ra t, kết hợp với 1 thành phần ngẫu nhiên :

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon \tag{3.7}$$

Trong đó ϵ là một biến ngẫu nhiên Gaussian gọi là **(nhiễu Gaussian)** có **zero mean** và **precision=** β. Do vậy, ta có:

$$p(t \mid x, \mathbf{w}, \beta) = \mathcal{N}(t \mid y(x, \mathbf{w}), \beta^{-1}) \tag{3.8}$$

Mô hình dữ liệu

- Giả sử biến mục tiêu t được tạo ra bởi:

$$t = y(\mathbf{x}, \mathbf{w}) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \beta^{-1})$$

Trong đó:

- $y(x, w)$: hàm dự đoán (deterministic function).
- ϵ: nhiễu Gaussian (zero-mean) với độ chính xác β

Maximum Likelihood và Least Squares

Phân phối xác suất của t

$$p(t \mid \mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}(t \mid y(\mathbf{x}, \mathbf{w}), \beta^{-1})$$

$$\Rightarrow \mathbb{E}[t|x, w, \beta] = y(x, w)$$

$\text{var}(t|x,w,\beta) = \beta^{-1}$

Với squared loss, dự đoán tối ưu là conditional mean:

$$\mathbb{E}[\, t \mid \mathbf{x}\,] = \int t\, p(t \mid \mathbf{x})\, dt = y(\mathbf{x}, \mathbf{w}) \tag{3.9}$$

Giả sử các điểm dữ liệu độc lập (i.i.d), ta có:

$$p(\mathbf{t} \mid \mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} p(t_n|x_n, w, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n \mid \mathbf{w}^\top \boldsymbol{\phi}(\mathbf{x}_n), \beta^{-1}) \tag{3.10}$$

X: toàn bộ dữ liệu

t: Toàn bộ đầu ra

Maximum Likelihood $\leftrightarrow$ Least Squares

Xét phân phối xác suất của t:

$$p(t \mid \mathbf{x}, \mathbf{w}, \beta) = \mathcal{N}\left(t \mid y(\mathbf{x}, \mathbf{w}), \beta^{-1}\right) \tag{3.8}$$

Lấy log-likelihood:

$$\ln p(\mathbf{t} \mid \mathbf{x}, \mathbf{w}, \beta) = \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \beta E_D(\mathbf{w}) \tag{3.11}$$

Trong đó:

$$E_D(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \left(t_n - \mathbf{w}^T \phi(x_n)\right)^2 \tag{3.12}$$

$E_D(w)$: Hàm sai số bình phương trung bình (sum-of-squares error).

Maximum Likelihood $\leftrightarrow$ Least Squares

Maximum Likelihood với w sẽ tương đương với việc minimize sum-of-squares error $E_D(w)$

Gradient:

$$\nabla_{\mathbf{w}} \ln p(\mathbf{t} \mid \mathbf{w}, \beta) = \sum_{n=1}^{N} (t_n - \mathbf{w}^T \phi(x_n))\, \phi(x_n)^T) \qquad (3.13)$$

Giải bằng phương trình **normal equation**:

$$\mathbf{w}_{\mathrm{ML}} = (\mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{t} \qquad (3.15)$$

Trong đó

- Φ: ma trận thiết kế, $\Phi \in \mathbb{R}^{N \times M}$, $\Phi_{nj} = \varphi_j(x_n)$
- $(\Phi^\top \Phi)^{-1} \Phi^\top$: Nghịch đảo giả Moore–Penrose

Geometry of least squares

Không gian N chiều

- Xét 1 không gian N chiều, mỗi trục ứng với một giá trị target t_n.
- **Vector target**

$$\mathbf{t} = (t_1, \ldots, t_N)^T$$

- Mỗi Basis function $\phi_j(x_n)$, evaluated tại N điểm dữ liệu, cũng có thể biểu diễn như một vector φ_j trong cùng không gian

  φ_j: cột thứ j của Φ

  $\phi(x_n)$: hàng thứ n của Φ

Geometry of least squares

Subspace và dự đoán

- Nếu số Basis function M $<$ N, các vector φ_j span một subspace S có dim M.
- vector dự đoán;

$$\mathbf{y} = (y(x_1, w), \ldots, y(x_N, w))^T$$

- Vì y là linear combination của $\varphi_j \rightarrow y$ nằm trong subspace S.

Least Squares = projection

- Sum-of-squares error: $E_D(\mathbf{w}) = \frac{1}{.}2.\sum_{n=1}^{N}(t_n - y_n)^2$.
- Normal equation: $\Phi^T \Phi w = \Phi^T t$.
- Ý nghĩa:

  - Least squares solution = orthogonal projection của t lên subspace S.

  - Tức là chọn y trong S sao cho gần t nhất.

Trong normal equations:

$$\mathbf{w}_{\mathrm{ML}} = (\Phi^T\Phi)^{-1}\Phi^T\mathbf{t} \tag{3.15}$$

Việc xác định w đòi hỏi phải lấy nghịch đảo ma trận ϕ^T
Ma trận $\Phi^T\Phi$ cần **invert** để tìm w.

Khi nào gặp vấn đề ?

Các basis vectors φ collinear hoặc gần collinear

Kết quả: $\Phi^T\Phi$ gần Singular.

Hiểu trực quan

- Subspace của các φ_j "gần trùng nhau" -> Least Squares không biết phải phân bổ weight thế nào.

- Projection lên subspace trở nên nhạy cảm với small perturbations trong dữ liệu.

Geometry of least squares

- Khi ma trận $(\Phi^T\Phi)$ gần suy biến, việc nghịch đảo trực tiếp có thể gây sai số số học.
- Trong thực tế, ta có thể sử dụng các kỹ thuật **ổn định hơn** (như SVD) để giải hệ phương trình một cách đáng tin cậy hơn.
- Mục tiêu: đảm bảo nghiệm w ổn định và chính xác hơn khi dữ liệu có quan hệ gần tuyến tính.

Sequential Learning

Hạn chế của Batch Learning

- **Batch techniques** (vd: Maximum Likelihood / Normal Equations) xử lý toàn bộ training set cùng lúc.
- Với dữ liệu lớn, tính toán có thể rất tốn kém.
- **Giải pháp**: Sequential / Online learning
  - Xử lý một điểm dữ liệu tại một thời điểm.
  - Cập nhật tham số mô hình ngay sau mỗi điểm.
  - Thích hợp cho streaming data hoặc realtime applications.

SGD algorithm (Stochastic Gradient Descent)

Nếu hàm lỗi là tổng quan các điểm dữ liệu:

$$E = \sum_n E_n = \sum_n E_n(w)$$

Thì sau khi đưa mẫu dữ liệu thứ n, thuật toán SGD sẽ cập nhật vector tham số w theo:

$$w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E_n \tag{3.22}$$

Trong đó:

- r: số vòng lặp.
- η: số tốc độ học.

Với hàm **lỗi bình phương**, ta có:

$$w^{(\tau+1)} = w^{(\tau)} + \eta(t_n - {w^{(\tau)}}^T \phi_n)\phi_n \tag{3.23}$$

Sequential Learning

Hàm mất mát mẫu n:

$$E_n(\mathbf{w}) = \frac{1}{2}(t_n - \mathbf{w}^T\phi_n)^2$$

Gradien:

$$\nabla E_n = -(t_n - \mathbf{w}^T\phi_n)\,\phi_n$$

Lưu ý:

- Learning rate $\eta \rightarrow$ cần chọn cẩn thận: quá lớn $\rightarrow$ không hội tụ, quá nhỏ $\rightarrow$ chậm.
- Cập nhật sau mỗi dữ liệu $\rightarrow$ giảm chi phí tính toán, thích hợp realtime.

Tổng quát về Regularization

Mục đích

Kiểm soát overfitting: giúp mô hình phức tạp không "học thuộc" dữ liệu quá mức.

Hàm lỗi tổng thể cần tối thiểu hóa:

$$E_{\text{total}}(\mathbf{w}) = E_D(\mathbf{w}) + \lambda E_W(\mathbf{w}) \qquad (3.24)$$

λ là hệ số regularization, cân bằng giữa lỗi dữ liệu $E_D(w)$ và regularizer $E_W(w)$.

Regularizer đơn giản: Weight Decay

Định nghĩa

$$E_W(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} \tag{3.25}$$

Hàm lỗi tổng thể kết hợp sum-of-squares:

$$E_{\text{total}}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(t_n - \mathbf{w}^T\phi(x_n)\right)^2 + \frac{\lambda}{2}\mathbf{w}^T\mathbf{w} \tag{3.27}$$

Ý nghĩa:

- Khuyến khích các trọng số w_j dần về 0 nếu dữ liệu không hỗ trợ.
- Đây là shrinkage, giúp hạn chế độ lớn của các tham số.
- Hàm lỗi vẫn là hàm bậc hai, nên có thể tìm cực tiểu chính xác.

Công thức giải closed-form

Trọng số tôi ưu w được tính bằng

$$\mathbf{w} = (\lambda I + \Phi^T \Phi)^{-1} \Phi^T \mathbf{t} \tag{3.28}$$

Giải thích các thành phần

w vector trọng số tối ưu, xác định cách kết hợp basis functions để dự đoán tốt.

Φ (design matrix): ma trận $N \times M$ mỗi hàng là

$$\phi(x_n) = \left[\phi_1(x_n), \quad \phi_2(x_n), \quad \ldots, \quad \phi_M(x_n)\right]^T$$

t: vector các giá trị target.

$\Phi^T \Phi$: tổng hợp mối quan hệ giữa các basis functions.

λI: regularization, giúp giảm overfit và giá trị trọng số w.

Regularizer tổng quát (q-norm)

Hàm lỗi tổng quát:

$$E_{\text{total}}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\left(t_n - \mathbf{w}^T\phi(\mathbf{x}_n)\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|^q \quad (3.29)$$

Các trường hợp đặc biệt:

$q = 2 \rightarrow$ quadratic regularizer (weight decay).

$q = 1 \rightarrow$ lasso, tạo mô hình sparse, một số $w_j = 0$ nếu λ lớn.

Vai trò của Regularization

- Cho phép huấn luyện mô hình phức tạp trên tập dữ liệu nhỏ mà không bị overfitting.
- **Thách thức**: xác định giá trị tối ưu của λ, thay vì chỉ quan tâm số lượng basis functions.

Phần 3

The Bias-Variance decomposition

Bias–Variance Decomposition

Vấn đề mô hình tuyến tính và overfitting

- Trước đây, ta giả định số lượng và dạng của các basis functions là cố định.
-Với **Maximum Likelihood** hoặc **Least Squares**, mô hình phức tạp sẽ dễ bị overfit, nhất là khi dữ liệu ít.
- Giới hạn số basis functions có thể giúp giảm tình trạng overfit nhưng lại làm mất tính linh hoạt.
- **Regularization** (tham số λ) giúp kiểm soát overfitting, nhưng việc chọn λ không đơn giản - nếu tối ưu cùng w, kết quả sẽ cho $\lambda = 0$ (không điều chuẩn).

Hướng tiếp cận: Frequentist vs Bayesian

- Overfitting là nhược điểm của **Maximum Likelihood** (Frequentist).
- Trong **Bayesian**, vấn đề này được khắc phục nhờ trung bình trên toàn bộ phân phối tham số thay vì dùng một giá trị duy nhất.

Với **squared loss**, dự đoán tối ưu là:

$$h(x) = \mathbb{E}[t \mid x] = \int t\, p(t \mid x)\, dt \quad (3.36)$$

Sai số kỳ vọng (expected squared loss):

$$\mathbb{E}[L] = \int (y(x) - h(x))^2\, p(x)\, dx + \int (h(x) - t)^2\, p(x, t)\, dx\, dt \quad (3.37)$$

- **Phần tử thứ 1**: Phụ thuộc vào mô hình $y(x)$.
- **Phần tử thứ 2**: Là **Noise** không thể tránh.

Frequentist view: trung bình trên nhiều bộ dữ liệu

Giả sử ta mô hình hóa hàm thật $h(x)$ bằng một mô hình tham số $y(x, w)$.

Ta có 2 cách nhìn nhận như sau:

- **Bayesian perspective:** Bất định trong mô hình được thể hiện bằng phân phối hậu nghiệm $p(w|D)$,
- **Frequentist perspective:** Ta chọn ước lượng điểm duy nhất $\hat{w}(D)$ dựa trên tập dữ liệu D,

Frequentist view: trung bình trên nhiều bộ dữ liệu

Ta xem quá trình huấn luyện mô hình là ngẫu nhiên, vì dữ liệu huấn luyện D được lấy ngẫu nhiên từ phân phối $p(x, t)$.
Giả sử ta có nhiều tập dữ liệu độc lập :

$$D^{(1)}, D^{(2)}, \ldots, D^{(L)} \sim p(x, t)$$

Ứng với mỗi $D^{(l)}$, mô hình sau huấn luyện là: $y(x, D^{(l)})$.
Mô hình này khác nhau, nên giá trị dự đoán và sai số cũng khác nhau.
Vì từng tập dữ liệu cho ra mô hình khác nhau, ta không thể đánh giá chỉ bằng 1 lần huấn luyện. Thay vào đó, ta đánh giá **hiệu suất trung bình** qua toàn bộ các tập dữ liệu.

$$\mathbb{E}_D\left[\left(y(x; D) - h(x)\right)^2\right] \tag{3.38}$$

$\rightarrow$ Đây là **sai số trung bình kỳ vọng**.

Frequentist view: trung bình trên nhiều bộ dữ liệu

Mục tiêu: Giải thích nguồn gốc của sai số trung bình qua các tập dữ liệu.

$$\mathbb{E}_D\Big[\big(y(x;D)-h(x)\big)^2\Big] \tag{3.38}$$

Ý tưởng chính Thêm bớt giá trị trung bình dự đoán $\mathbb{E}_D[\,y(x;D)\,]$ để tách 2 loại sai lệch:

$$\mathbb{E}_D\big\{y(x;D)-h(x)\big\}^2 =$$

$$\underbrace{\big(\mathbb{E}_D[y(x;D)]-h(x)\big)^2}_{\text{bias}^2}+\underbrace{\mathbb{E}_D\big\{y(x;D)-\mathbb{E}_D[y(x;D)]\big\}^2}_{\text{variance}}$$

(3.40)

Bias^2 : Sai lệch hệ thống giữa trung bình dự đoán và hàm thật $h(x)$.

Variance: dao động của dự đoán khi thay đổi tập dữ liệu.

Mở rộng sang toàn bộ dữ liệu

Cho đến nay, chúng ta xét sai số bình phương dự kiến tại một giá trị đầu vào duy nhất x:

$$\mathbb{E}_D\left[(y(x;D)-h(x))^2\right]$$

$$= \underbrace{(\mathbb{E}_D[y(x;D)]-h(x))^2}_{\text{Bias}^2} + \underbrace{\mathbb{E}_D\left[(y(x;D)-\mathbb{E}_D[y(x;D)])^2\right]}_{\text{Variance}} \tag{3.40}$$

Nếu mở rộng về kỳ vọng của hàm mất mát bình phương toàn cục:

$$\mathbb{E}[L] = \iint (y(x;D)-t)^2\, p(x,t)\, dx\, dt$$

Mở rộng sang toàn bộ dữ liệu

Thay biểu thức khai triển theo Bias–Variance vào, ta được decomposition đầy đủ:

$$\mathbb{E}[L] = \text{Bias}^2 + \text{Variance} + \text{Noise} \tag{3.41}$$

với các thành phần tích phân:

$$(\text{bias})^2 = \int \{\mathbb{E}_D[y(x; D)] - h(x)\}^2 \, p(x) \, dx \tag{3.42}$$

$$\text{variance} = \int \mathbb{E}_D\{y(x; D) - \mathbb{E}_D[y(x; D)]\}^2 \, p(x) \, dx \tag{3.43}$$

$$\text{noise} = \int \{h(x) - t\}^2 \, p(x, t) \, dx \, dt \tag{3.44}$$

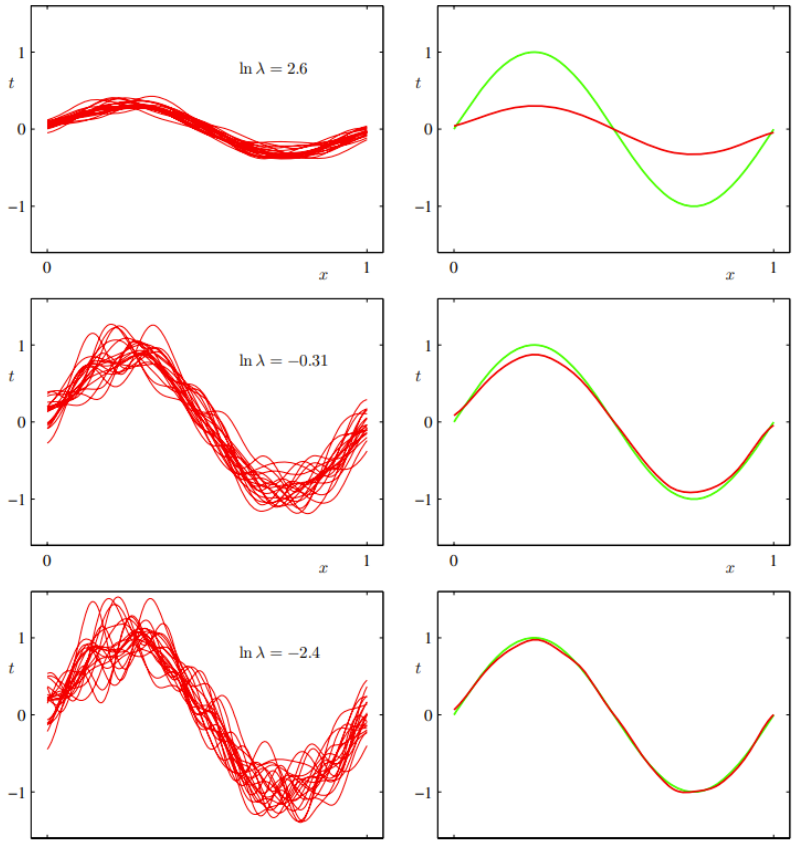
Giải thích

- **Bias**^{2}: Sai khác giữa dự đoán trung bình của thuật toán và hàm hồi quy thực $h(x)$, tích hợp qua phân phối đầu vào $p(x)$.
- **Variance**: Mức độ dự đoán dao động khi thay đổi tập dữ liệu, cũng tích hợp theo $p(x)$.
- **Noise**: Sai số không tránh được do bản chất ngẫu nhiên trong dữ liệu (t khác với $h(x)$), tích hợp theo phân phối chung $p(x, t)$.

Như vậy, tổng sai số dự kiến toàn cục được phân tách rõ ràng thành ba thành phần:

$$\mathbb{E}[L] = \text{Bias}^2 + \text{Variance} + \text{Noise}$$

Illustration Trade of bias-variance



Hình: Minh họa Bias-Variance theo độ phức tạp mô hình

Minh họa sự phụ thuộc của **bias** và **variance** vào độ phức tạp của mô hình, được điều khiển bởi tham số regularization λ, sử dụng tập dữ liệu hình sin trong Chương 1.

Cột bên trái cho thấy kết quả khi fit mô hình vào các tập dữ liệu với các giá trị khác nhau của $\ln \lambda$ (chỉ hiển thị 20 trong số 100 kết quả fit).
Cột bên phải cho thấy giá trị trung bình tương ứng của 100 kết quả fit (màu đỏ) cùng với hàm sinus từ đó các tập dữ liệu được sinh ra (màu xanh lá).

Phần 4

Bayesian Linear Regression

Kiểm soát độ phức tạp mô hình trong Hồi quy tuyến tính

Vấn đề

- Độ phức tạp hiệu quả của mô hình (số lượng hàm cơ sở) cần phù hợp với kích thước dữ liệu.
- Dùng maximum likelihood $\rightarrow$ mô hình quá phức tạp $\rightarrow$ overfitting.

Regularization:

- Thêm thuật ngữ điều chuẩn vào log-likelihood.
- Kiểm soát độ phức tạp thông qua hệ số λ.
- Số lượng và dạng hàm cơ sở vẫn quan trọng.

Chọn độ phức tạp mô hình

Dùng hold-out data có thể xác định, nhưng tốn kém và lãng phí dữ liệu.

Paramater distribution

Ý tưởng chính:

Trong hồi quy tuyến tính Bayes, ta không tìm một giá trị tối ưu duy nhất của tham số w mà xem w là một biến ngẫu nhiên với phân phối xác suất.

Phân phối tiên nghiệm (Prior Distribution)

Đặt phân phối Gauss cho tham số:

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w} \mid \mathbf{m}_0, \mathbf{S}_0) \tag{3.48}$$

m_0: Kỳ vọng ban đầu (mean)

S_0: Hiệp phương sai (covariance)

Phản ánh niềm tin ban đầu về w trước khi quan sát.

Tham số nhiễu β được xem là biết trước.

Phân phối hậu nghiệm (Posterior Distribution)

Dựa vào định lý Bayes:

$$p(\mathbf{w} \mid \mathbf{t}) = \frac{p(\mathbf{t} \mid \mathbf{w})\, p(\mathbf{w})}{p(\mathbf{t})}$$

Vì Gaussian đối hợp (conjugate) với Gaussian, nên hậu nghiệm cũng là Gaussian:

$$p(\mathbf{w} \mid \mathbf{t}) = \mathcal{N}(\mathbf{w} \mid \mathbf{m}_N, \mathbf{S}_N) \tag{3.49}$$

Với:

$$\mathbf{m}_N = \mathbf{S}_N \left(\mathbf{S}_0^{-1} \mathbf{m}_0 + \beta\, \Phi^T \mathbf{t} \right) \tag{3.50}$$

$$\mathbf{S}_N^{-1} = \alpha I + \beta\, \Phi^T \Phi \tag{3.51}$$

Paramater distribution

Mode và Mean trùng nhau:
Khi posterior là Gaussian, mode (MAP) và mean (posterior expectation) trùng nhau:

$$\mathbf{w}_{\text{MAP}} = \mathbf{m}_N$$

Do Gaussian có đồ thị đối xứng, nên *đỉnh* = *trung bình*.

Prior rộng ($\alpha \to 0$):
Khi prior rất rộng ($\alpha \to 0$), ảnh hưởng của prior không đáng kể:

$$\text{MAP} \approx \text{Maximum Likelihood (ML)}$$

Không có dữ liệu (N = 0):
Khi chưa quan sát dữ liệu, posterior chính là prior:

$$p(\mathbf{w} \mid D = \emptyset) = p(\mathbf{w})$$

Paramater distribution

Cập nhật tuần tự (Sequential Update):
Trong Bayesian learning:

$$\text{Posterior hiện tại} \equiv \text{Prior cho dữ liệu mới}$$

Phù hợp với *online / sequential learning*.

Predictive Distribution:
Posterior là phân phối của trọng số $\mathbf{w}$, không chỉ là giá trị cố định. Dùng để dự đoán dữ liệu mới:

$$p(t \mid \mathbf{x}, \mathbf{t}, \alpha, \beta) = \int p(t \mid \mathbf{x}, \mathbf{w}, \beta)\, p(\mathbf{w} \mid \mathbf{t}, \alpha, \beta)\, d\mathbf{w} \quad (3.57)$$

Paramater distribution

Trường hợp: Zero-mean isotropic prior

$$p(\mathbf{w} \mid \alpha) = \mathcal{N}(\mathbf{w} \mid \mathbf{0}, \alpha^{-1} I) \tag{3.52}$$

Khi đó:

$$\mathbf{m}_N = \beta\, \mathbf{S}_N\, \Phi^{\top} \mathbf{t} \tag{3.53}$$

Liên hệ với điều chuẩn (Regularization)

Log posterior:

$$\ln p(\mathbf{w} \mid \mathbf{t}) = -\frac{\beta}{2} \sum_{n=1}^{N} \left(t_n - \mathbf{w}^T \phi(x_n)\right)^2 - \frac{\alpha}{2} \mathbf{w}^T \mathbf{w} + \text{const} \tag{3.55}$$

Tối đa hóa posterior - tối thiểu hóa hàm lỗi có điều chuẩn:

$$E(\mathbf{w}) = \frac{\beta}{2} \sum_{n=1}^{N} \left(t_n - \mathbf{w}^T \phi(x_n)\right)^2 + \frac{\alpha}{2} \mathbf{w}^T \mathbf{w}, \quad \lambda = \frac{\alpha}{\beta}$$

Sequential Bayesian Learning

Sau mỗi quan sát, posterior được cập nhật $\rightarrow$ làm prior cho vòng sau.

Khi số điểm dữ liệu tăng:

- Posterior ngày càng sắc nét quanh giá trị thật của tham số.
- Khi $N \rightarrow \infty$: posterior $\rightarrow$ delta function tại tham số đúng.

Paramater Distribution

Xét prior tổng quát hơn:

$$p(\mathbf{w} \mid \alpha) = \left[\frac{q}{2}\left(\frac{\alpha}{2}\right)^{1/q}\frac{1}{\Gamma(1/q)}\right]^M \exp\left[-\frac{\alpha}{2}\sum_{j=1}^{M}|w_j|^q\right] \tag{3.56}$$

- Khi $q = 2$: trở về Gaussian, đối hợp với likelihood Gaussian $\rightarrow$ posterior Gaussian.
- Khi $q \neq 2$: Posterior không còn Gaussian, Mode $\neq$ Mean.

Predictive distribution

Mục tiêu:

Chúng ta thường không quan tâm trực tiếp đến giá trị tham số w mà muốn dự đoán giá trị t mới cho các input. Điều này yêu cầu tính predictive distribution:

$$p(t \mid x, \mathbf{t}, \alpha, \beta) = \int p(t \mid x, \mathbf{w}, \beta)\, p(\mathbf{w} \mid \mathbf{t}, \alpha, \beta)\, d\mathbf{w} \quad (3.57)$$

Dự đoán Gaussian

Posterior $p(w|t)$ là Gaussian, likelihood cũng Gaussian $\rightarrow$ predictive distribution cũng Gaussian:

$$p(t \mid x, \mathbf{t}, \alpha, \beta) = \mathcal{N}(t \mid \mathbf{m}_N^T \phi(x), \sigma_N^2(x)) \quad (3.58)$$

Phương sai predictive:

$$\sigma_N^2(x) = \beta^{-1} + \phi(x)^T \mathbf{S}_N \phi(x) \quad (3.59)$$

Equivalent kernel

Ý tưởng chính

Posterior mean m_N cho mô hình tuyến tính:

$$y(x, \mathbf{m}_N) = \mathbf{m}_N^T \phi(x) = \sum_{n=1}^{N} k(x, x_n)\, t_n \qquad (3.61)$$

$k(x, x_n)$: được gọi là equivalent kernel (smoother matrix).

Đặc điểm của equivalent kernel

Đối với localized basis functions (ví dụ Gaussian):

- $k(x, x_n)$ lớn nếu x gần x_n, nhỏ nếu xa trọng số cục bộ .
- Tương tự với polynomial hoặc sigmoidal basis $\rightarrow$ vẫn có sự phụ thuộc vào khoảng cách/không gian đầu vào.
- Các trọng số kernel tổng lại bằng 1: $\sum_{n=1}^{N} k(x, x_n) = 1$

Equivalent kernel

Liên hệ với covariance

$$\text{cov}[y(x), y(x')] = \phi(x)^T \mathbf{S}_N \phi(x') = \beta^{-1} k(x, x') \quad (3.63)$$

Tính chất

- Kernel có thể âm hoặc dương, không nhất thiết là tổ hợp lồi.
- Có thể viết dưới dạng inner product của vector phi mới $\psi(x)$:

$$k(x, z) = \psi(x)^T \psi(z), \quad \psi(x) = \beta^{1/2} \mathbf{S}_N^{1/2} \phi(x) \quad (3.65)$$

- Điều này liên hệ trực tiếp tới kernel trick trong machine learning.