Content

Bài toán

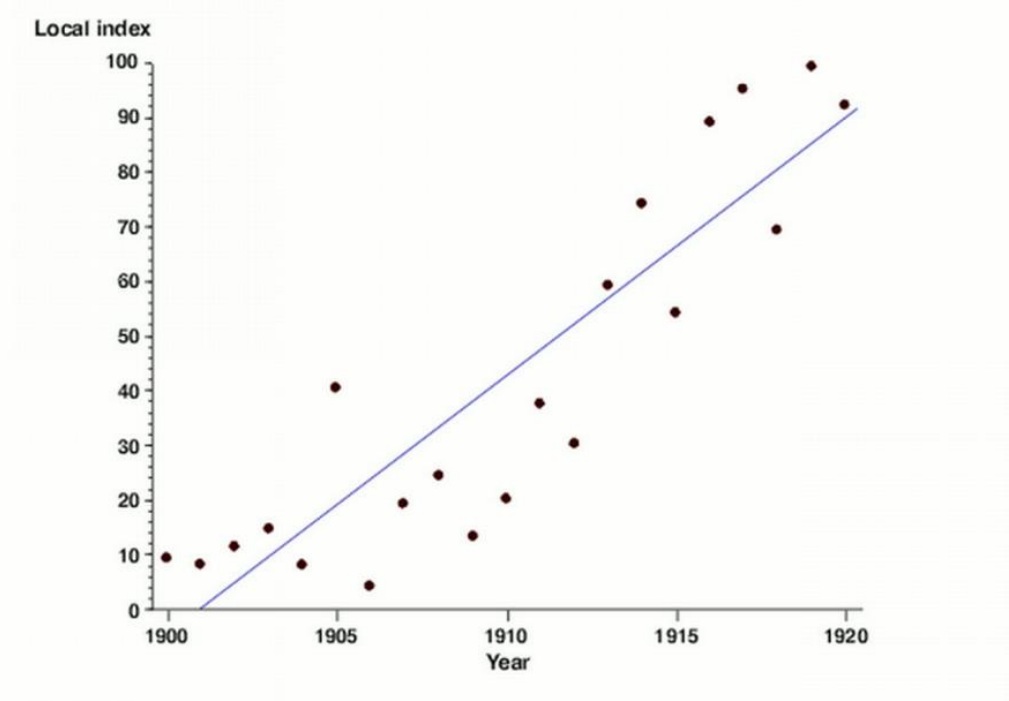
Background

About us

Topic

Linear và Logistic Regression

Logistic Regrssion
Linear Regression



Linear Regression và Logistic Regression là 2 bài toán trong Linear models

Content

Bài toán

Background

About us

Topic

20520079	20520070	20520055
Nguyễn Tư Thành Nhân	Lê Nhật Minh	Nguyễn Vinh Hưng

20520052	20520410
Phạm Nhật Hoàng	Trần Thái Bảo

Team MiFaFa

- Giải tích vector
- Lí thuyết xác suất
- Đại số tuyến tính

Cho điểm Lí, Hóa, Sinh của các bạn như sau

Tên	Lí	Hóa	Sinh	Khối
Nhân	9	8	6	A
Hoàng	6	8	8	A
Minh	7	3	6	B
Bảo	5	6	4	A
Hưng	2	3	3	B
Duy	2	10	3	B

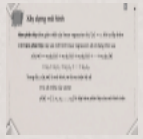
Các bạn thi khối A,B, cho biết các bạn thi khối nào.
Cho biết điểm Toán của các bạn

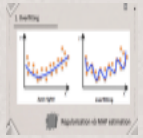
Linear Regression

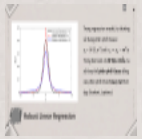
Logistic Regression

Linear Regression

Linear Regression


Xây dựng mô hình


Regularization và MAP estimation


Robust Linear Regression


Bayesian Linear Regression

Xây dựng mô hình

Hàm phân lớp đơn giản nhất của linear regression là $f(x) = x$. Khi ta lắp thêm một **hàm phân lớp** này vào mô hình linear regression sẽ có dạng như sau

$$y(\mathbf{x}, \mathbf{w}) = w_0\phi_0(\mathbf{x}) + w_1\phi_1(\mathbf{x}) + w_2\phi_2(\mathbf{x}) + \cdots + w_D\phi_D(\mathbf{x})$$

$$= w_0 + w_1x_1 + w_2x_2 + \cdots + w_Dx_D$$

Trong đó, $y(\mathbf{x}, \mathbf{w})$ là mô hình, $\mathbf{w}$ là ma trận hệ số

D là số chiều của vector

$\phi(\mathbf{x}) = [1, x_1, x_2, \ldots., x_D]$ là tập hàm phân lớp của mô hình trên

1. Các hàm phân lớp thường dùng được sử dụng

Chúng ta có thể thay thế các hàm phân lớp để phù hợp với bản chất dữ liệu

- Hồi quy đa thức: $\phi_i(x) = x^i$
- Hàm phân lớp Gauss: $\phi_i(x) = \exp\left(-\frac{(x-\mu)^2}{2\delta^2}\right)$
- Hàm phân lớp Sigmoid: $\phi_i(x) = \sigma\left(-\frac{x-\mu}{\delta}\right)$. Ta có thể thay bằng hàm tanh
- Hàm phân lớp hình sin: $\phi_i(x) = {}^1/_2 \sin(ax + b) + 1$
- Biến đổi mũ
- ...

1. Các hàm phân lớp thường dùng được sử dụng

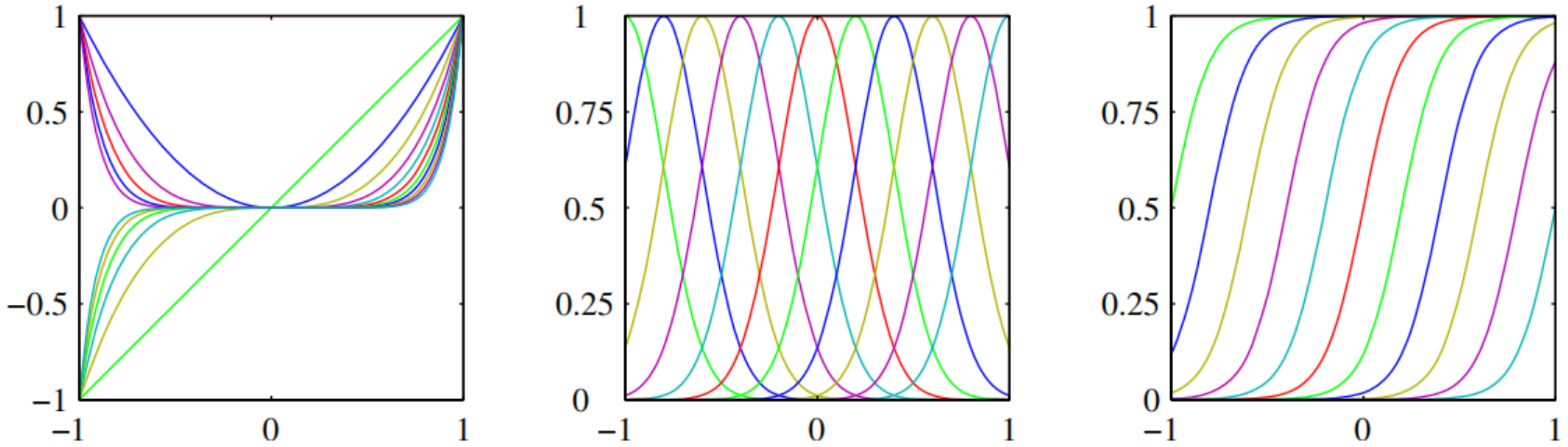


Figure 3.1 Examples of basis functions, showing polynomials on the left, Gaussians of the form (3.4) in the centre, and sigmoidal of the form (3.5) on the right.

Ngoài ra để biểu thị quan hệ giữa các chiều dữ liệu, người ta còn sử dụng đặc trưng chéo

2. Hướng tiếp cận MLE/LS trong xây dựng loss fucntion

Chúng ta giả sử giá trị mục tiêu (nhãn) $t = y(\mathbf{x}, \mathbf{w}, \phi) + \epsilon$ với ϵ là một phân phối chuẩn kỳ vọng 0 và độ chuẩn xác là số thực β.

Ta có hàm khả năng:

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

$$\mathbb{E}[t|\mathbf{x}] = \int tp(t|\mathbf{x})dt = y(\mathbf{x}, \mathbf{w}, \phi)$$

2. Hướng tiếp cận MLE/LS trong xây dựng loss fucntion

Xét tập dữ liệu $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N]$ nhận các giá trị mục tiêu $t_1, t_2, \dots, t_N$ được nhóm lại thành vector $\mathbf{t}$

Ta có hàm khả năng: $p(t|\mathbf{x}, \mathbf{w}, \phi, \beta) = \mathcal{N}\left(t\middle|\mathbf{w}^{\mathrm{T}}\mathbf{x}, \beta^{-1}\mathbf{I}\right)$

$$\mathrm{LL}(\mathbf{w}) = \log p(\,\mathbf{t}|\mathbf{x}, \mathbf{w}, \phi, \beta\mathbf{I}) = \sum_{n=1}^{N} \log\mathcal{N}\left(t_n\middle|\mathbf{w}^{\mathrm{T}}\mathbf{x}, \beta^{-1}\mathbf{I}\right)$$

$$= \frac{N}{2}\log\beta - \frac{N}{2}\log(2\pi) - \frac{1}{2}\beta\sum_{n=1}^{N}\left(t_n - \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x}_n)\right)^2$$

Với $\sum_{n=1}^{N}\left(t_n - \mathbf{w}^{\mathrm{T}}\phi(\mathbf{x})\right)^2$ là hàm tổng bình phương SSE (RSS)

2. Hướng tiếp cận MLE/LS trong xây dựng loss fucntion

Do $\mathrm{LL}(\mathbf{w}) = \mathrm{SSE} + \mathrm{const}$ nên ta có thể xem bản chất của SSE và MLE là như nhau

$$\nabla_{\mathbf{w}} \mathrm{LL}(\mathbf{w}) = \nabla_{\mathbf{w}} \log p(t|\mathbf{x}, \mathbf{w}, \phi, \beta)$$

$$= -\beta \sum_{n=1}^{N} \left(t_n - \mathbf{w}^{\mathrm{T}} \phi(\mathbf{x}_n) \right) \phi(\mathbf{x}_n)^{\mathrm{T}}$$

$$= -\beta \sum_{n=1}^{N} t_n \, \phi(\mathbf{x}_n)^{\mathrm{T}} + \beta \mathbf{w}^{\mathrm{T}} \sum_{n=1}^{N} \phi(\mathbf{x}_n) \phi(\mathbf{x}_n)^{\mathrm{T}}$$

$$\hat{\mathbf{w}} = \mathrm{argmax}_{\mathbf{w}} \mathrm{LL}(\mathbf{w}) \Leftrightarrow \nabla_{\mathbf{w}} \mathrm{LL}(\hat{\mathbf{w}}) = 0$$

Giải phương trình trên ta tìm được $\hat{\mathbf{w}} = (\mathbf{\Phi}^{\mathrm{T}} \mathbf{\Phi})^{-1} \mathbf{\Phi}^{\mathrm{T}} t$, trong đó $\mathbf{\Phi}$ là ma trận kích thước $N \times M$ với $\mathbf{\Phi}_{n,j} = \phi_j(\mathbf{x}_n)$.

Ta sẽ chứng minh $\hat{\mathbf{w}}$ là duy nhất bằng đạo hàm cấp 2. Đây còn gọi là thuật toán OLS

3. Những khó khăn khi thực hiện thuật toán OLS

- Đôi khi, việc tính toán nghiệm ước lượng hợp lý tối đại có chi phí rất lớn bởi các bộ test lớn. Vậy nên, việc thực hiện các thuật toán dạng chuỗi như SGD sẽ làm cho $\nabla_{\mathbf{w}}\mathrm{LL}$ tiến về 0 nhanh hơn.

$$\mathbf{w}_{r+1} = \mathbf{w}_{r+1} - \eta\nabla_{\mathbf{w}}\mathrm{LL}\left(\mathbf{x}_{\mathrm{rand}(1,N)}, \mathbf{w}_r\right)$$

$$= \mathbf{w}_r - \eta\left(t_{\mathrm{rand}(1,N)} - \mathbf{w}^T\phi\left(\mathbf{x}_{\mathrm{rand}(1,N)}\right)\right)\phi\left(\mathbf{x}_{\mathrm{rand}(1,N)}\right)$$

- Tuy nhiên, thuật toán OLS không được khuyến khích dùng nhiều bởi còn một rào cản: ma trận $\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}$ có thể không được tốt cho lắm(thưa, gần với không khả nghịch, ...). Trong trường hợp ma trận rất thưa(số lượng bộ dữ liệu nhỏ hơn nhiều so với số lượng chiều của đầu vào bài toán) ta sẽ sử dụng SVD.

3. Những khó khăn khi thực hiện thuật toán OLS

- Để tính nghịch đảo $\hat{\mathbf{w}}$ bằng SVD, ta làm như sau:

  Phân rã ma trận $\boldsymbol{\Phi} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathrm{T}}$

  Ta tính được

  $$\hat{\mathbf{w}} = \left(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi}\right)^{-1}\boldsymbol{\Phi}^{\mathrm{T}}t = \left(\mathbf{V}\boldsymbol{\Sigma}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}\mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^{\mathrm{T}}\right)^{-1}\mathbf{V}\boldsymbol{\Sigma}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}} = \mathbf{V}\left(\boldsymbol{\Sigma}^{\mathrm{T}}\boldsymbol{\Sigma}\right)^{-1}\boldsymbol{\Sigma}^{\mathrm{T}}\mathbf{U}^{\mathrm{T}}$$

  Bằng cách này, độ phức tạp là $\mathrm{O}(DN^2)$ với N là số lượng bộ dữ liệu, D là số chiều của đầu vào bài toán

  Ta còn có thể sử dụng ma trận $\boldsymbol{\Sigma}$ để giảm số chiều của bài toán(dữ liệu không cần thiết).

- Ngoài ra, QR decompostion cũng được dùng để giải quyết vấn đề này ở trường hợp khác.

4. Bài toán nhiều đầu ra

Đến đây, chúng ta chỉ giải quyết trường hợp có duy nhất một biến mục tiêu t. Trong nhiều trường hợp, chúng ta cần phải dự đoán một lượng $K > 1$ biến ngẫu nhiên, ta kí hiệu lại biến mục tiêu là vectơ $\mathbf{t}$ cho trường hợp tổng quát hơn. Điều này có thể được thực hiện bằng cách với mỗi biến mục tiêu, ta dùng một tập các hàm phân lớp khác nhau, tuy nhiên, một cách tiếp cận thông thường và hấp dẫn hơn là dùng một tập duy nhất các hàm phân lớp để xây mô hình sao cho:

$$y(\mathbf{x}, \mathbf{w}, \phi) = \mathbf{W}^{\mathrm{T}}\phi(\mathbf{x})$$ với $\mathbf{W}$ là một ma trận có kich thước $M \times K$

4. Bài toán nhiều đầu ra

Ta có hàm khả năng tương ứng sẽ là:

$$p(t|\mathbf{x}, \mathbf{w}, \phi, \beta) = \mathcal{N}\left(t\middle|\mathbf{W}^{\mathrm{T}}\phi(\mathbf{x}), \beta^{-1}\right)$$

Và hàm log-likelihood sẽ là

$$\mathrm{LL}(\mathbf{w}) = \sum_{n}^{N} \log \mathcal{N}\left(t_n\middle|\mathbf{W}^{\mathrm{T}}\phi(\mathbf{x}_n), \beta^{-1}\right) = \frac{NK}{2}\log\frac{\beta}{2\pi} - \frac{\beta}{2}\sum_{n=1}^{N}[t_n - \mathbf{W}^{\mathrm{T}}\phi(\mathbf{x}_n)]^2$$

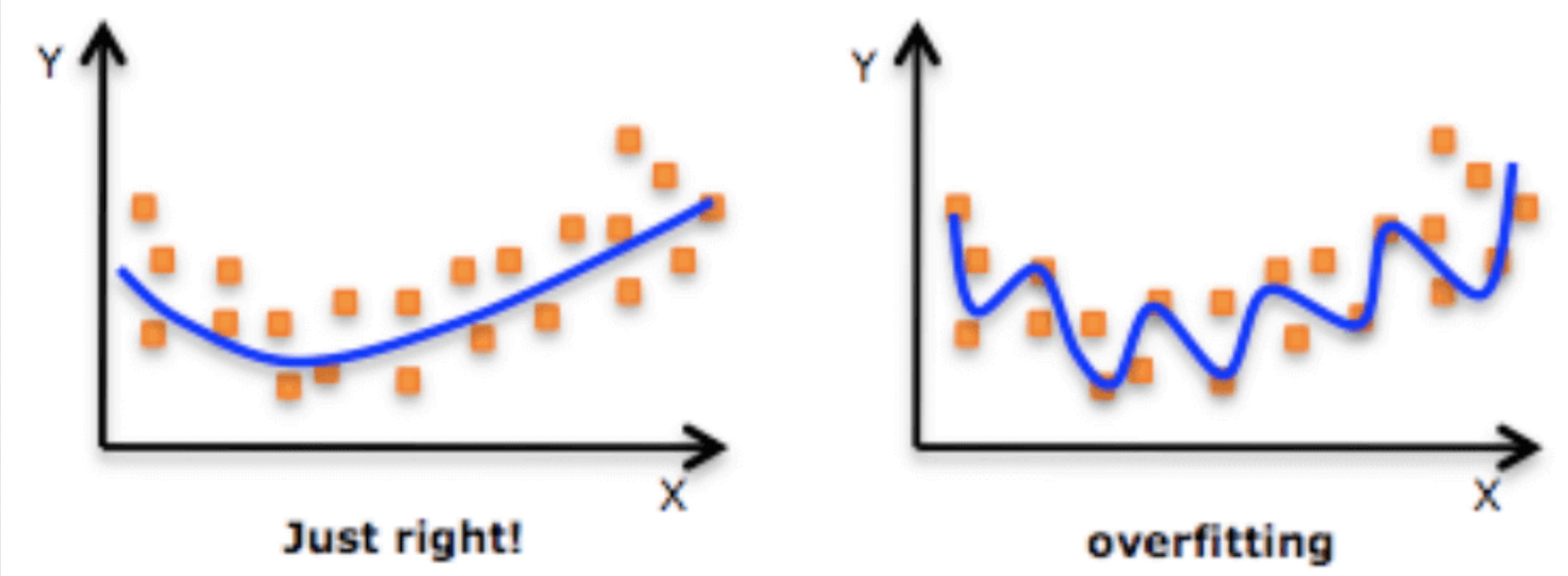
4. Bài toán nhiều đầu ra

- Ở đây, để ký hiệu đỡ rối, các bạn hãy tưởng tượng rằng trước đây mỗi phần tử x là một biến thuộc tập $\mathbb{R}$, thì sau đó là một biến thuộc tập $\mathbb{R}^K$, các hàm khác được xây dựng tương tự.

$$\widehat{\mathbf{W}} = \left(\mathbf{\Phi}^{\mathrm{T}}\mathbf{\Phi}\right)^{-1}\mathbf{\Phi}^{\mathrm{T}}\mathbf{t}$$

- Vậy bài toán nhiều đầu ra có thể được phân rã thành bài toán một đầu ra một cách dễ dàng, từ giờ trở đi, chúng ta chỉ xét bài toán một đầu ra nhằm giảm độ phức tạp của bài viết.

1. Overfitting



Regularization và MAP estimation

2. Ridge Regression

Công thức Ridge Regression

$$p(\text{Model}|\text{Data}) = \frac{p(\text{Data}|\text{Model})p(\text{Model})}{p(\text{Data})}$$

Hay là

$$p(y(\mathbf{x}, \mathbf{w}, \phi)|\mathcal{D}) = \frac{p(\mathcal{D}|y(\mathbf{x}, \mathbf{w}, \phi))p((\mathbf{x}, \mathbf{w}, \phi))}{p(\mathcal{D})}$$

$p(\mathcal{D}|y(\mathbf{x}, \mathbf{w}, \phi))$ là hàm likelihood, được xấp xĩ theo phân phối Gauss

$\mathcal{N}(\mathcal{D}|y(\mathbf{x}, \mathbf{w}, \phi), \beta^{-1})$

2. Ridge Regression

- Ở ước lượng hợp lí cực đại, chúng ta tìm cách tối ưu hóa $p(y(\mathbf{x}, \mathbf{w}, \phi)|\mathcal{D})$ bằng cách tối ưu hóa $p(\mathcal{D}|y(\mathbf{x}, \mathbf{w}, \phi))$ và cho rằng $\frac{p((\mathbf{x},\mathbf{w},\phi)}{p(\mathcal{D})}$ là một hằng số hay là phân phối đều. Về bản chất hàm regularization ℓ_2 là cho rằng $p(y(\mathbf{x}, \mathbf{w}, \phi)|\mathcal{D})$ xấp xĩ phân phối chuẩn $\mathcal{N}\left(\mathbf{w}^{\mathrm{T}}\mathbf{w}\middle|0, \lambda^{-1}\mathbf{I}\right)$
- Tương tự như hàm khả năng, sau khi tính $\mathrm{LL}(p(y(\mathbf{x}, \mathbf{w}, \phi)))$ tỉ lệ thuận với hàm bình phương

2. Ridge Regression

Gọi $E_{\mathrm{D}}(\mathbf{w})$ là hàm lỗi của train data (ở đây ta tính bằng (SSE)

$E_{\mathrm{T}}(\mathbf{w})$ là hàm lỗi của test/real data

$E_{\mathrm{W}}(\mathbf{w})$ là hàm regularization, với λ là hệ số chính quy

Ta có: $E(\mathbf{w}) = E_{\mathrm{D}}(\mathbf{w}) + \lambda E_{\mathrm{W}}(\mathbf{w})$

$$= \mathrm{SSE}(\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2 = \sum_{n=1}^{N} \left(t_n - \mathbf{w}^{\mathrm{T}} \phi(\mathbf{x}_n) \right)^2 + \lambda \|\mathbf{w}\|_2^2$$

Khi đó $\mathbf{g}(\mathbf{w}) = \nabla_{\mathbf{w}} E(\mathbf{w}) = \nabla_{\mathbf{w}} \mathrm{SSE}(\mathbf{w}) + 2\lambda \mathbf{w}$

3. Lasso Regression

- Một vấn đề của Ridge Regression là sẽ không thu nhỏ các hệ số về chính xác bằng 0.
- Một giải pháp khác đó là Lasso (least absolute shrinkage and selection operator)
- Lasso hoạt động khá giống Ridge regression. Ngoại trừ việc thay đổi regularization

3. Lasso Regression

Các hệ số Lasso sẽ giảm thiểu công thức:

$$\text{Minimize: } f(\mathbf{w}) = \text{SSE} + \lambda\ell_1 = \sum_{i=1}^{N}\left|y_i - \sum_{j=1}^{P} x_{ij}w_j\right|^2 + \lambda\sum_{j=1}^{P}|w_j|$$

Trong đó λ là tham số. $\mathbf{X}$ là dữ liệu huấn luyện. $\mathbf{y}$ là dự đoán. Regularization $\ell_1 = \|\mathbf{w}\|_1 = \sum_{j=1}^{P}|w_j|$ khác với regularization $\ell_2 = \|\mathbf{w}\|_2^2 = \sum_{j=1}^{P} w_j^2$

3. Lasso Regression

Phân tích

$$\sum_{i=1}^{N}\left|y_i - \sum_{j=1}^{P} x_{ij}w_j\right|^2 + \lambda\sum_{j=1}^{P}|w_j| = \text{SSE} + \lambda\ell_1$$

Giả sử $\mathbf{w} = (w_1, w_2)$

$$\text{SSE} = \sum_{i=1}^{N}|y_i - w_1 - x_i w_2|^2$$

$$= \sum_{i=1}^{n}({y_i}^2 - y_i w_1 - y_i x_i w_2 + 2w_1 x_i w_2 + {w_1}^2 + {x_i}^2 {w_2}^2)$$

$$= \sum_{i=1}^{n} y_i^2 - w_1\sum_{i=1}^{n} y_i - w_2\sum_{i=1}^{n} y_i x_i + 2w_1 w_2\sum_{i=1}^{n} x_i + nw_1^2 + w_2^2\sum_{i=1}^{n} x_i^2$$

3. Lasso Regression

Phân tích

$$\sum_{i=1}^{N}\left|y_i - \sum_{j=1}^{P} x_{ij}w_j\right|^2 + \lambda\sum_{j=1}^{P}|w_j| = \text{SSE} + \lambda\ell_1$$

$$\text{SSE} = \sum_{i=1}^{n} y_i^2 - w_1\sum_{i=1}^{n} y_i - w_2\sum_{i=1}^{n} y_i x_i + 2w_1w_2\sum_{i=1}^{n} x_i + nw_1^2 + w_2^2\sum_{i=1}^{n} x_i^2$$

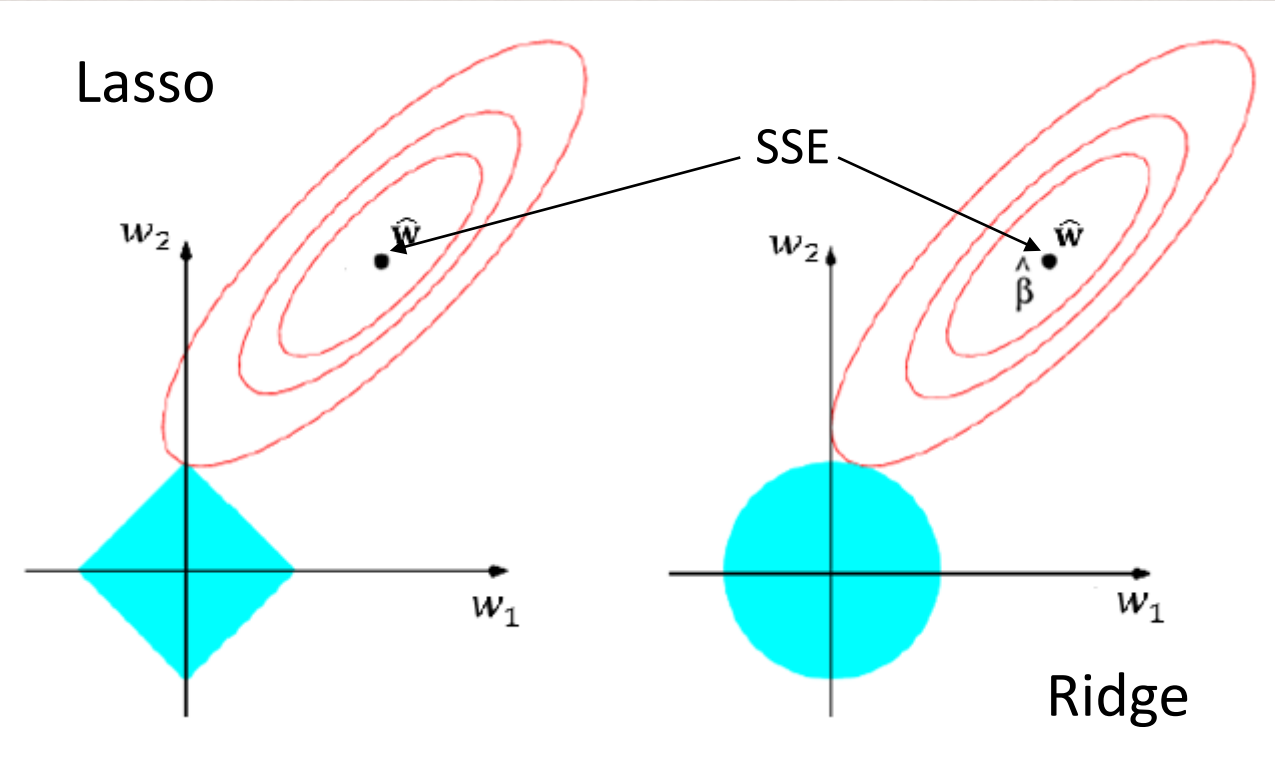
SSE chính là công thức hình elip có dạng:

$$Aw_1^2 + Bw_1w_2 + Cw_2^2 + Dw_1 + Ew_2 + F = 0$$

3. Lasso Regression

Phân tích

Và ℓ_1 sẽ có dạng hình thoi vì $|w_1| + |w_2|$ công thức đối trị tuyệt đối đối xứng qua các trục Ox và Oy



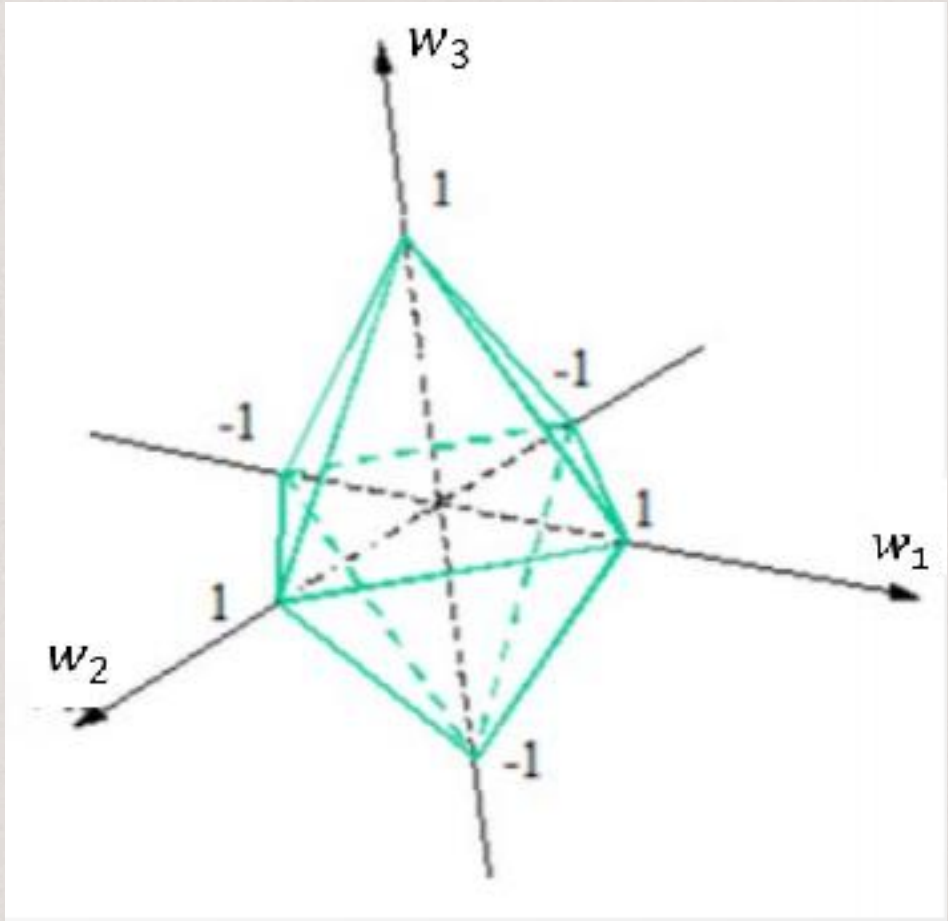
Lasso: Minimize SSE + $\lambda\ell_1$

Ridge: Minimize SSE + $\lambda\ell_2$

3. Lasso Regression

Phân tích

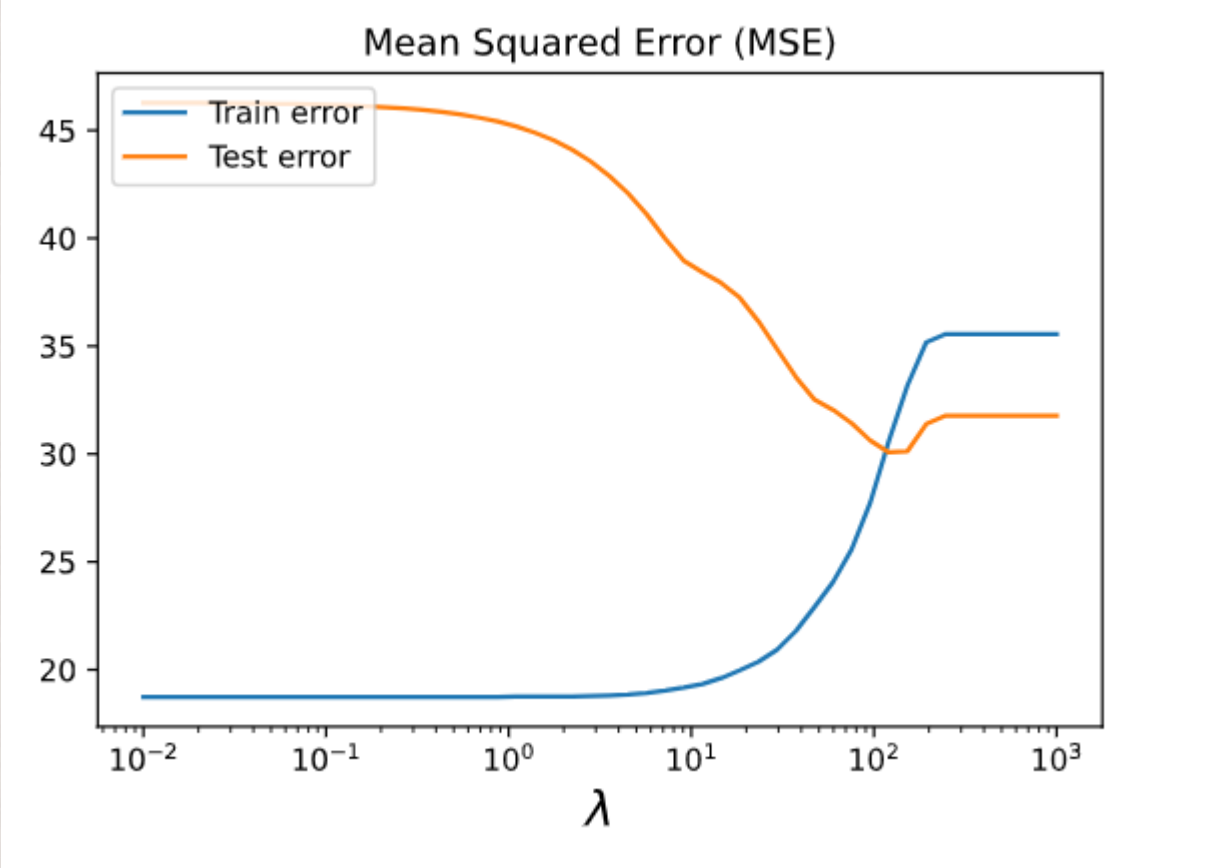
Tương tự với vector có 3 trọng số: $\mathbf{w} = (w_1, w_2, w_3)$



3. Lasso Regression

Chọn tham số λ

Tham số λ tăng sẽ giảm sự quá khớp với dữ liệu test. Qua đó việc đánh giá mô hình sẽ ổn định hơn.



3. Lasso Regression

So sánh đặc điểm Lasso và Ridge

- Bởi vì xu hướng thu hẹp các hệ số dần về 0, nên Lasso tạo ra một mô hình dự đoán đơn giản hơn.
- Giống Ridge ở điểm: Khi λ tăng → phương sai giảm và độ lệch tăng
- Nhược điểm Lasso: Số lượng biến bị giới hạn bởi bộ dữ liệu. Và việc các = 0 nhiều dẫn đến thông tin bị mất đi

3. Lasso Regression

Elastic Net

Là sự kết hợp giữa Lasso và Ridge:

$$\text{Minimize: } f(\mathbf{w}) = \text{SSE} + \lambda_1 \ell_1 + \lambda_2 \ell_2$$

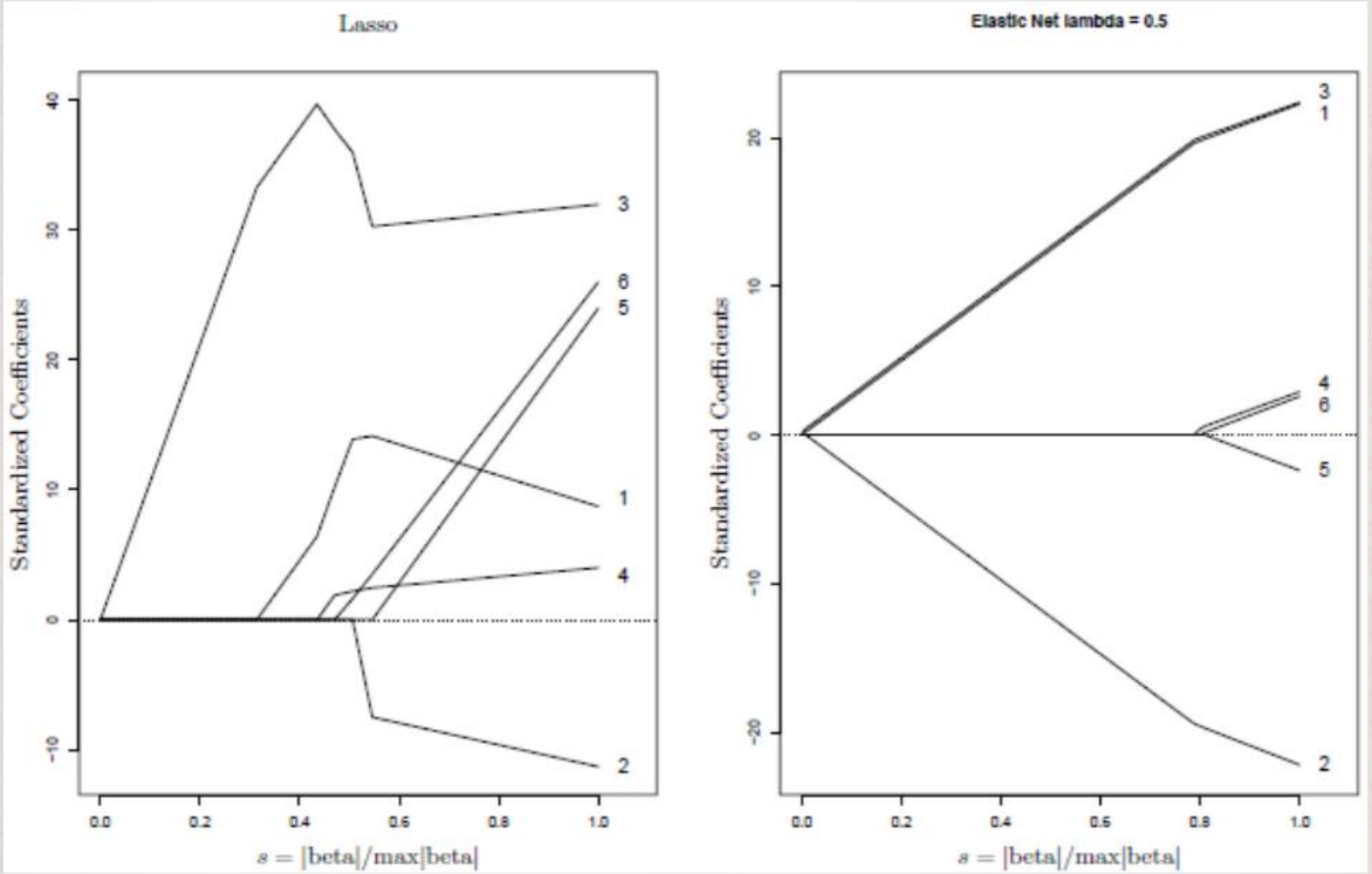
Trong đó :

- ℓ_1: Buộc các biến nhất định bằng 0
- ℓ_2: Giúp hệ số các biến tương quan với nhau có xu hướng bằng nhau

  Ổn định ℓ_1. Do đó cải thiện dự đoán

3. Lasso Regression

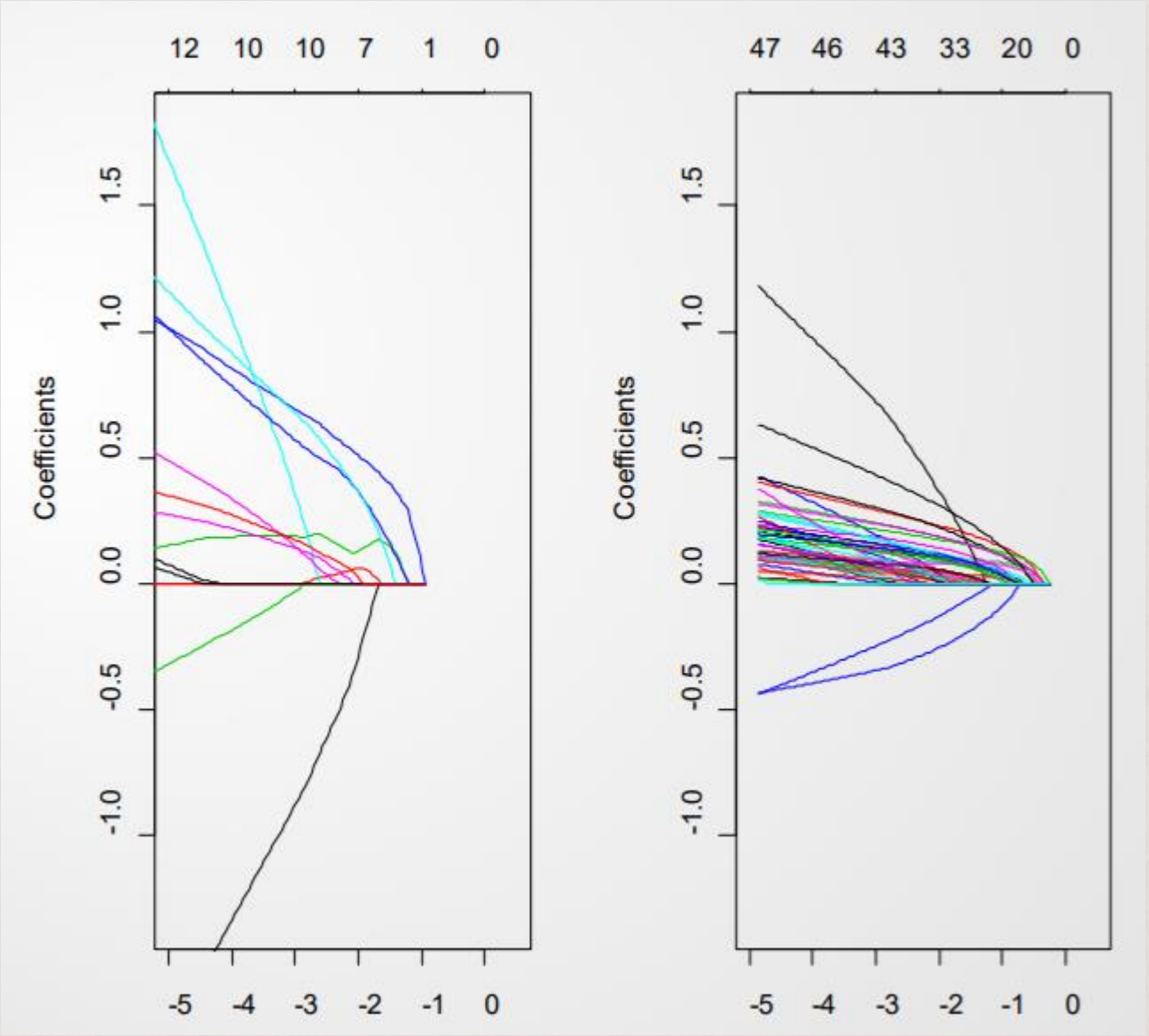
Elastic Net và Lasso



3. Lasso Regression

Elastic Net và Lasso

Elastic Net có nhiều hệ số khác 0 hơn so với Lasso nhưng với cường độ nhỏ hơn.



4. Các vấn đề khi thực hiện regularization

Việc chính quy hóa giúp cho những mô hình phức tạp có thể được huấn luyện trên các tập dữ liệu nhỏ, làm giảm thiểu độ phức tạp của mô hình. Tuy nhiên vấn đề về tìm độ phức tạp tối ưu của mô hình là tương đương với việc chọn hàm chính quy hóa E_{W} và hệ số chính quy λ phù hợp. Một trong các cách đó có thể được liệt kê như sau:

- **Kiểm tra chéo k phần(K-fold cross validation)**: Chia Data thành 3 phần(Train, Test, Evaluate) và chia tập train thành k phần
- **Elastic Net Regression**: $E(\mathbf{w}) = E_{\mathrm{D}}(\mathbf{w}) + \lambda_1 \ell_1(\mathbf{w}) + \lambda_2 \ell_2(\mathbf{w})$

5. Đánh đổi giữa độ lệch (Bias) và Phương sai (Variance)

Việc tối ưu ước lượng hợp lý lớn nhất hay tổng bình phương lỗi nhỏ nhất có thể làm vấn đề quá khớp nghiêm trọng. Tuy nhiên, việc hạn chế số lượng hàm phân lớp nhằm để tránh sự quá khớp làm giảm đi độ linh hoạt của mô hình để nắm bắt được những xu hướng thú vị và quan trọng của dữ liệu. Mặc dù đưa vào hệ số chính quy hóa λ để kiểm soát sự quá khớp, điều này dẫn đến các vấn đề khác đã được nêu ở trên, và rõ ràng nếu ta xem λ như là một biến để tối ưu hàm lỗi dựa trên dữ liệu, thì $\lambda = 0$ là một đáp án. Rõ ràng đây không phải cách để giải quyết vấn đề này.

5. Đánh đổi giữa độ lệch (Bias) và Phương sai (Variance)

Đặt $h(\mathbf{x})$ là hàm đưa ra dự đoán tối ưu,

$$h(\mathbf{x}) = \mathbb{E}[t|\mathbf{x}] = \int tp(t|\mathbf{x})dt$$

Trong đó , $\mathbf{x}$ là vector đầu vào, t là biến mục tiêu

Ta tính kì vọng mất mát khi chọn hàm mất mát $\mathcal{L}\big(t, y(\mathbf{x})\big) = \big(t - y(\mathbf{x})\big)^2$ như sau

$$\mathbb{E}\left[\mathcal{L}\right] = \iint \mathcal{L}\big(t, y(\mathbf{x})\big)p(\mathbf{x}, t)d\mathbf{x}dt = \iint \big(t - y(\mathbf{x})\big)^2 p(\mathbf{x}, t)d\mathbf{x}dt$$

5. Đánh đổi giữa độ lệch (Bias) và Phương sai (Variance)

Ta thêm bớt một lượng $h(\mathbf{x})$ vào $t - y$

$$\mathbb{E}[\mathcal{L}] = \iint \big(t - h(\mathbf{x}) + h(\mathbf{x}) - y(\mathbf{x})\big)^2 p(\mathbf{x}, t) d\mathbf{x} dt$$

$$= \iint \Big((h(\mathbf{x}) - t)^2 + \big(y(\mathbf{x}) - h(\mathbf{x})\big)^2 + 2\big(t - h(\mathbf{x})\big)\big(h(\mathbf{x}) - y(\mathbf{x})\big) \Big) d\mathbf{x} dt$$

$$= \int (h(\mathbf{x}) - t)^2 p(\mathbf{x}, t) d\mathbf{x} dt + \int \big(y(\mathbf{x}) - h(\mathbf{x})\big)^2 p(\mathbf{x}) d\mathbf{x} = \text{variance} + \text{bias}^2$$

Lấy tích phân trên lớp t trước

5. Đánh đổi giữa độ lệch (Bias) và Phương sai (Variance)

Tuy nhiên, ta làm việc với bộ dữ liệu giới hạn, nên ta sẽ viết là như sau

$$\mathbb{E}_{\mathcal{D}}\left[\mathcal{L}(\mathbf{x},\mathcal{D})\right] = \left(\mathbb{E}_{\mathcal{D}}\left[y(\mathbf{x},\mathcal{D})\right] - h(\mathbf{x})\right)^2 + \mathbb{E}_{\mathcal{D}}\left[\left(y(\mathbf{x},\mathcal{D}) - \mathbb{E}_{\mathcal{D}}\left[y(\mathbf{x},\mathcal{D})\right]\right)^2\right]$$

Trong đó, $\mathbb{E}_{\mathcal{D}}\left[y(\mathbf{x},\mathcal{D})\right] - h(\mathbf{x})$ là độ lệch (bias)

$\mathbb{E}_{\mathcal{D}}\left[\left(y(\mathbf{x},\mathcal{D}) - \mathbb{E}_{\mathcal{D}}\left[y(\mathbf{x},\mathcal{D})\right]\right)^2\right]$ là phương sai (variance)

Vậy kì vọng lỗi là tổng của binh phương độ lệch, phương sai và nhiễu. Trong đó nhiễu là bản chất xác suất của bài toán, không thể tối ưu hóa được, có thể xem là hằng số

5. Đánh đổi giữa độ lệch (Bias) và Phương sai (Variance)

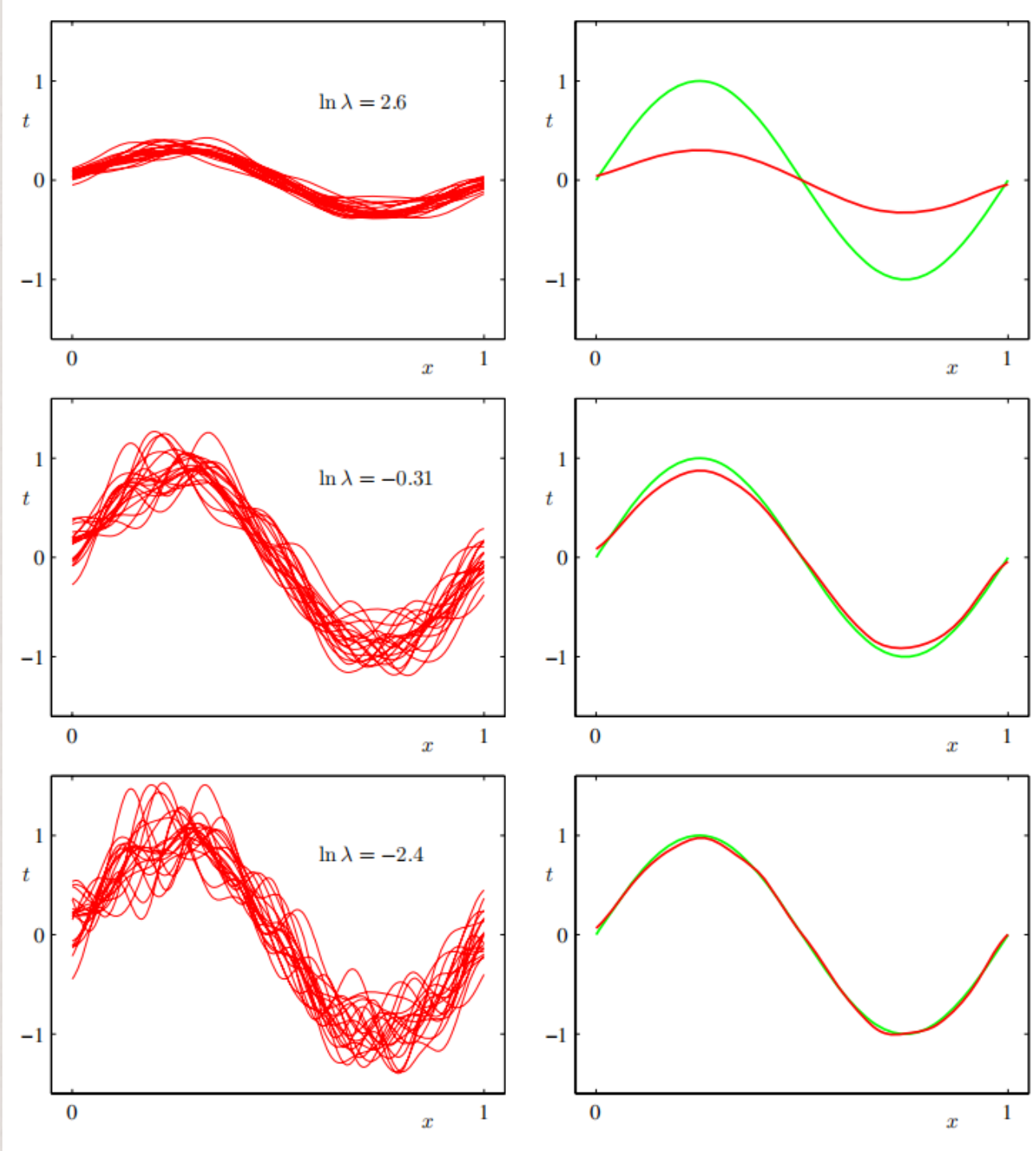
Mục đích của chúng ta là tối thiểu hóa kì vọng lỗi mà ta đã đưa về 3 thành phần.

Sau đây chúng ta sẽ thấy rằng có một sự đánh đổi giữa thiên kiến và phương sai, ví dụ về 100 bộ dữ liệu trong đó mỗi bộ dữ liệu chứa 25 điểm dữ liệu độc lập với $h(x) = \sin(2\pi x)$.

Các bộ dữ liệu l được đánh số từ 1 đến L, với L = 100.

Với mỗi bộ dữ liệu ta sẽ tối ưu dùng một mô hình y_l với 24 hàm phân lớp Gauss với các hệ số chính quy khác nhau:

5. Đánh đổi giữa độ lệch (Bias) và Phương sai (Variance)



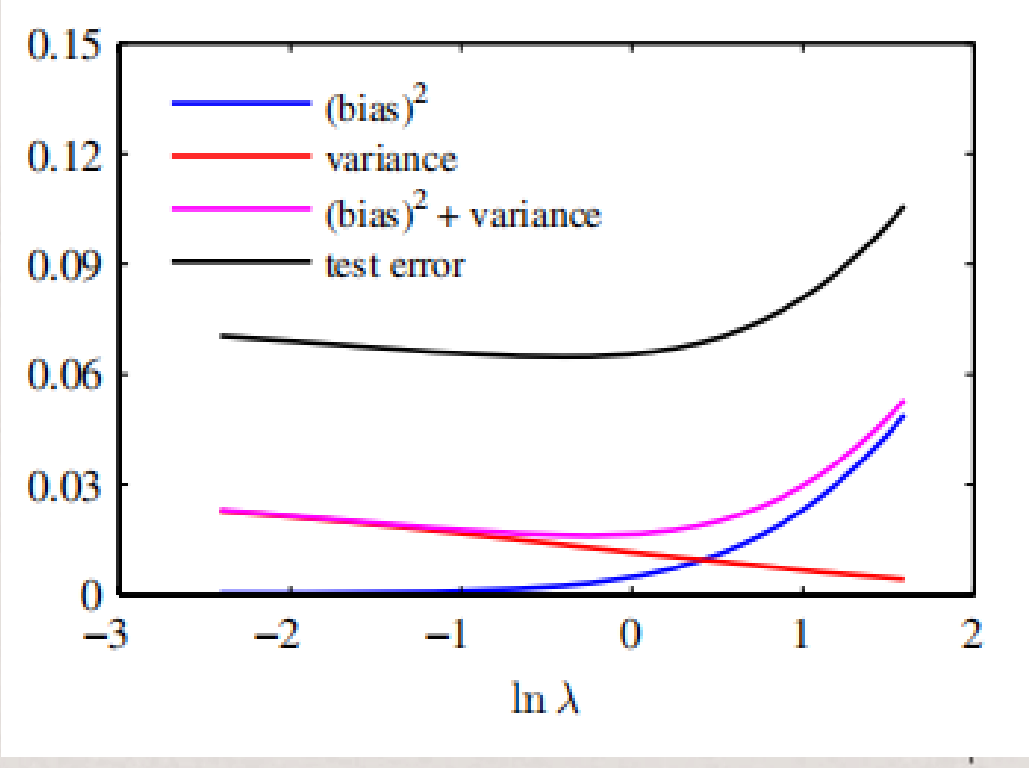
5. Đánh đổi giữa độ lệch (Bias) và Phương sai (Variance)

Đường màu đỏ bên trái là 100 đường y_l ,

Đường màu đỏ bên phải là trung bình của các

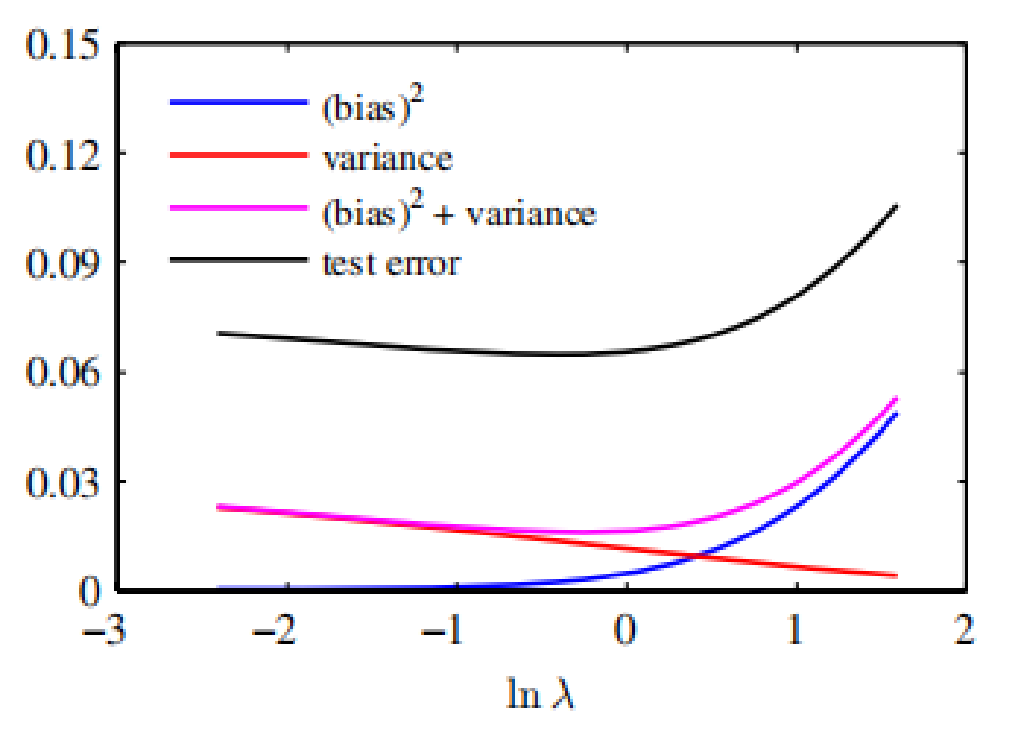
Đường màu đỏ bên trái và đường màu xanh là $h(x)$

Trung bình các đáp án cho một mô hình phức tạp có thể là đưa ra kết quả rất hợp lý. Thật vậy, trung bình là trọng tâm của cách tiếp cận Bayes(nhưng không phải trung bình các bộ dữ liệu mà dựa trên xác suất hậu nghiệm của các tham số).



Biểu đồ quan hệ giữa bias^2, variance ,bias^2 + variance , lỗi với hệ số chính quy λ :

5. Đánh đổi giữa độ lệch (Bias) và Phương sai (Variance)

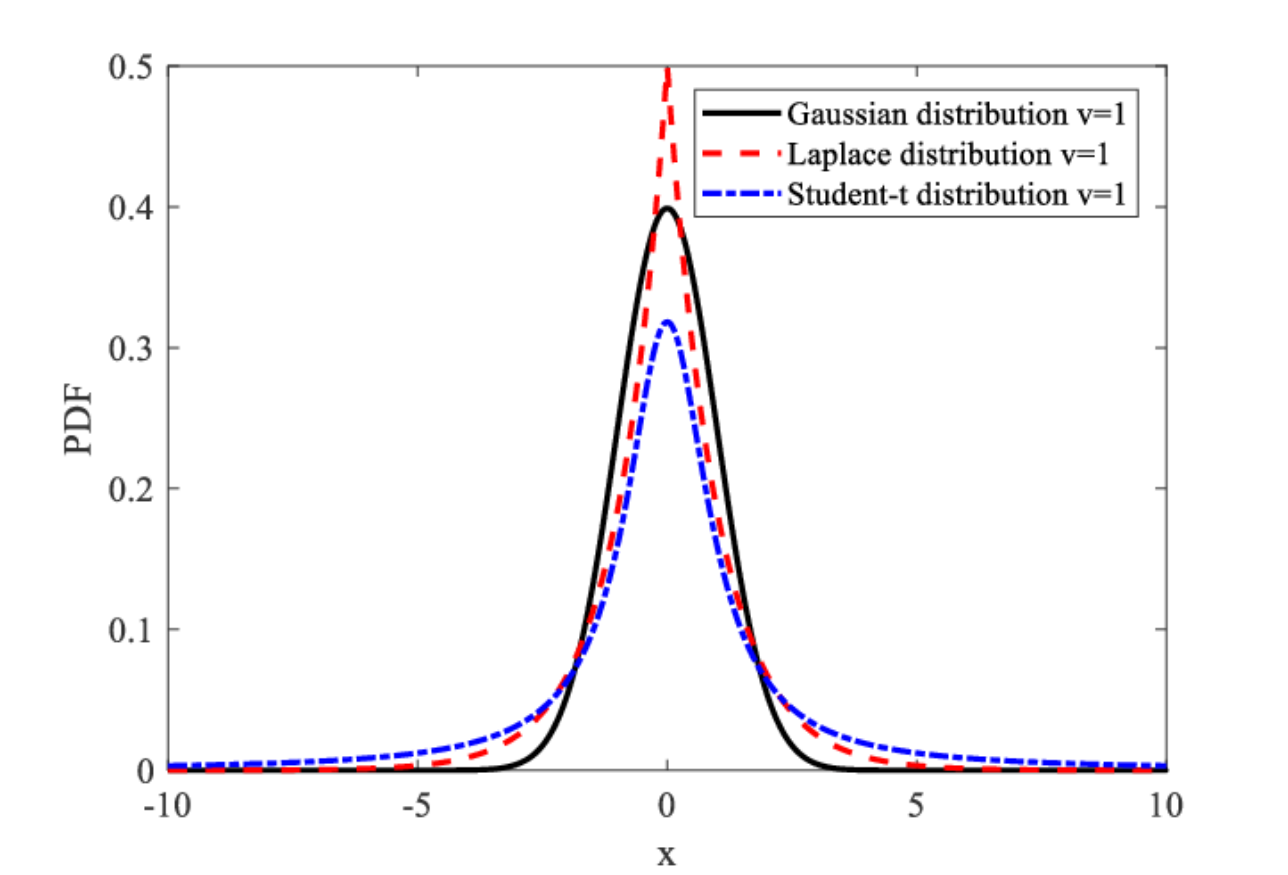


Biểu đồ quan hệ giữa bias^2, variance ,bias^2 + variance , lỗi với hệ số chính quy λ :

Trước khi đi qua phần Bayesian Linear Regression, chúng ta cũng sẽ rút qua công thức trong phần này

$$\text{bias}^2 = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{L}\sum_{l=1}^{L} y_l(x_n) - h(x_n)\right)^2$$

$$\text{variance} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{L}\sum_{l=1}^{L}\left(y_l(x_n) - \frac{1}{L}\sum_{l=1}^{L} y_l(x_n)\right)^2$$



Robust Linear Regression

Trong regression model, ta thường sử dụng phân phối Gauss:

$r_n \sim \mathcal{N}(0, \sigma^2)$ với $r_n = y_n - \mathbf{w}^{\mathrm{T}}\mathbf{x}$

Trong bài toán có **dữ liệu nhiễu**, ta sẽ thay thế **phân phối Gauss** bằng các phân phối khác **heavy tail** hơn (eg: Student, Laplace)

1. Student-t likelihood

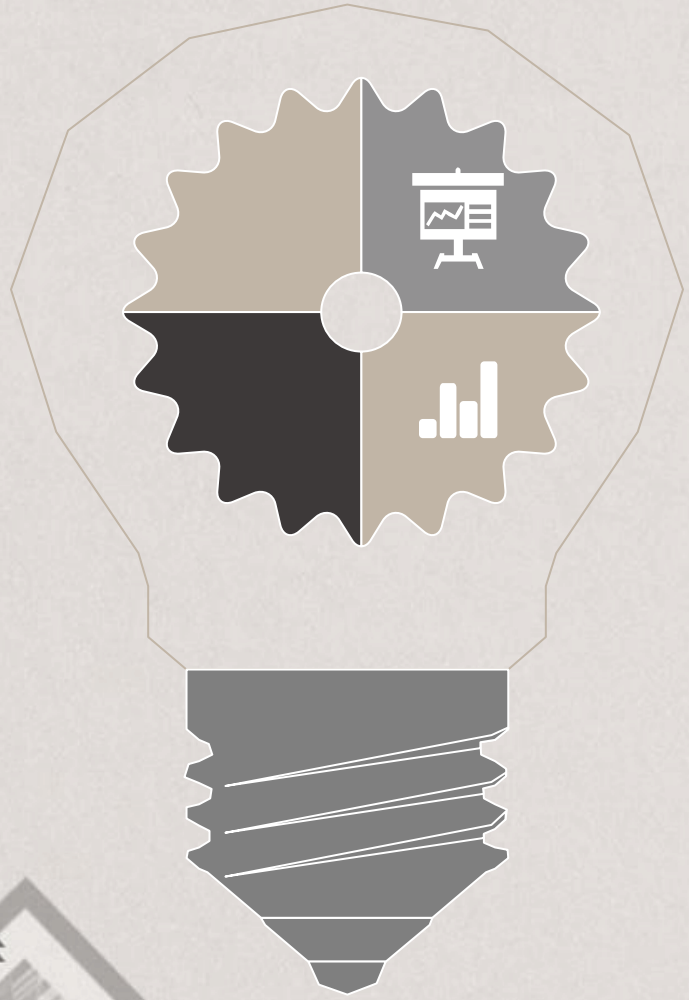
$$p(y|\mathbf{x}, \mathbf{w}, \sigma^2, \nu) = \mathcal{T}\left(y\middle|\mathbf{w}^{\mathrm{T}}\mathbf{x}, \sigma^2, \nu\right)$$

Ta sẽ sử dụng **thuật toán EM** để tính toán MLE

Ý tưởng thuật toán:

- Hàm Likelihood: $L(\boldsymbol{\theta}) = p(\mathbf{y}, \mathbf{z}|\boldsymbol{\theta})$
- Đầu tiên, EM sẽ gán $\boldsymbol{\theta}$ với bộ dữ liệu khởi điểm, sau đó EM sẽ thực hiện vòng lặp như sau cho đến khi $\boldsymbol{\theta}$ hội tụ, tại vòng lặp $t + 1$
  - E-step: Tính kì vọng $Q(\boldsymbol{\theta}|\boldsymbol{\theta}^t) = \mathbb{E}_{\mathbf{z}|\mathbf{x},\boldsymbol{\theta}}[\log L(\boldsymbol{\theta})]$
  - M-step: Ước lượng tham số cực đại ở E-step: $\boldsymbol{\theta}^{t+1} = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, Q(\boldsymbol{\theta}|\boldsymbol{\theta}^t)$

2. Laplace likelihood



$$p(y|\mathbf{x}, \mathbf{w}, b) = \text{Lap}(y|\mathbf{w}^{\text{T}}\mathbf{x}, b) \propto \exp\left(-\frac{1}{b}|y - \mathbf{w}^{\text{T}}\mathbf{x}|\right)$$

Ta sẽ sử dụng **quy hoạch tuyến tính** để tính toán MLE

2. Laplace likelihood

Ý tưởng thuật toán:

Ta cần tìm $\text{argmin}_{\mathbf{v}}\mathbf{c}^{\mathrm{T}}\mathbf{v}$ thỏa mãn $\mathbf{A}\mathbf{v} \leq \mathbf{b}$

Định nghĩa $\mathbf{v} = (w_1, w_2, \dots, w_D, e_1, e_2, \dots, e_N) \in \mathbb{R}^n$, trong đó $e_i = |y_i - \hat{y}_i|$

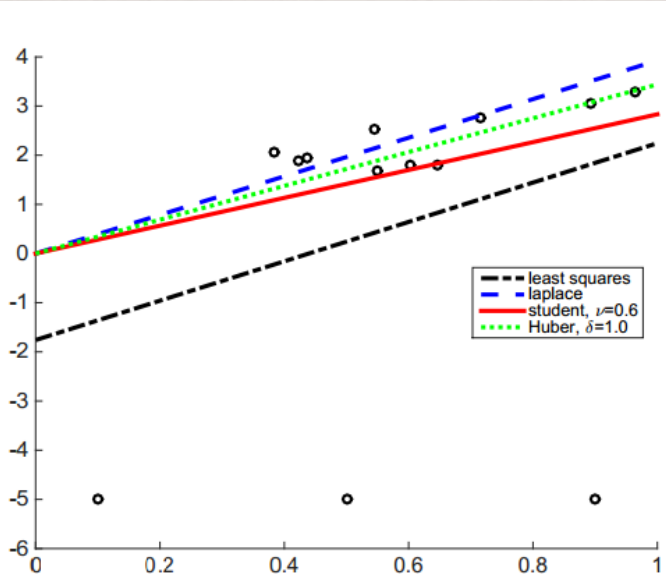
Vì chúng ta cần tối thiểu hóa các sai số tuyệt đối nên định nghĩa $\mathbf{c} = (0, \dots, 0, 1, \dots, 1)$

Ở đây chúng ta sẽ tính được

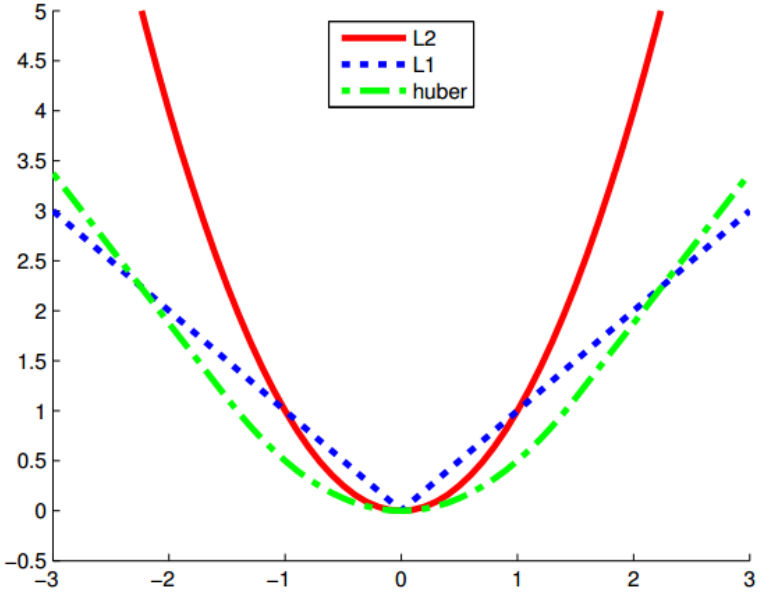
$$\mathbf{A} = \begin{pmatrix} x_1 & 1 & 0 & 0 & 0 & \cdots & 0 \\ -x_1 & 1 & 0 & 0 & 0 & \cdots & 0 \\ x_2 & 0 & 1 & 0 & 0 & \cdots & 0 \\ -x_2 & 0 & 1 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_N & 0 & 0 & 0 & 0 & \cdots & 1 \\ -x_N & 0 & 0 & 0 & 0 & \cdots & 1 \end{pmatrix}, \mathbf{b} = (y_1, -y_1, y_2, -y_2, \cdots, y_N, -y_N)$$

3. Huber loss

$$\ell_{huber}(\delta, r) = \begin{cases} r^2/2, & \text{nếu} \quad |r| \le \delta \\ \delta|r| - \delta^2/2, & \text{nếu} \quad |r| > \delta \end{cases}$$



(a)



(b)

4. RANSAC

RANSAC (random sample consensus - sự đồng thuận của các mẫu ngẫu nhiên)

Trước hết, ta giả sử tất cả các điểm đều là inliers, và tính toán $\mathbf{w}_0$ với mỗi vòng lặp t, chúng ta xác định các outliers là các điểm có khoảng cách lớn so với model $\widehat{\mathbf{w}}_t$, rồi loại bỏ chúng ra khỏi model, rồi tính toán trên các điểm còn lại để có $\widehat{\mathbf{w}}_{t+1}$

Bayesian Linear Regression

Phân phối dự đoán hậu nghiệm

Hàm likelihood

Xác suất tiền nghiệm

mang ý nghĩa **niềm tin ban đầu** của người quan sát về khả năng **w** đúng

$$p(\mathbf{w}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{w})p(\mathbf{w})}{p(\mathcal{D})}$$

Xác suất hậu nghiệm

mang ý nghĩa **niềm tin sau khi quan** về khả năng **w** đúng

$$p(\mathbf{y}|\mathbf{x}, \mathcal{D}) = \int p(\mathbf{y}|\mathbf{x}, \mathbf{w})p(\mathbf{w}|\mathcal{D})d\mathbf{w}$$

Phân phối dự đoán hậu nghiệm mang ý nghĩa niềm tin về output tương lai là đúng

Bayesian Linear Regression

Likelihood	Tiền nghiệm	Hậu nghiệm	Tên gọi
Gaussian	Uniform	Suy biến	Least Squares Linear Regression
Gaussian	Gaussian	Suy biến	Ridge Regression
Gaussian	Laplace	Suy biến	Lasso Regression
Student-t	Uniform	Suy biến	Robust Regression
Laplace	Uniform	Suy biến	Robust Regression
Gaussian	Gauss-gama	Gauss-gama	Bayesian Linear Regression

Phân phối suy biến được tính là $\delta(\widehat{\mathbf{w}} - \mathbf{w})$. Trong đó, $\widehat{\mathbf{w}}$ là MAP estimation.

MLE được sử dụng với xác suất tiền nghiệm Uniform (đều) và xác suất hậu nghiệm là phân phối suy biến

1. Tính toán xác suất hậu nghiệm

Xác suất tiền nghiệm: $p(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}\middle|\breve{\mathbf{w}}, \breve{\boldsymbol{\Sigma}}\right)$

Hàm likelihood: $p(\mathcal{D}|\mathbf{w}, \sigma^2) = \prod_n p\left(y_n, \mathbf{w}^{\mathrm{T}}\mathbf{x}, \sigma^2\right) = \mathcal{N}\left(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}_N\right)$

Xác suất hậu nghiệm $\propto$ Hàm likelihood $\times$ Xác suất tiền nghiệm

Xác suất hậu nghiệm

$$p(\mathbf{w}|\mathbf{x}, \mathbf{y}, \sigma^2) \propto \mathcal{N}\left(\mathbf{w}\middle|\breve{\mathbf{w}}, \breve{\boldsymbol{\Sigma}}\right)\mathcal{N}\left(\mathbf{y}|\mathbf{X}\mathbf{w}, \sigma^2\mathbf{I}_N\right) = \mathcal{N}\left(\mathbf{w}|\mathbf{w}_N, \boldsymbol{\Sigma}_N\right)$$

trong đó, $\mathbf{w}_N \triangleq \boldsymbol{\Sigma}_N\left(\breve{\boldsymbol{\Sigma}}^{-1}\breve{\mathbf{w}} + \frac{1}{\sigma^2}\mathbf{X}^{\mathrm{T}}\mathbf{y}\right)$

$$\boldsymbol{\Sigma}_N \triangleq \left(\breve{\boldsymbol{\Sigma}}^{-1} + \frac{1}{\sigma^2}\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}$$

2. Tính toán phân phối dự đoán hậu nghiệm

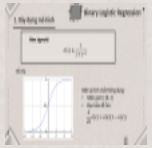
Phân phối dự đoán hậu nghiệm

$$p(\mathbf{y}|\mathbf{x}, \mathcal{D}, \sigma^2) = \int \mathcal{N}\left(\mathbf{w}\middle|\mathbf{x}^{\mathrm{T}}\mathbf{w}, \sigma^2\right)\mathcal{N}(\mathbf{w}|\mathbf{w}_N, \boldsymbol{\Sigma}_N)d\mathbf{w} = \mathcal{N}\left(y\middle|\mathbf{w}_N^{\mathrm{T}}\mathbf{x}, \sigma_N(\mathbf{x})\right)$$

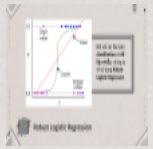
trong đó, $\sigma_N(\mathbf{x}) \triangleq \sigma^2 + \mathbf{x}^{\mathrm{T}}\boldsymbol{\Sigma}_N\mathbf{x}$

Logistic Regression

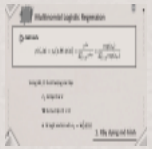
Logistic Regression



Binary Logistic Regression



Robust Logistic Regression



Multinomial Logistic Regression

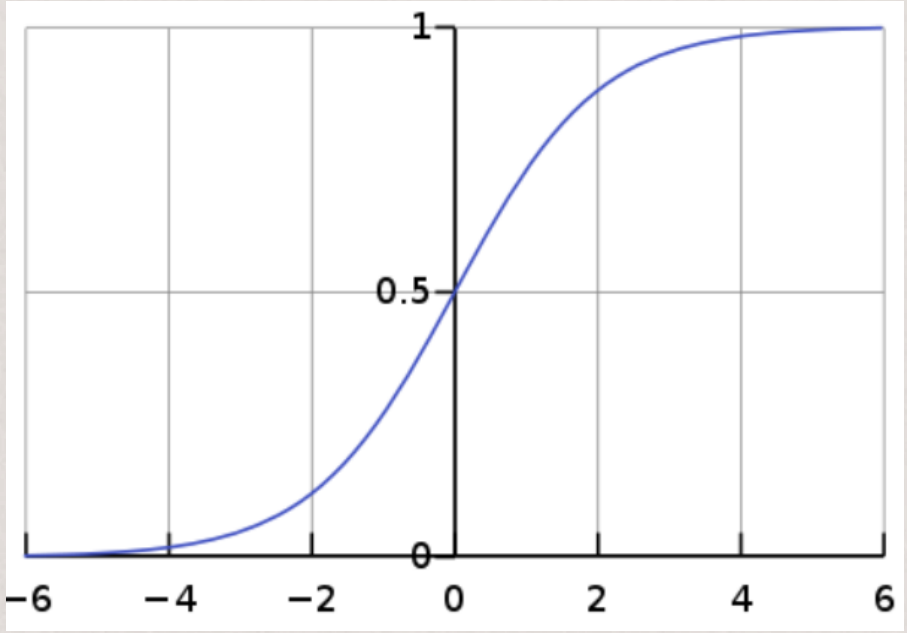
Bayesian Logistic Regression

1. Xây dựng mô hình

Hàm sigmoid:

$$\sigma(x) \triangleq \frac{1}{1+e^{-x}}$$

Đồ thị:



Một vài tính chất thông dụng:

- Miền giá trị: [0, 1]
- Đạo hàm dễ tìm:

$$\frac{d}{dx}\sigma(x) = \sigma(x)(1-\sigma(x))$$

1. Xây dựng mô hình

Phân phối Bernoulli:

$$p(y) = \text{Ber}(y|\lambda) = \lambda^y(1-y)^{1-y}$$

trong đó $p(y=1) = \lambda$ và $p(y=0) = 1-\lambda$.

Model cho bài toán:

$$p(y|\mathbf{x}, \boldsymbol{\theta}) = \text{Ber}(y|\sigma(f(\mathbf{x}, \boldsymbol{\theta})))$$

Ở đây, ta sử dụng hàm tuyến tính $f(\mathbf{x}, \boldsymbol{\theta}) = \mathbf{w}^{\text{T}}\mathbf{x} + \text{b}$

2. Maximum Likelihood Estimation

Xác định hàm likelihood:

$$p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{\theta}) = \prod_{n=1}^{N} p(y_n|\mathbf{x}_n, \boldsymbol{\theta}) = \prod_{n=1}^{N} \mathrm{Ber}(y_n|\mu_n)$$

Trong đó, $\mu_n = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x} + \mathrm{b})$

Mục đích: Chọn $\boldsymbol{\theta}$ để tối đa hóa likelihood

2. Maximum Likelihood Estimation

Xác định hàm loss:

$$\mathrm{NLL}(\boldsymbol{\theta}) = -\log \prod_{n=1}^{N} \mathrm{Ber}(y_n|\mu_n) = -\sum_{n=1}^{N} [y_n \log(\mu_n) + (1-y_n)\log(1-\mu_n)]$$

Hàm này chính là **cross entropy**

2. Maximum Likelihood Estimation

Tối ưu hàm mất mát

Ta tính được **Gradient** và **Hesse** như sau

Gradient:

$$\mathbf{g}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \mathrm{NLL}(\boldsymbol{\theta}) = \sum_{n=1}^{N} (\mu_n - y_n)\, \mathbf{x_n}$$

Hesse:

$$\mathbf{H}(\boldsymbol{\theta}) = \nabla^2_{\boldsymbol{\theta}} \mathrm{NLL}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \nabla^{\mathrm{T}}_{\boldsymbol{\theta}} \mathrm{NLL}(\boldsymbol{\theta}) = \mathbf{X}^{\mathrm{T}} \mathrm{S} \mathbf{X}$$

Trong đó, $\mathbf{X} = [\mathbf{x}_1 \;\; \mathbf{x_2} \;\; \ldots \;\; \mathbf{x}_N]$ là ma trận dữ liệu

$\mathbf{y} = [y_1 \;\; y_2 \;\; \ldots \;\; y_N]^{\mathrm{T}}$ là vector cột biểu thị truth label

$\mathbf{S} \triangleq \mathrm{diag}(\mu_1(1-\mu_1), \ldots, \mu_N(1-\mu_N))$

2. Maximum Likelihood Estimation

Tối ưu hàm mất mát

NLL có đúng một điểm có giá trị nhỏ nhất

Thật vậy, ta thấy $\mathbf{H}$ là ma trận xác định dương bởi với mọi vector $\mathbf{v}$ khác $\mathbf{0}$, ta luôn có

$$\mathbf{v}^{\mathrm{T}}\mathbf{H}\mathbf{v} = \left(\mathbf{v}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{S}^{\frac{1}{2}}\right)\left(\mathbf{S}^{\frac{1}{2}}\mathbf{X}\mathbf{v}\right) = \left\|\mathbf{v}^{\mathrm{T}}\mathbf{X}^{\mathrm{T}}\mathbf{S}^{\frac{1}{2}}\right\|_{\mathbf{2}}^{\mathbf{2}} > 0$$

2. Maximum Likelihood Estimation

Tối ưu hàm mất mát

Tìm nghiệm:

Gradient Descent (Stochastic)

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t(\mu_n - y_n)\mathbf{x}_{\boldsymbol{n}}$$

Newton's Method

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t\mathbf{H}^{-1}(\boldsymbol{\theta}_t)\mathbf{g}(\boldsymbol{\theta}_t)$$

Vì NLL chỉ có đúng một điểm có giá trị nhỏ nhất nên hai phương pháp trên luôn cho nghiệm tối ưu

Non-linear classifier

Bằng cách tiền xử lí dữ liệu thông qua hàm ϕ, ta được classifier mới:

$$f(x, \mathbf{w}) = \mathbf{w}^{\mathrm{T}} \phi(x)$$

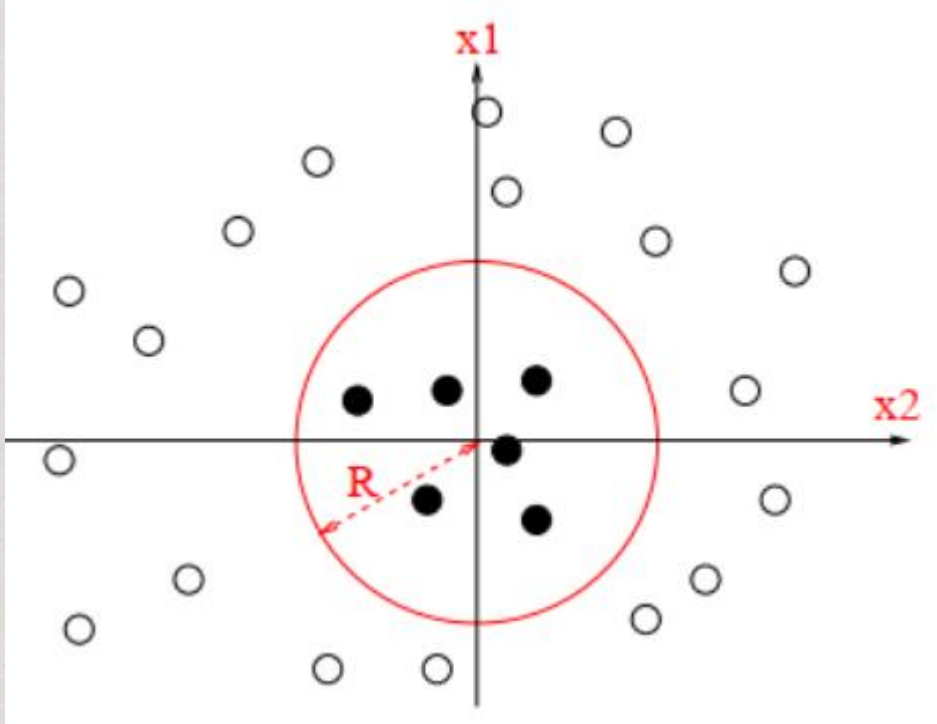
trong đó $\phi(x) = \begin{bmatrix} 1 \\ x \\ x^2 \\ \vdots \\ x^D \end{bmatrix}$

Khi đó, thay vì xử lí $f(x, w) = wx$

ta sẽ xử lí $f(x, w) = w_0 + w_1 x + w_2 x^2 + \cdots + w_D x^D$

Non-linear classifier

Ví dụ:



Xét các điểm dữ liệu có dạng $\mathbf{x} = (x_1, x_2)$ như hình bên.

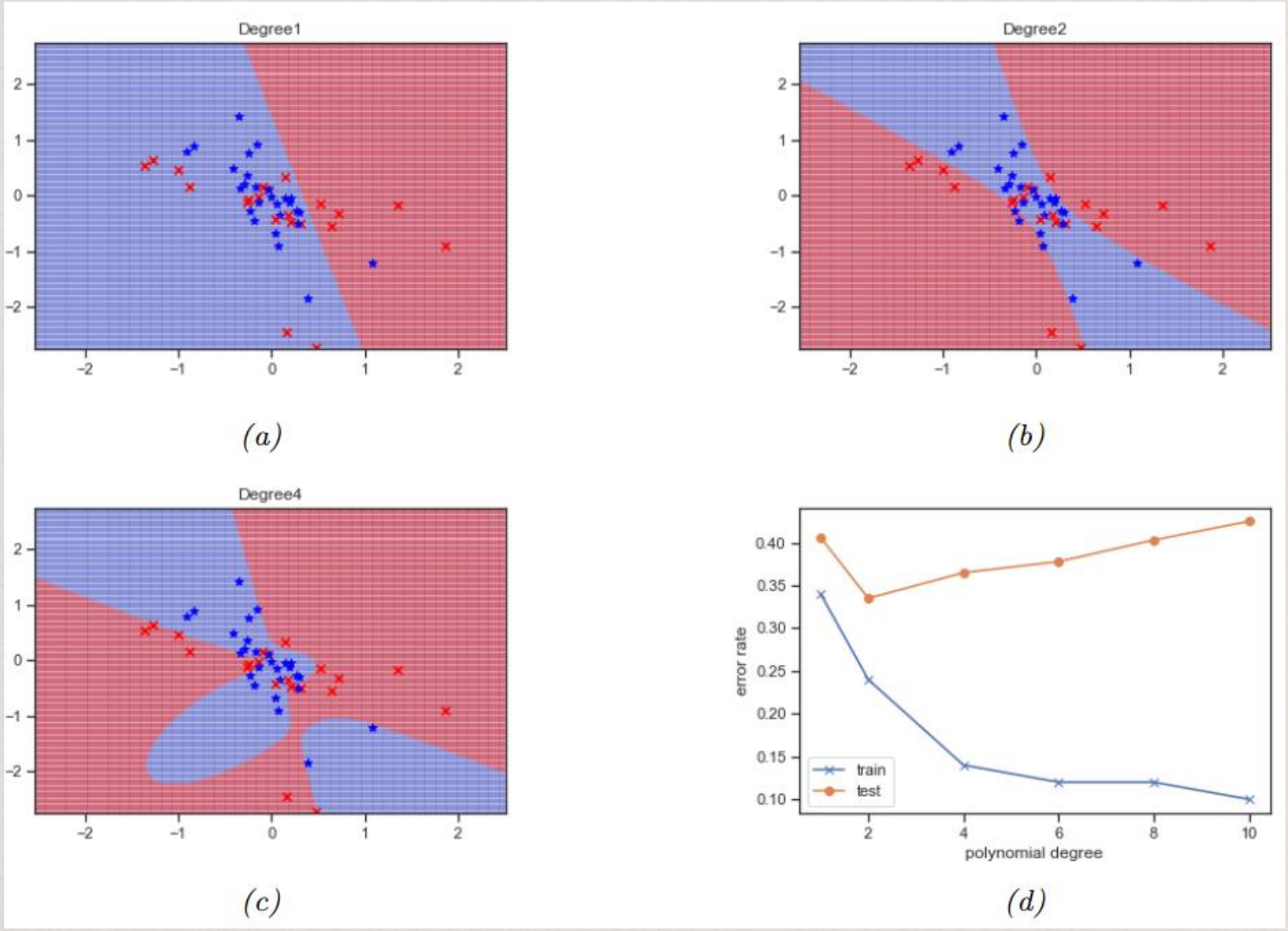
Thay vì sử dụng

$$f_1(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1 + w_2 x_2$$

Ta sẽ sử dụng

$$f_2(\mathbf{x}, \mathbf{w}) = w_0 + w_1 x_1^2 + w_2 x_2^2$$

3. Maximum a posteriori estimation



3. Maximum a posteriori estimation

Để ngăn tình trạng Overfitting, ta thêm vào hàm mất mát một đại lượng phạt

$C(\boldsymbol{\theta}) = -\log(p(\boldsymbol{\theta}))$ trong đó $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\mathbf{0}, C\boldsymbol{I})$

Do đó, ta cần tối thiểu đại lượng mới sau:

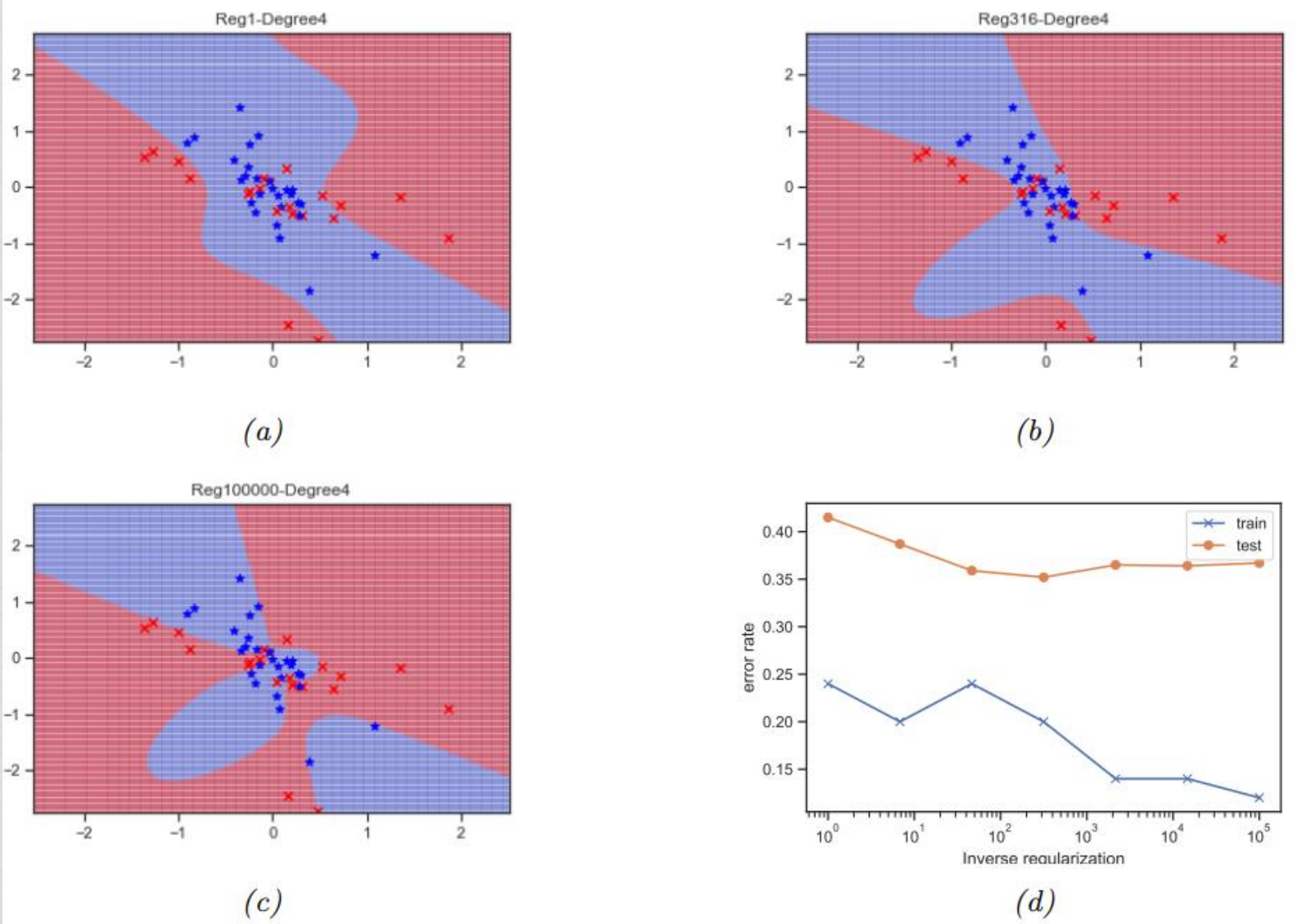
$$E(\boldsymbol{\theta}) = \mathcal{L}(\boldsymbol{\theta}) + C(\boldsymbol{\theta})$$

$$\Leftrightarrow E(\boldsymbol{\theta}) = \mathrm{NLL}(\boldsymbol{\theta}) + \lambda\|\boldsymbol{\theta}\|_2^2$$

trong đó $\lambda = \frac{1}{C}$, và $\nabla_{\boldsymbol{\theta}} E(\boldsymbol{\theta}) = \mathbf{g}(\boldsymbol{\theta}) + 2\lambda\boldsymbol{\theta}$,

$$\nabla_{\boldsymbol{\theta}}^2 E(\boldsymbol{\theta}) = \mathbf{H}(\boldsymbol{\theta}) + 2\lambda\mathbf{I}$$

3. Maximum a posteriori estimation



Multinomial Logistic Regression

Mô hình:

$$p(C_k|\mathbf{x}) = y_k\big(\mathbf{x}, \mathbf{W}, \phi(\mathbf{x})\big) = \frac{e^{a_k}}{\sum_{k'=0}^{K} e^{a_{k'}}} = \frac{\exp(a_k)}{\sum_{k'=0}^{K} \exp(a_{k'})}$$

trong đó, K là số lượng các lớp

C_k là lớp thứ k

$\mathbf{W}$ là ma trận $N \times K$

$\mathbf{a}$ là logit vector với $a_k = \mathbf{w}_k^{\mathrm{T}} \phi(\mathbf{x})$

1. Xây dựng mô hình

1. Xây dựng mô hình

Ở đây Logistic Regression sử dụng cho nhiều phân lớp đã sử dụng hàm Softmax

$S_k = \frac{\exp(a_k)}{\sum_{k'=0}^{K} \exp(a_{k'})}$ để ánh xạ tuyến tính logit a_k từ tập $\mathbb{R} \to [0,1]$ thỏa mãn tính chất $\sum_k^K S_k = 1$

Với $k = 2$, hàm Softmax cũng chính là hàm Sigmoid

$$\frac{e^{xa_1}}{e^{xa_1} + e^{xa_2}} = \frac{1}{1 + e^{x(a_2 - a_1)}}$$

2. Maximum Likelihood Estimation

Xét tập dữ liệu $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N]$ với các giá trị mục tiêu $t_1, t_2, \dots, t_N$ được nhóm lại vector $\mathbf{t}$, trong đó $\mathbf{t}_i$ là các vector nếu như $\mathbf{t}_{ic} = 1$, khi $\mathbf{x}_i = c$

Ta có hàm likelihood như sau.

$$p(t|\mathbf{X}, \mathbf{w}, \phi) = \prod_{n=1}^{N} \prod_{k=1}^{K} p(C_k|\phi(\mathbf{x}_n))^{t_{nk}} = \prod_{n=1}^{N} \prod_{k=1}^{K} y_{nk}^{t_{nk}}$$

Trong đó, đặt $p(C_k|\phi(\mathbf{x}_n)) = y_{nk}$

2. Maximum Likelihood Estimation

Ta xây dựng hàm loss

$$\mathrm{NLL}(\mathbf{w}) = -\log p(t|\mathbf{X}, \mathbf{w}, \phi) = -\sum_{n=1}^{N}\sum_{k=1}^{K} \log(y_{nk}) \times t_{nk}$$

Đặt $\mathrm{MLE} = \hat{\mathbf{w}} = \mathrm{argmin}_{\mathbf{w}} \mathrm{NLL}(\mathbf{w})$

do đó $\mathbf{g}(\hat{\mathbf{w}}) = \nabla_{\mathbf{w}} \mathrm{NLL}(\hat{\mathbf{w}}) = 0$

2. Maximum Likelihood Estimation

Ta có: $\mathbf{g}(\mathbf{w}) = \nabla_{\mathbf{w}_j} \mathrm{NLL}(\mathbf{w}) = \frac{\delta \mathrm{NLL}(\mathbf{w})}{\delta w} = \frac{\delta \mathrm{NLL}(\mathbf{w})}{\delta y} \frac{\delta y}{\delta a} \frac{\delta a}{\delta t}$

$$= -\sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \frac{1}{y_{nk}} y_{nk} \left(I_{kj} - y_{nj}\right) \phi(\mathbf{x})$$

$$= \sum_{n=1}^{N} (t_{nj} - y_{nj}) \phi(\mathbf{x})$$

Từ công thức đạo hàm trên ta dùng SGD để tìm được $\widehat{\mathbf{w}}$ một cách dễ dàng

Ngoài ra ta còn tính được

$$\mathbf{H}(\mathbf{w}) = \nabla_{\mathbf{w}}^2 \mathrm{NLL}(\mathbf{w}) = \sum_{n=1}^{N} y_{nk} \left(I_{kj} - y_{nj}\right) \phi(\mathbf{x}) \phi(\mathbf{x})^{\mathrm{T}}$$

Ta chứng minh được ma trận Hesse trên là ma trận xác định dương, nên ta cũng có thể sử dụng thuật toán Newton-Raphson để tính $\widehat{\mathbf{w}}$

3. Maximum a Posteriori Estimation

Giả sử $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|0, \lambda^{-1}\mathbf{I})$, ta sẽ sử dụng ℓ_2 regularization

Gọi $E_{\mathrm{D}}(\mathbf{w})$ là hàm lỗi của train data

$E_{\mathrm{T}}(\mathbf{w})$ là hàm lỗi của test/real data

$E_{\mathrm{W}}(\mathbf{w})$ là hàm regularization (hàm chinh quy hóa)

Ta có: $E(\mathbf{w}) = E_{\mathrm{D}}(\mathbf{w}) + \lambda E_{\mathrm{W}}(\mathbf{w})$

$$= \mathrm{NLL}(\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2$$

Khi đó $\mathbf{g}(\mathbf{w}) = \nabla_{\mathbf{w}} E(\mathbf{w}) = \nabla_{\mathbf{w}} \mathrm{NLL}(\mathbf{w}) + 2\lambda\mathbf{w}$

Hàm regularization có thể là $\|\mathbf{w}\|_p^p$. Nếu $p = 1$ thì là ℓ_1 regularization (Lasso Regression), $p = 2$ thì là ℓ_2 regularization (Ridge Regression)

4. Maximum Entropy Classifier (MaxEnt)

MaxEnt là model có dạng như sau:

$$p(y = c|\mathbf{x}, \mathbf{w}) = \frac{1}{Z(\mathbf{w}, \mathbf{x})} e^{\mathbf{w}^T \phi(\mathbf{x}, c)}$$

Trong đó, Z là một hàm phân hoạch

$\mathbf{w}$ là weight vector

$\mathbf{x}$ là vector dữ liệu

$\phi(\mathbf{x}, c)$ là vector đặc trưng cho lớp c

Multinomial Logistic Regression là một trường hợp đặc biệt với $\mathbf{w}^{\mathrm{T}} \phi(\mathbf{x}, c) = \mathbf{w}^{\mathrm{T}} c$

4. Maximum Entropy Classifier (MaxEnt)

MaxEnt được sử dụng nhiều trong xử lý ngôn ngữ tự nhiên. Một ví dụ điển hình là bài toán gán nhãn vai trò ngữ nghĩa rằng chúng ta cần phân loại từ $\mathbf{x}$ vào một trong các vai trò như người, nơi chốn hoặc đồ vật. Chúng ta có thể có một số đặc trưng(nhị phân) như:

$\phi_1(y, \mathbf{x}) = \mathbb{I}\,(y = \text{người và } \mathbf{x} \text{ là tên riêng thông thường})$

$\phi_2(y, \mathbf{x}) = \mathbb{I}\,(y = \text{người và } \mathbf{x} \text{ là xuất hiện sau từ “của”})$

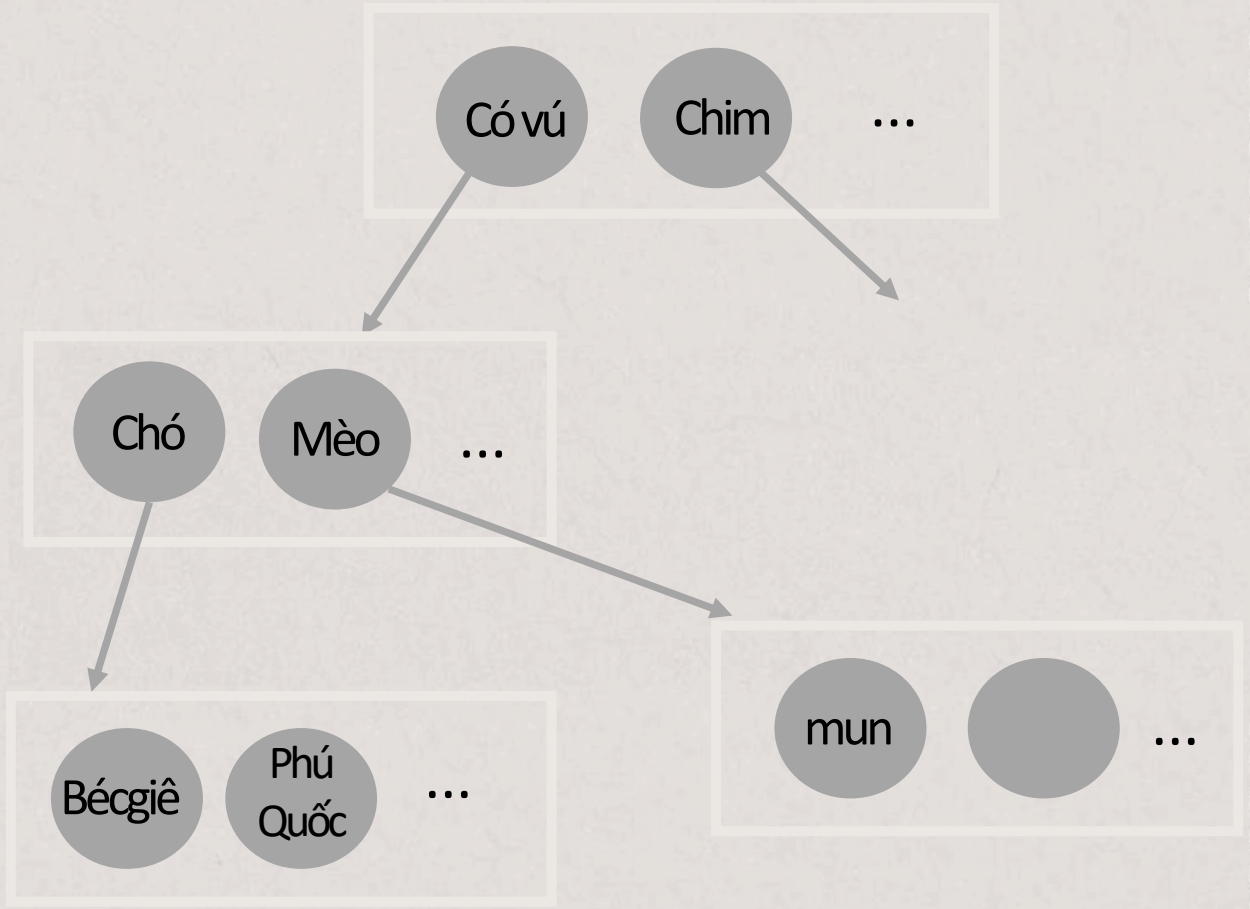
$\phi_3(y, \mathbf{x}) = \mathbb{I}\,(y = \text{nơi chốn và } \mathbf{x} \text{ chứa từ “hồ”})$

4. Maximum Entropy Classifier (MaxEnt)

- Vậy làm sao để tạo ra được các hàm đặc trưng như vậy?
  - Tạo ra thật nhiều hàm bằng tay, sau đó sử dụng các thuật toán chọn lựa hàm đặc trưng, như sử dụng nhóm Lasso(bởi chính quy hóa ℓ_1 sẽ có xu hướng loại bỏ các đặc trưng, hàm phân lớp không cần thiết).
  - Xây dựng thuật toán lần lượt xây dựng và thêm các hàm đặc trưng vào mô hình bằng phương pháp heuristic.

5. Hierarchical classification

Đôi khi, nhãn có thể được tổ chức dưới dạng hình cây. Ví dụ như ta muốn dự đoán rằng đây là con vật gì. Nếu nó là chó, thì nó có thể là giống chó Bécgiê hay giống chó Phú Quốc, ... Giả sử như nó dự đoán rằng p(chó) = 0.89, p(Bécgiê|chó) = 0.3, p(Phú Quốc|chó) = 0.35,... thì tốt nhất mô hình của chúng ta nên gán nhãn con vật đó là con chó.

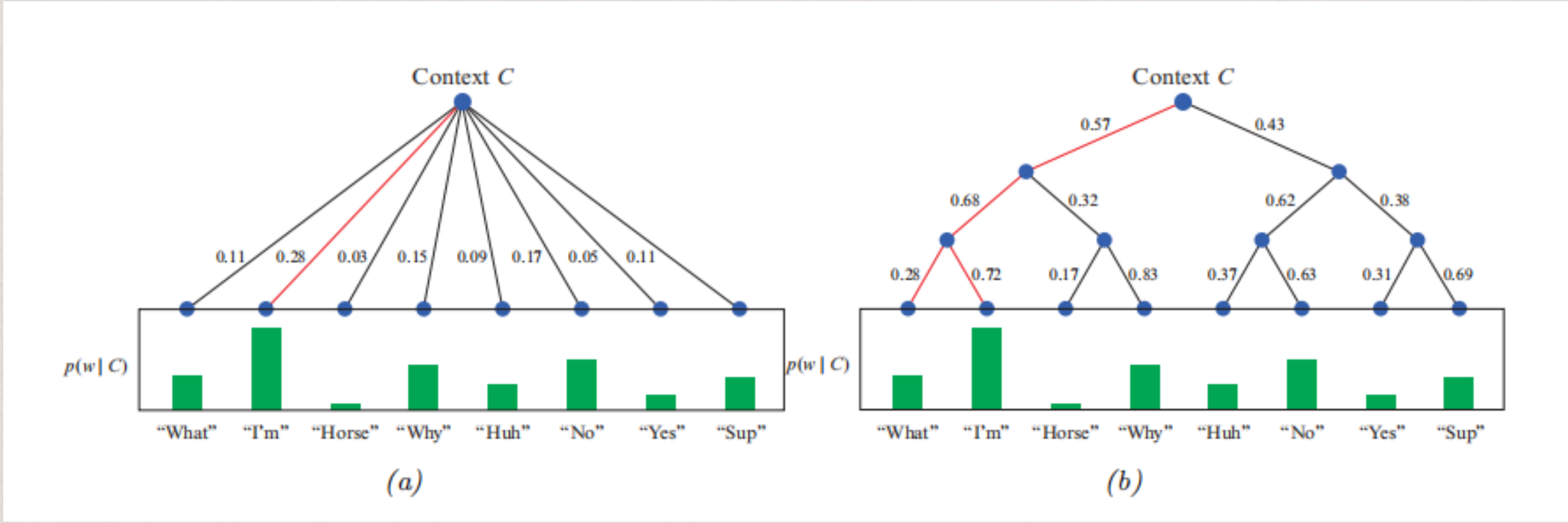


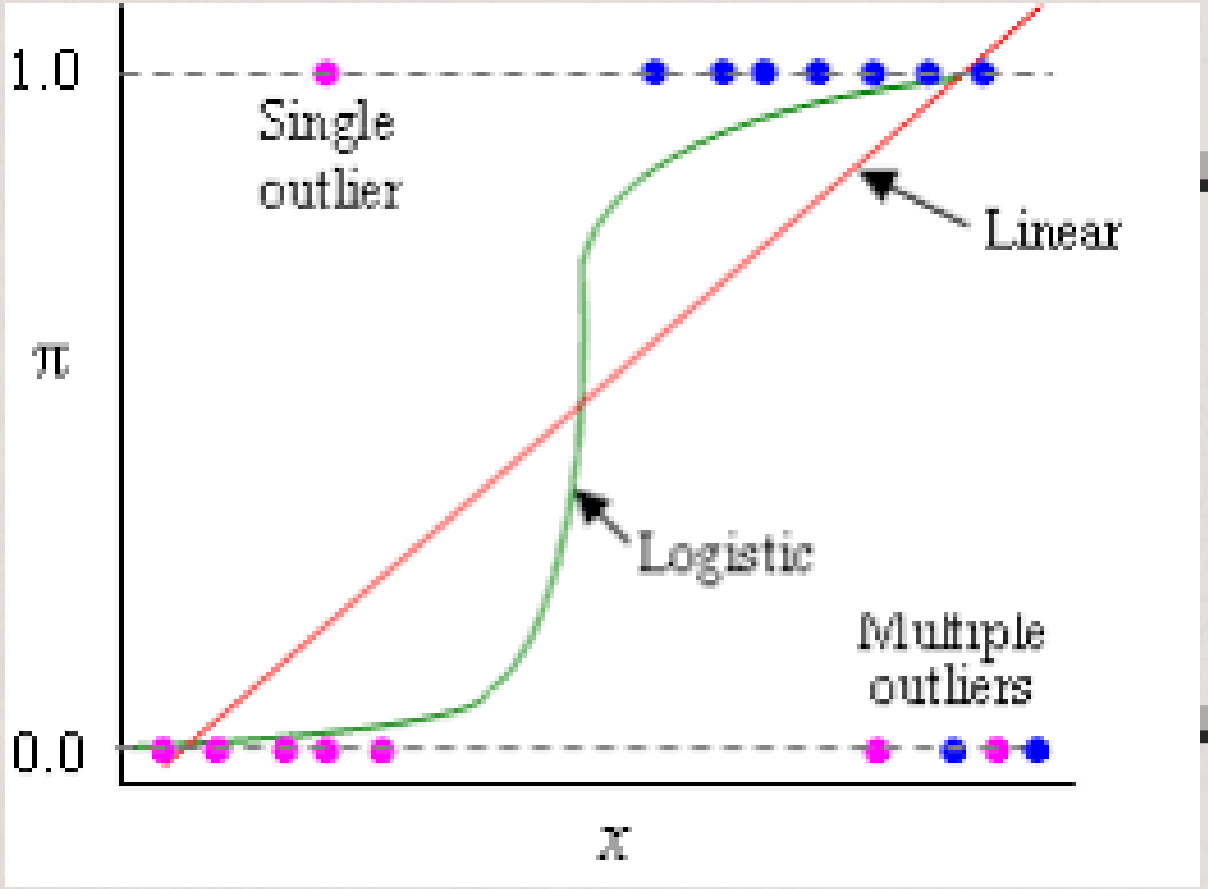
5. Hierarchical classification

Một cách đơn giản là ta tiền xử lý dữ liệu nhãn đầu vào. Như nếu dữ liệu nhãn là chó Phú Quốc, thì ta gán cho nó thêm nhãn chó, và động vật có vú vào. Sau đó ta chạy một mô hình logistic để phân loại. Để khắc phục được điều này, ta thêm vào một số điều kiện để thỏa mãn tính chất của cây.

Điều này cũng làm giảm đáng kể chi phí train của thuật toán. Tương tự ý tưởng đó, người ta cũng xấp xỉ hàm softmax bằng cây. Điều này đặc biệt hiệu quả khi số lượng lớp quá nhiều, điển hình như trong mô hình word2vec phải phân loại với số lượng lớp là số lượng từ trong ngôn ngữ. Cách xây dựng cây phổ biến ở mô hình softmax hình cây là sử dụng mã hóa Huffman.

5. Hierarchical classification





Đối với các bài toán **classification** có **dữ liệu nhiễu**, chúng ta sẽ sử dụng **Robust Logistic Regression**

Robust Logistic Regression

1. Sử dụng mixture model của các likelihood

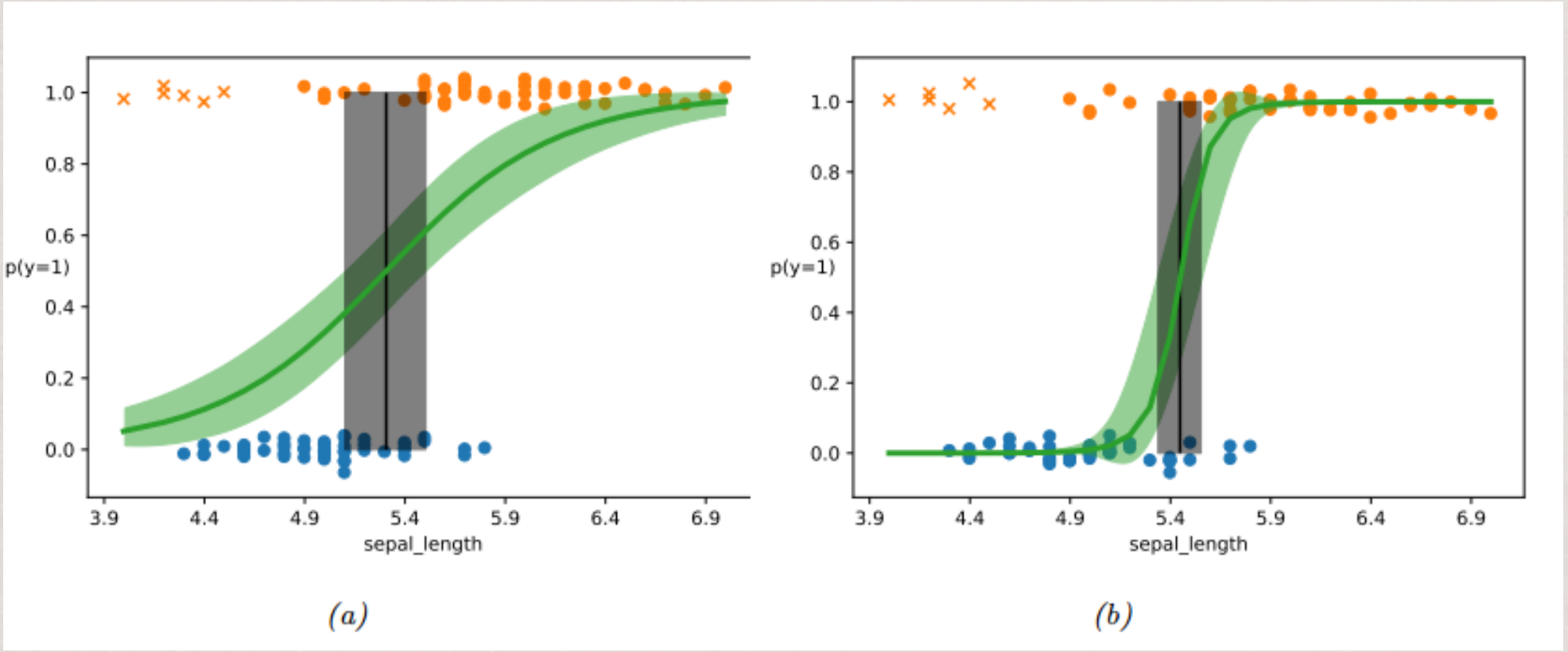
$\begin{cases} \pi & \text{là xác suất output label y được tạo ra ngẫu nhiên} \\ 1-\pi & \text{là xác suất output label y được tạo ra một cách thông thường} \end{cases}$

Ví dụ. Bài toán Binary Classification

$$p(y|\mathbf{x}) = \pi \text{Ber}(y|0.5) + (1-\pi)\text{Ber}\left(y|\sigma\left(\mathbf{w}^{\text{T}}\mathbf{x}\right)\right)$$

1. Sử dụng mixture model của các likelihood

Ví dụ.



Hình (a) sử dụng Bayesian logistic regression model. Hình (b) sử dụng Robust model

2. Bi-tempered loss

Đối với **điểm dữ liệu** được dán nhãn sai **ở xa đường phân cách**, ta dùng **tham số "tempered"** $0 \leq t_1 \leq 1$

Ta xét **entropy loss function** như sau: $\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}_c) = \mathbb{H}(\mathbf{y}, \hat{\boldsymbol{y}}_c) = \sum_c y_c \log \hat{y}_c$

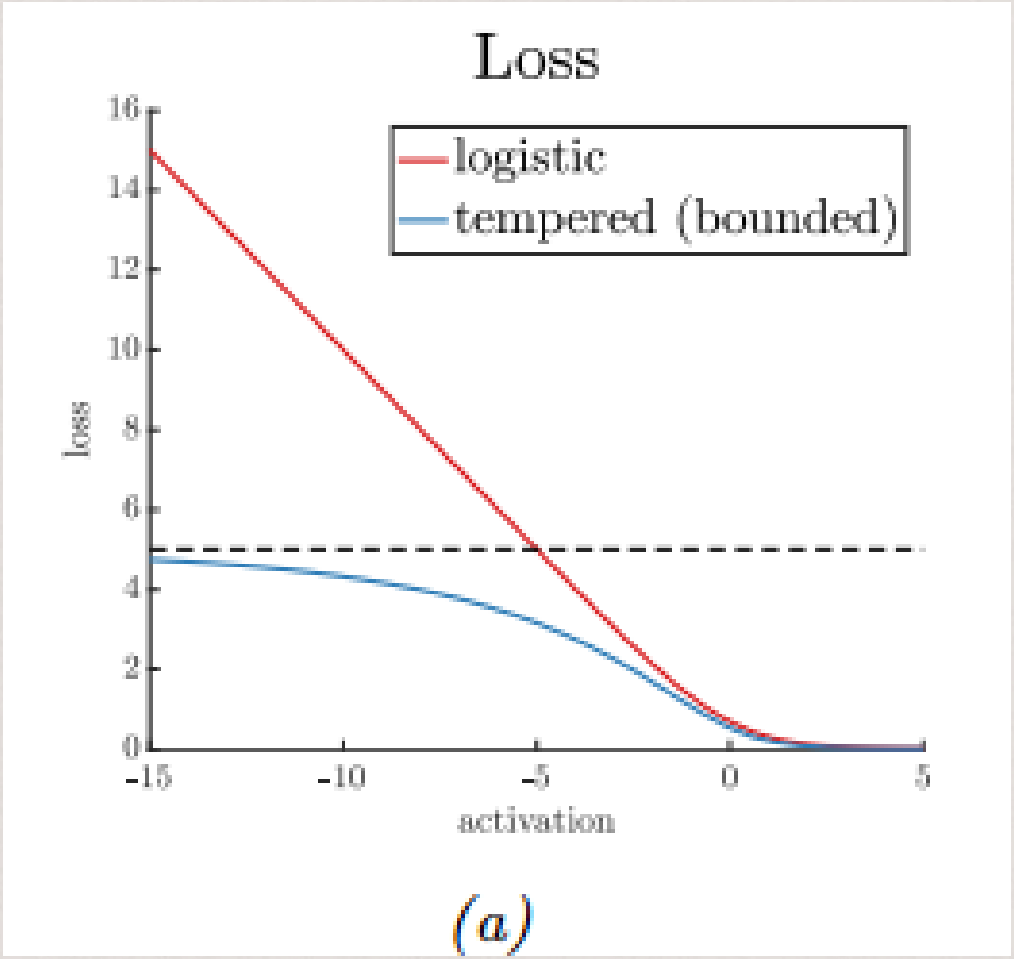
Khi đó, ta định nghĩa hàm $\log_t(x) \triangleq \begin{cases} \log(x), & \text{khi } x = 1 \\ \frac{1}{1-t}(x^{1-t} - 1), & \text{khi } x \neq 1 \end{cases}$

Ta sử dụng hàm loss mới được định nghĩa như sau:

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}_c) = \sum_c \left[y_c \left(\log_{t_1} y_c - \log_{t_1} \hat{y}_c \right) - \frac{1}{2 - t_1} \left(y_c^{2-t_1} - \hat{y}_c^{2-t_1} \right) \right]$$

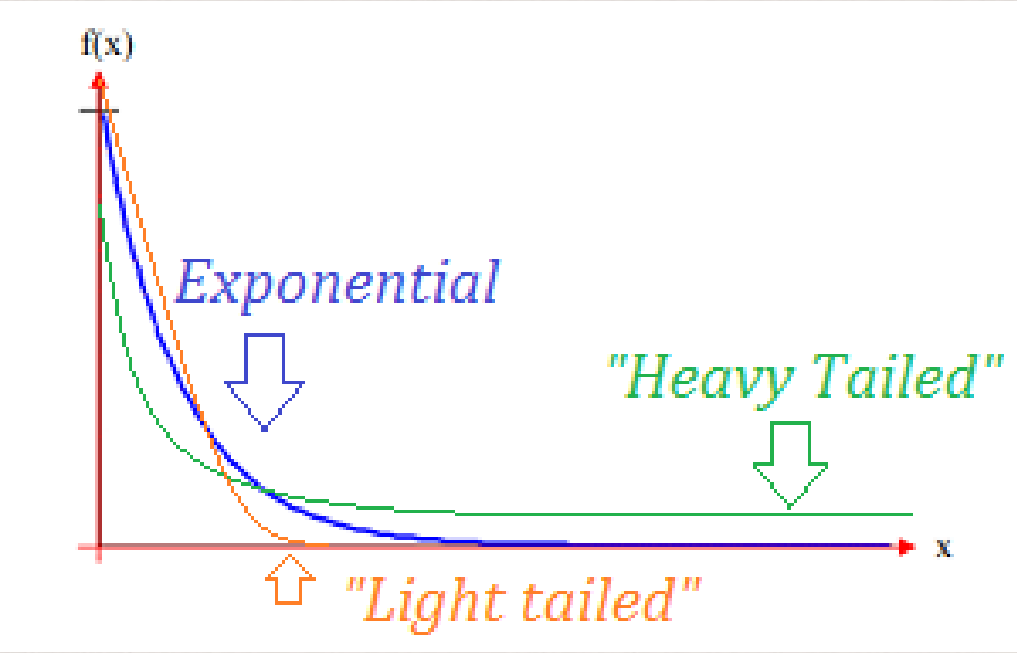
2. Bi-tempered loss

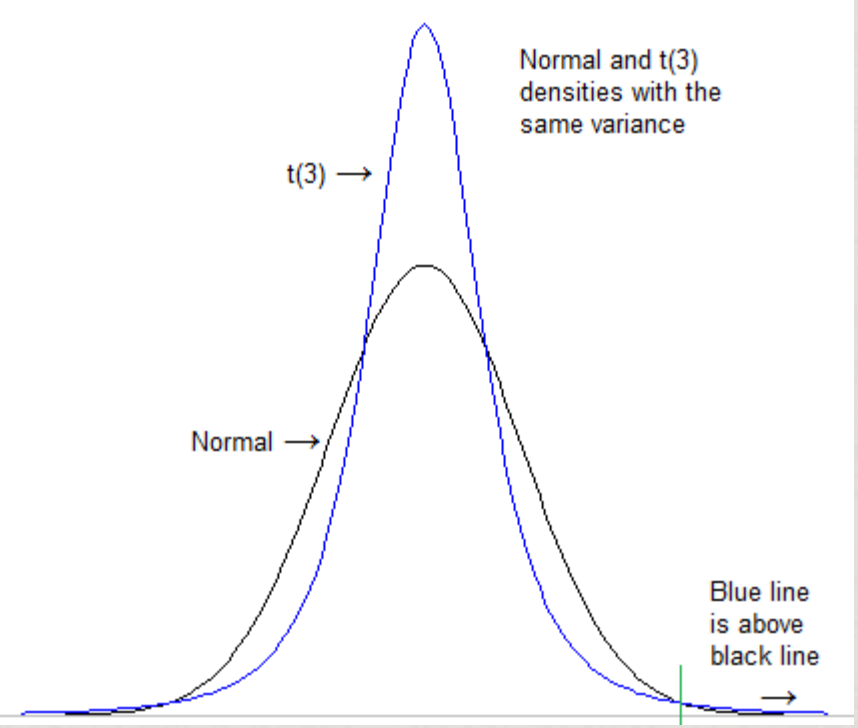
- Đối với **điểm dữ liệu** được dán nhãn sai **ở xa đường phân cách**



2. Bi-tempered loss

Đối với **điểm dữ liệu** được dán nhãn sai **ở gần đường phân cách**, cần dùng một hàm thay thế hàm softmax (ánh xạ từ $\mathbb{R}^C$ đến xác suất $[0,1]^C$) nó **heavy tails** hơn so với hàm softmax.





2. Bi-tempered loss

Đối với **điểm dữ liệu** được dán nhãn sai **ở gần đường phân cách**

Ta có hàm softmax tiêu chuẩn như sau:

$$\hat{y}_c = \frac{\exp(a_c)}{\sum_{c'} \exp(a_{c'})} = \exp\left[a_c - \log \sum_{c'} \exp(a_{c'})\right]$$

trong đó, $\mathbf{a} = \mathbf{W}\mathbf{x}$ là logit vector

Ta sử dụng tham số $t_2 \geq 1 \geq t_1$, sao cho

$$\exp_t(x) = \begin{cases} \exp(x), & \text{khi } x = 1 \\ (1 + (1-t)x)_+^{1/1-t} & \text{khi } x > 1 \end{cases}$$

2. Bi-tempered loss

Đối với **điểm dữ liệu** được dán nhãn sai **ở gần đường phân cách**

Ta sử dụng hàm softmax mới được định nghĩa như sau:

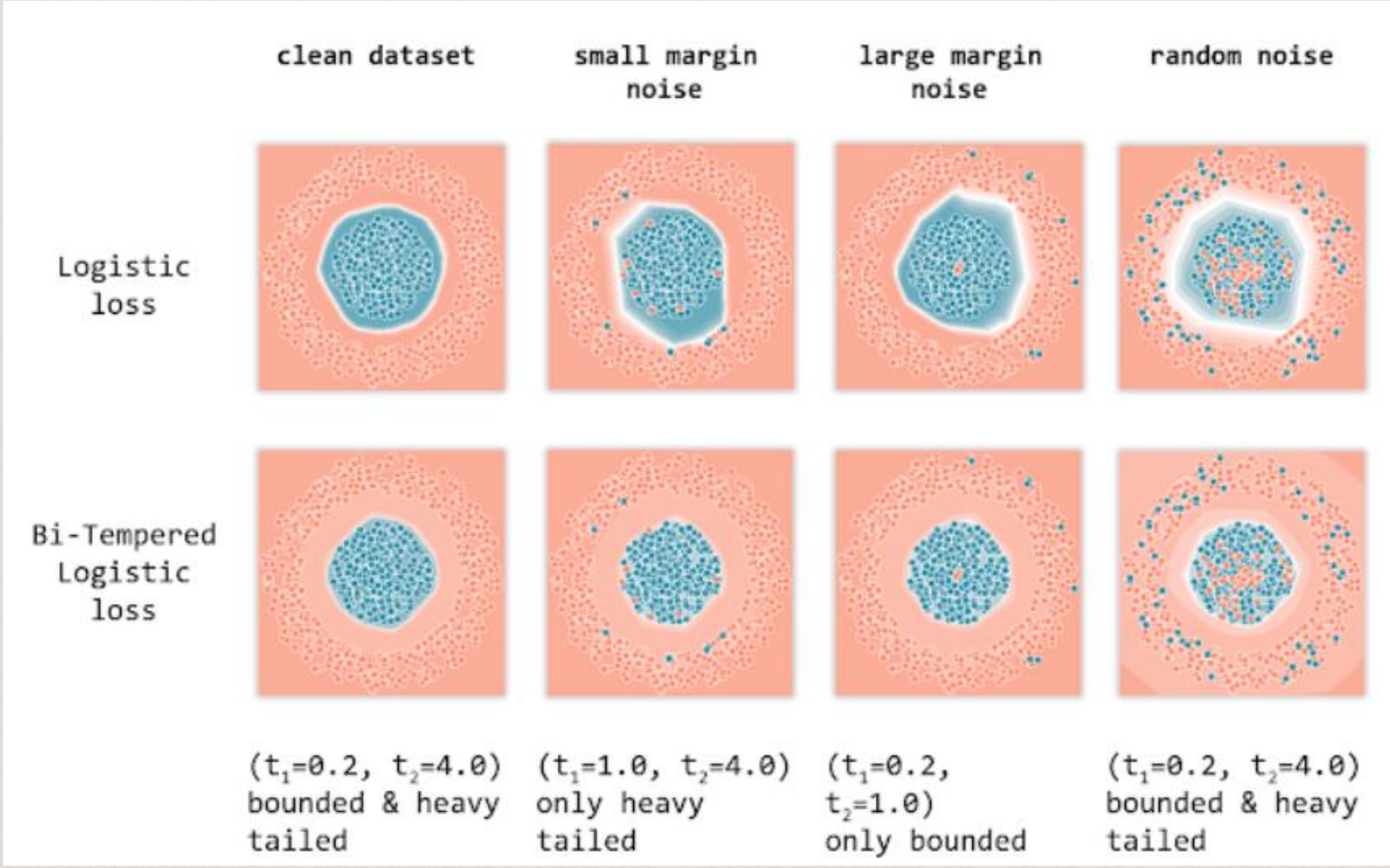
$$\hat{y}_c = \exp_{t_2}\left[a_c - \lambda_{t_2}(\mathbf{a})\right]$$

Vấn đề của việc này là tính $\lambda_{t_2}(\mathbf{a})$, biết nó thỏa mãn

$$\sum_c \exp_{t_2}\left[a_c - \lambda_{t_2}(\mathbf{a})\right] = 1$$

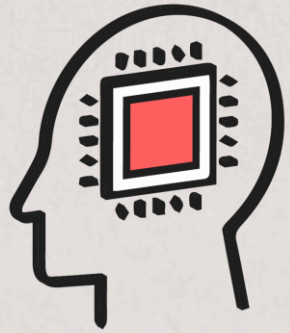
Ta sẽ tính λ bằng binary search hoặc vòng lặp

2. Bi-tempered loss



Bayesian Logistic Regression

Xác suất hậu nghiệm, phân phối dự đoán hậu nghiệm: không thể tính chính xác được ⇒ Sử dụng xấp xĩ



1. Xấp xĩ xác suất hậu nghiệm

Ta sử dụng xấp xĩ Laplace:

$$p(\mathbf{w}|\mathcal{D}) \approx \mathcal{N}(\hat{\mathbf{w}}, \mathbf{H}^{-1})$$

Trong đó, $\hat{\mathbf{w}} =$ MAP estimate

$\mathbf{H}^{-1}$ là ma trận Hesse nghịch đảo của $\log p(\hat{\mathbf{w}}|\mathcal{D})$

2. Xấp xĩ phân phối dự đoán hậu nghiệm

$$p(\mathbf{y}|\mathbf{x}, \mathcal{D}) = \int p(\mathbf{y}|\mathbf{x}, \mathbf{w}) p(\mathbf{w}|\mathcal{D}) d\mathbf{w}$$

Nếu coi xác suất hậu nghiệm bị suy biến $p(\mathbf{w}|\mathcal{D}) \approx \delta(\mathbf{w} - \hat{\mathbf{w}})$

Trong đó, $\hat{\mathbf{w}} =$ MAP estimate

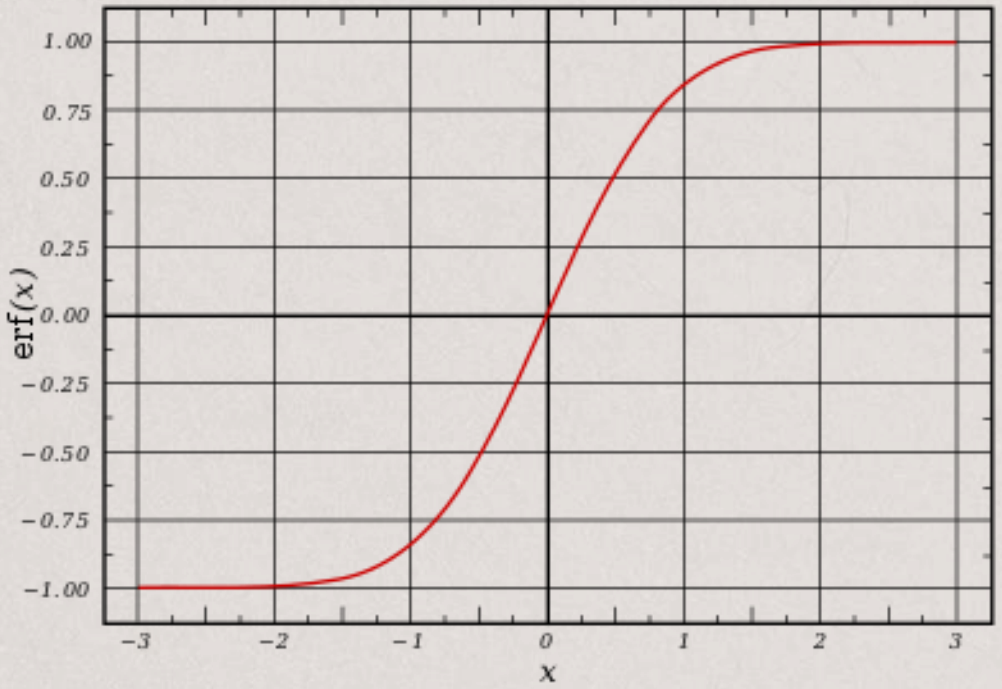
Khi đó phân phối dự đoán hậu nghiệm là

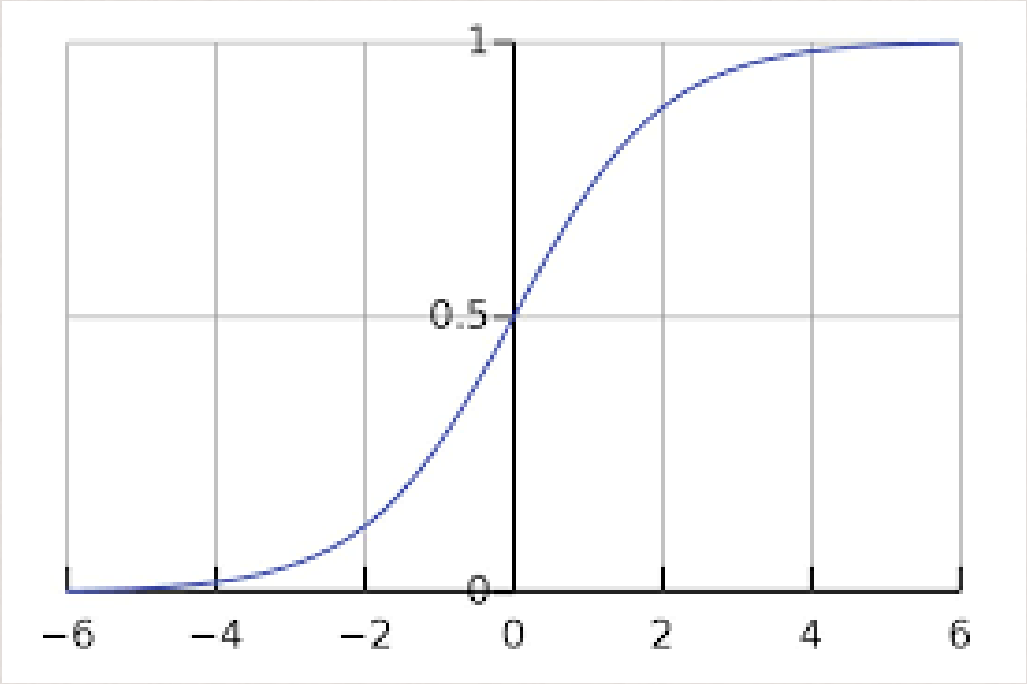
$$p(\mathbf{y}|\mathbf{x}, \mathcal{D}) = \int p(\mathbf{y}|\mathbf{x}, \mathbf{w}) \delta(\mathbf{w} - \hat{\mathbf{w}}) d\mathbf{w} \approx p(\mathbf{y}|\mathbf{x}, \hat{\mathbf{w}})$$

2. Xấp xĩ phân phối dự đoán hậu nghiệm

Vì $p(\mathbf{w}|\mathcal{D}) \approx \mathcal{N}(\hat{\mathbf{w}}, \mathbf{H}^{-1}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, nên bài toán sẽ đơn giản hơn rất nhiều. Ta sử dụng xấp xĩ Probit như sau

Ta thấy hàm Sigmoid $\sigma(x)$ có hình dáng tương tự như phân phối Gauss $\boldsymbol{\Phi}(x)$





$\sigma(x) \approx \boldsymbol{\Phi}(\lambda x)$ với $\lambda = \pi^2/8$

2. Xấp xĩ phân phối dự đoán hậu nghiệm

Ta có: $\int \mathbf{\Phi}(\lambda a)\mathcal{N}(a|m,v)da = \mathbf{\Phi}\left(\frac{m}{(\lambda^{-2}+v)^{1/2}}\right) = \mathbf{\Phi}\left(\frac{\lambda m}{(1+\lambda^2 v)^{1/2}}\right) \approx \sigma(\kappa(v)m)$

Nếu như $a = \mathbf{x}^{\mathrm{T}}\mathbf{w}$,

Với $\kappa(v) = \left(1 + \frac{\pi v}{8}\right)^{-1/2}$

$$m = \mathbb{E}[a] = \mathbf{x}^{\mathrm{T}}\boldsymbol{\mu},$$

$$v = \mathbb{V}[a] = \mathbb{V}[\mathbf{x}^{\mathrm{T}}\mathbf{w}] = \mathbf{x}^{\mathrm{T}}\mathbf{\Sigma}\mathbf{x}$$

$$p(y = 1|\mathbf{x}, \mathcal{D}) = \sigma(\kappa(v)m)$$

Trong trường hợp multiclass, sử dụng phương pháp Spiegelhalter

$$p(y = c|\mathbf{x}, \mathcal{D}) = \frac{\exp(\kappa(v_c)m_c)}{\sum_{c'}\exp(\kappa(v_{c'})m_{c'})}$$

Linear Regression

- Hàm xấp xỉ: $f_{\mathbf{w}}(\mathbf{x}) = \hat{y} = \mathbf{w}^{\mathrm{T}}\mathbf{x}$
- Hàm loss: $\mathcal{L}(y, \hat{y}) = \|y - \hat{y}\|_2^2$
- Regurlarization: $\ell(\mathbf{w}) = \|\mathbf{w}\|_2^2,\ \|\mathbf{w}\|_1, \ldots$

Logistic Regression

- Hàm xấp xỉ: $f_{\mathbf{w}}(\mathbf{x}) = \hat{y} = \sigma(\mathbf{w}^{\mathrm{T}}\mathbf{x})$
- Hàm loss: $\mathcal{L}(y, \hat{y}) = -\hat{y} \log\ y - (1 - \hat{y}) \log(1 - y)$
- Regurlarization: $\ell(\mathbf{w}) = \|\mathbf{w}\|_2^2,\ \|\mathbf{w}\|_1, \ldots$

Summaries

Ưu điểm

- Các trọng số được minh bạch
- Dễ thực hiện, sử dụng

Khuyết điểm

- Đôi khi không trực quan
- Non-linear cần phải làm thủ công
- Khả năng dự đoán không tốt

Linear Regression

Ưu điểm

- Các trọng số được minh bạch
- Dễ thực hiện, sử dụng

Khuyết điểm

- Biểu diễn bị hạn chế
- Bị chia cắt hoàn toàn

Logistic Regression

Support Vector Machines

- SVMs được xây dựng từ Linear models với Hinge loss
- SVMs là kết quả của Maximal-Margin Classifier
- SVMs xây dựng bài toán dual từ primal
- SVMs có thể giải quyết được bài toán nonlinear với Kernal tricks

- LR và SVMs được xây dựng từ linear models có liên kết với nhau khá chặt chẽ
- LR cho chúng ta xác suất có hiệu chuẩn
- LR có thể sử dụng tốt với Bayes model
- SVMs có dual form khá đẹp, giải quyết được bài toán nonlinear nhờ vào Kernal trick

LR và SVMs giải quyết những bài toán nào?

Tùy vào bài toán mình giải quyết

Nguồn tham khảo

- Github: Philipp Biedenkopf
- Github: Aakash Paul
- Website: Scikit-learn web - Linear Regression
- Sách: Probabilistic Machine Learning An Introduction-booktree
- Sách: Pattern Recognition and Machine Learning
- Paper: Gaussian process regression with Student-t likelihood
- Paper: Robust mixture regression model fitting by Laplace distribution
- Paper: Random Sample Consensus: A Paradigm for Model Fitting with Apphcatlons to Image Analysis and Automated Cartography

THANK YOU

Open with Colab