

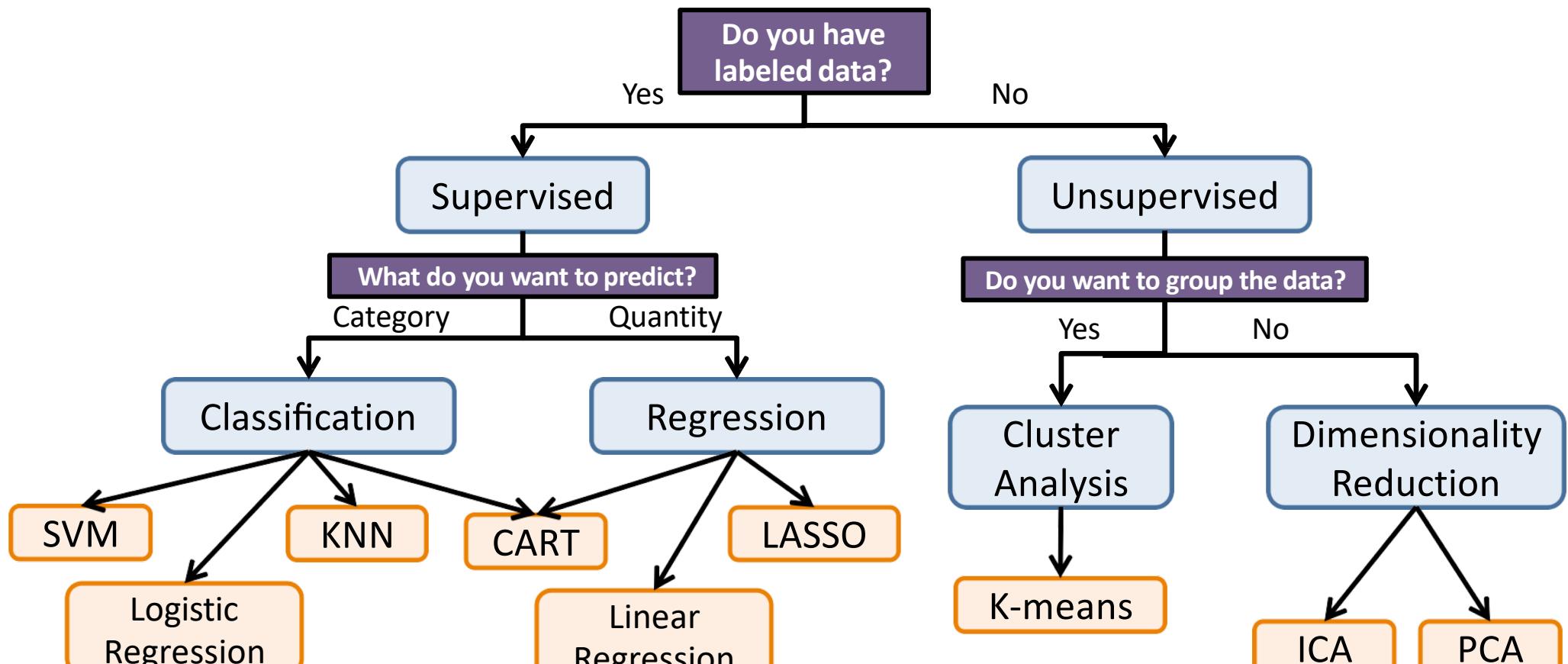
Hồi quy tuyến tính

Nguyễn Thanh Tùng
Khoa Công nghệ thông tin – Đại học Thủ Dầu Một
tungnt@tlu.edu.vn

Website môn học: <http://lms.tlu.edu.vn/course/view.php?id=772>



Các dạng giải thuật học máy



Giới thiệu về Học có giám sát



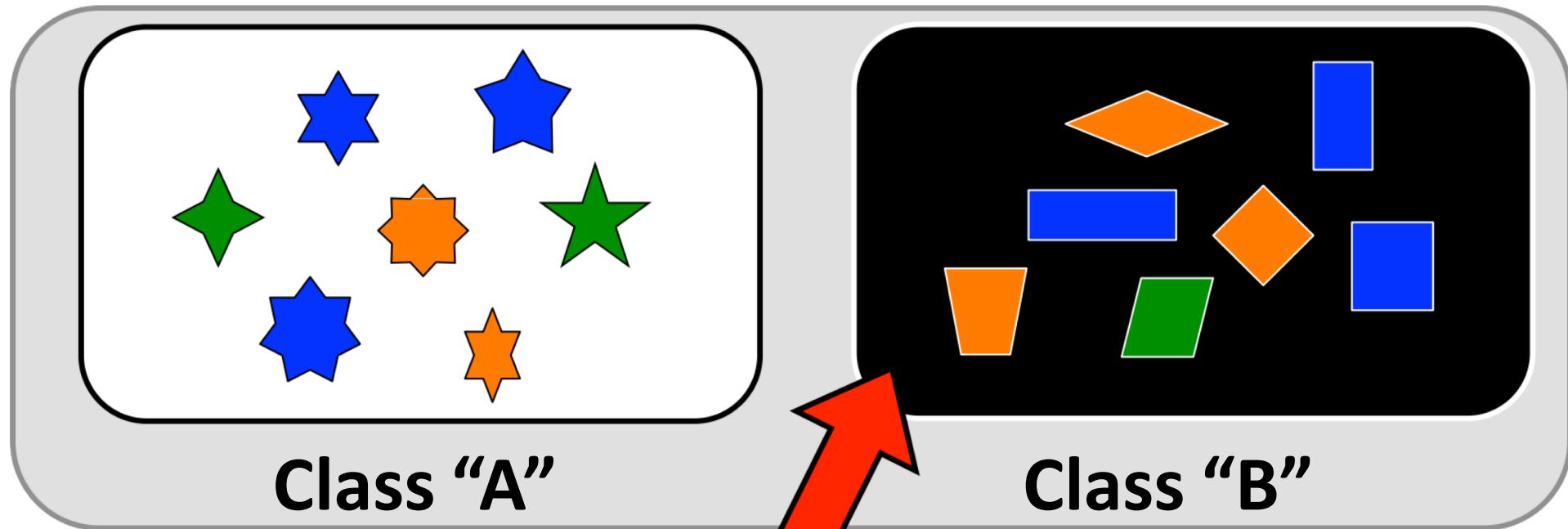
Học giám sát

- Xét:
$$Y = f(X) + \epsilon$$
- Các phương pháp học giám sát:
 - Học bởi các ví dụ (quan sát)-“Learn by example”
 - Xây dựng mô hình \hat{f} sử dụng tập các quan sát đã được gắn nhãn

$$\left(X^{(1)}, Y^{(1)} \right), \dots, \left(X^{(n)}, Y^{(n)} \right)$$



Dữ liệu học



Dữ liệu học

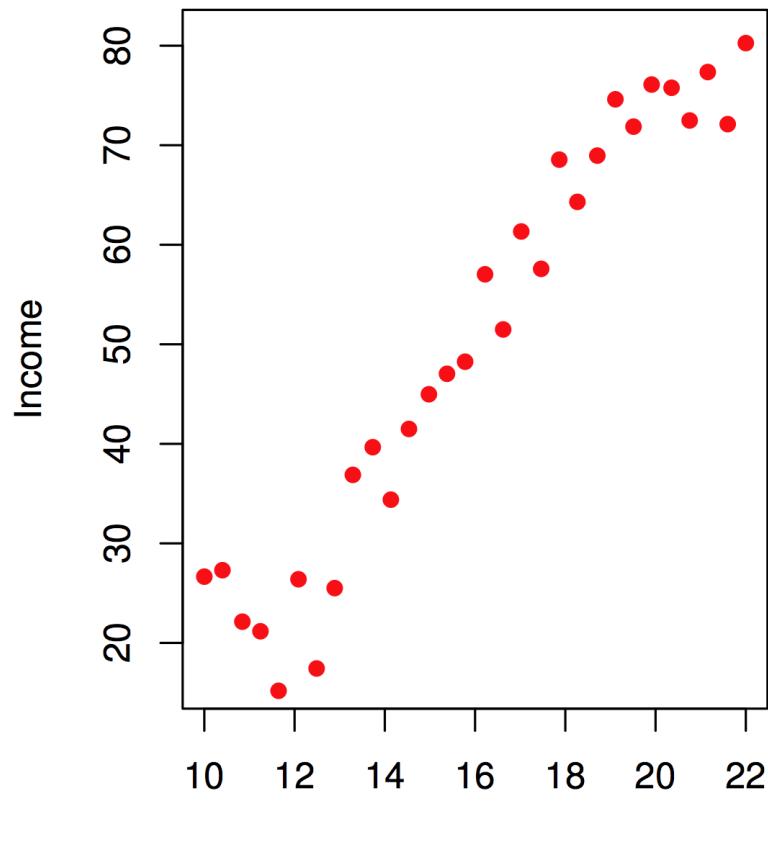


Figure 2.2 , ISL 2013



Years of Education

CSE 445: Học máy, K60 | Học kỳ 1, 2021-2022

Học có giám sát

- Giải thuật học có giám sát
 - Lấy hàm ước lượng “tốt nhất” \hat{f} trong tập các hàm
- Ví dụ: Hồi quy tuyến tính
 - Chọn 1 ước lượng tốt nhất từ *dữ liệu học* trong tập các hàm tuyến tính

$$f(X) = \beta_0 + \beta_1 X_1 + \dots + \beta_d X_d$$



Độ chính xác của mô hình



Đo hiệu năng bài toán hồi quy

- Hàm tổn thất (Loss function): loại hàm dùng để đo lường sai số của mô hình
- Vd: Sai số bình phương trung bình (Mean squared error - MSE)
 - Độ đo thông dụng dùng để tính độ chính xác bài toán hồi quy

$$MSE = \frac{1}{n} \sum_{i=1}^n \left(\hat{y}^{(i)} - y^{(i)} \right)^2$$

- Tập trung đo các sai số lớn hơn là các sai số nhỏ



Đo hiệu năng bài toán hồi quy

- Mục tiêu: xây dựng mô hình khái quát hóa (*generalizes*)
 - Ta muốn cực tiểu hóa lỗi trên dữ liệu chưa biết, không phải trên dữ liệu học.
 - Vd: Dự đoán giá cổ phiếu *trong tương lai* vs. giá cổ phiếu trong quá khứ
- Chúng ta muốn cực tiểu hóa tổn thất kỳ vọng (*expected loss*)
 - Vấn đề: Ta không thể cực tiểu lỗi trên dữ liệu huấn luyện.

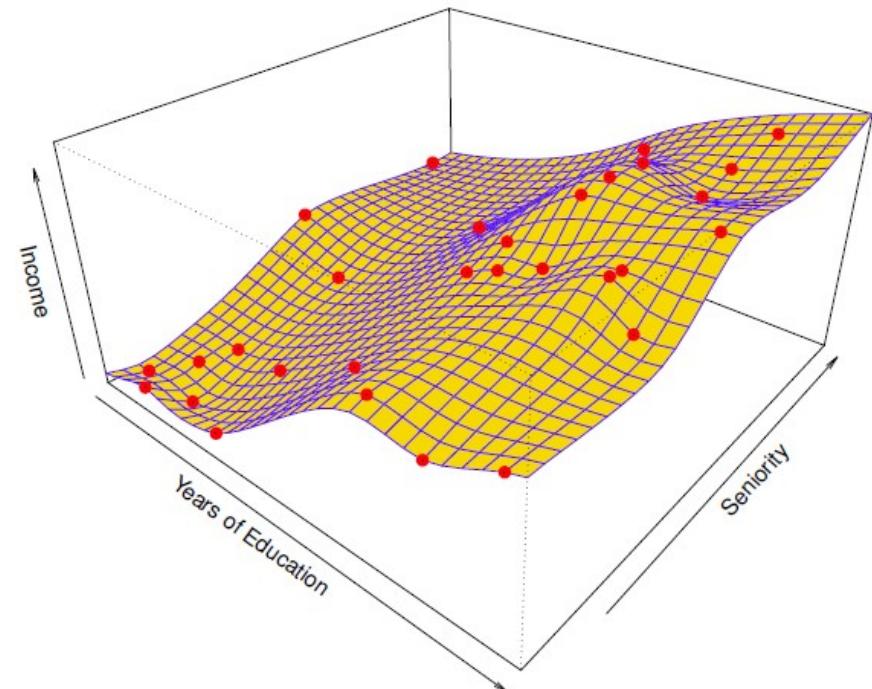
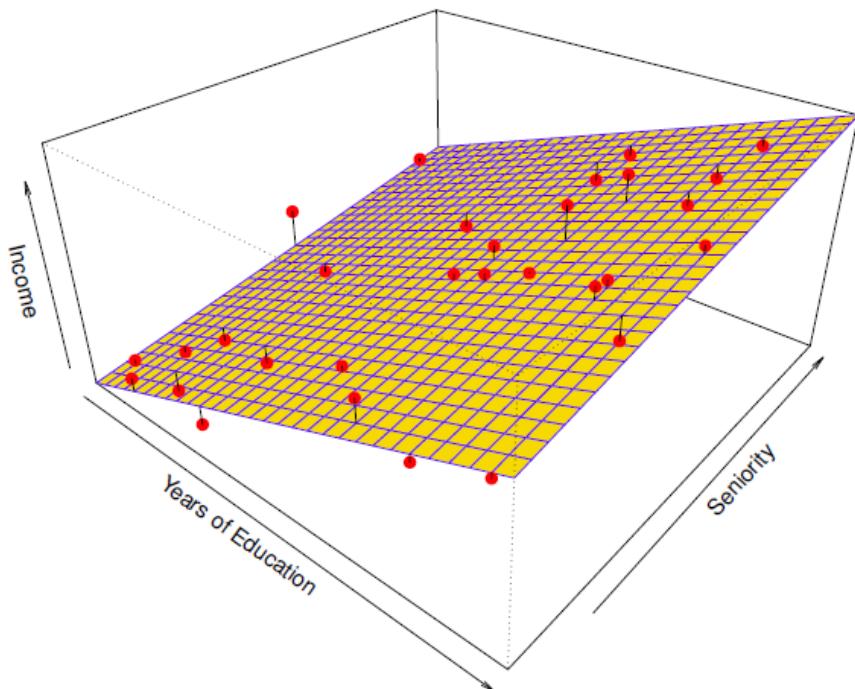


Vấn đề: Overfitting

- *Quá khớp (Overfitting)*: Học sự biến thiên ngẫu nhiên trong dữ liệu hơn là xu hướng cơ bản
- Đặc điểm của overfitting:
 - Mô hình có hiệu năng cao trên dữ liệu học nhưng kém trên tập dữ liệu thử nghiệm.



Vấn đề: Overfitting



Figures 2.4 and 2.6 , ISL 2013



Đánh giá hiệu năng

- Lỗi huấn luyện và lỗi kiểm thử thể hiện khác nhau
 - Tính linh hoạt của mô hình tăng lên...
 - *Lỗi huấn luyện* giảm
 - *Lỗi kiểm thử ban đầu* giảm,
Nhưng sau đó tăng lên vì overfitting → “U-shaped” lỗi kiểm thử dạng chữ U.



Đánh giá hiệu năng

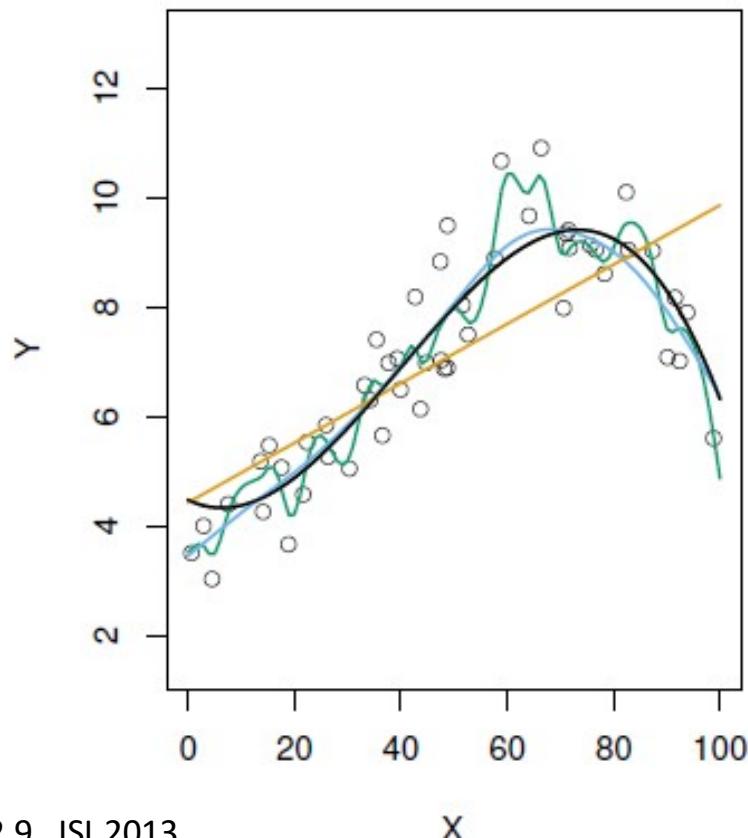
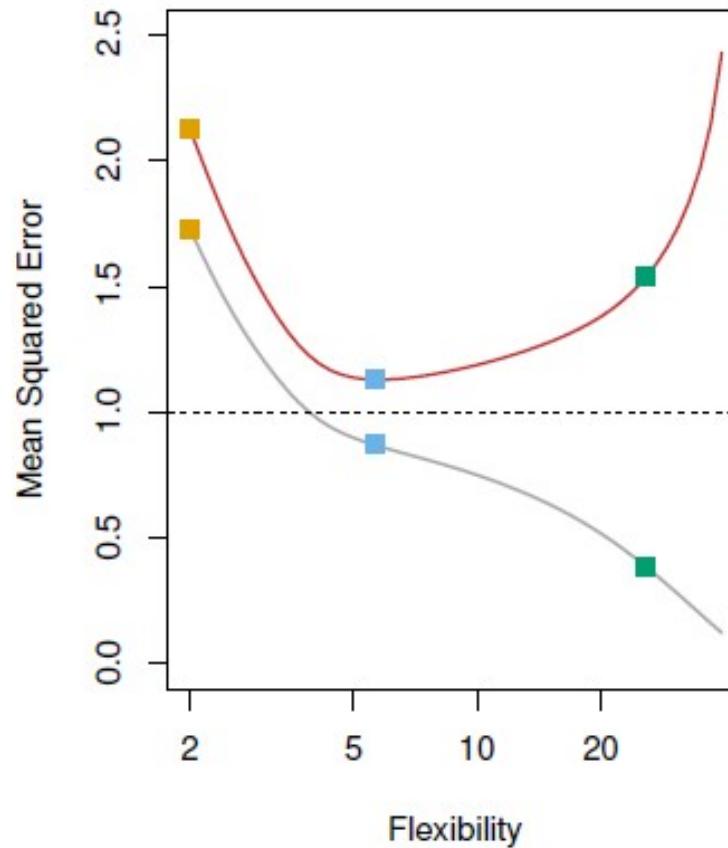


Figure 2.9 , ISL 2013



Đánh giá hiệu năng

- Làm sao để ước lượng lỗi kiểm thử để tìm một mô hình tốt?
- *Kỹ thuật kiểm tra chéo (Cross-validation):* một tập các kỹ thuật nhằm sử dụng dữ liệu huấn luyện để ước lượng lỗi tổng quát (generalization error)



Dữ liệu

- *Dữ liệu huấn luyện (Training data)*
 - Tập các quan sát (bản ghi) được sử dụng để xây dựng (học) mô hình.
- *Dữ liệu kiểm chứng (Validation data)*
 - Tập các quan sát dùng để ước lượng lỗi nhằm tìm tham số hoặc lựa chọn mô hình.
- *Dữ liệu kiểm thử (Test data)*
 - Tập các quan sát dùng để đánh giá hiệu năng trên dữ liệu chưa biết (unseen) trong tương lai.
 - Dữ liệu này không sử dụng cho giải thuật học máy trong quá trình xây dựng mô hình.



Trade-off: Độ lệch vs. Phương sai

- Lỗi kiểm thử đường cong hình chữ U (U-shaped) xảy ra dựa trên 2 đặc điểm của mô hình học máy:

$$\mathbb{E} [\text{test error}] = \text{var}(\hat{f}) + \text{bias}(\hat{f})^2 + \text{var}(\epsilon)$$

- $\text{var}(\hat{f})$: *Phương sai (variance)* của hàm ước lượng
- $\text{bias}(\hat{f})$: *Độ chêch/sai lệch (bias)* của hàm ước lượng

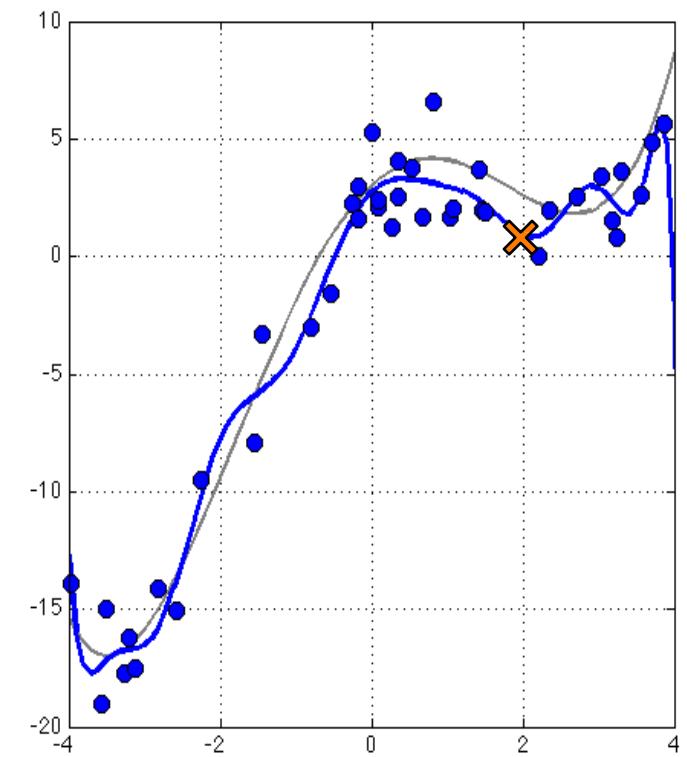
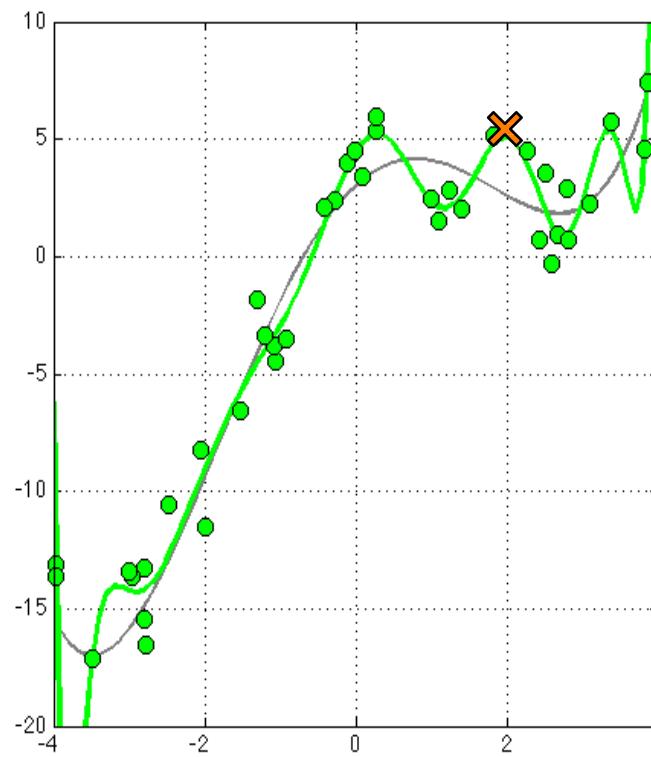
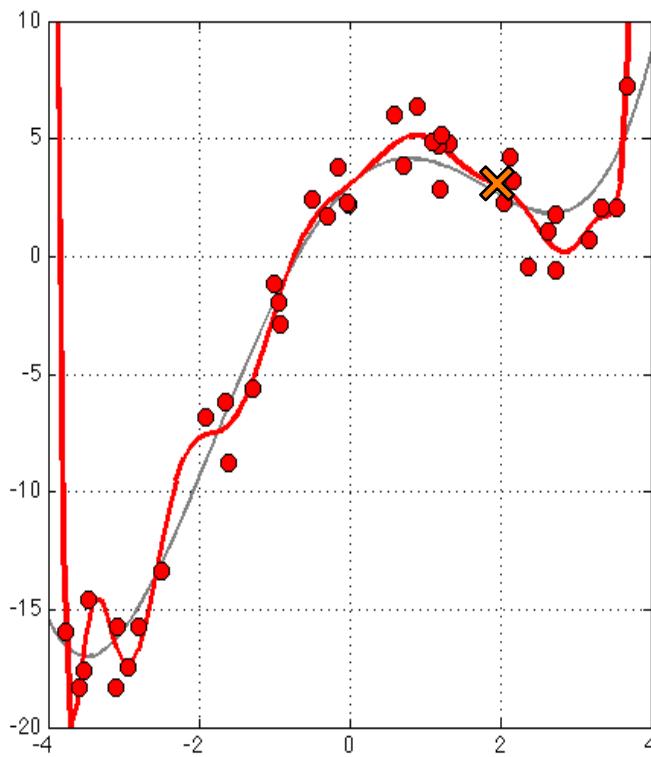


Trade-off: Độ lệch vs. Phương sai

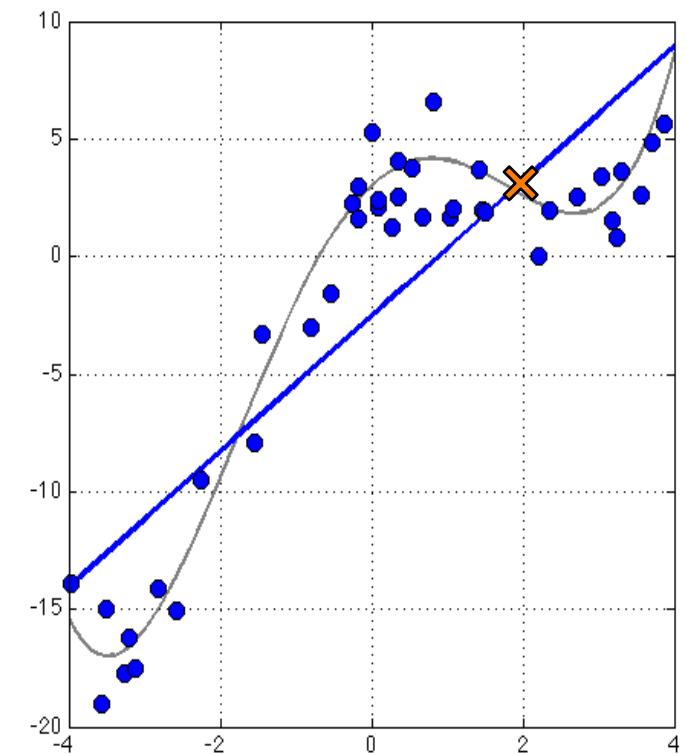
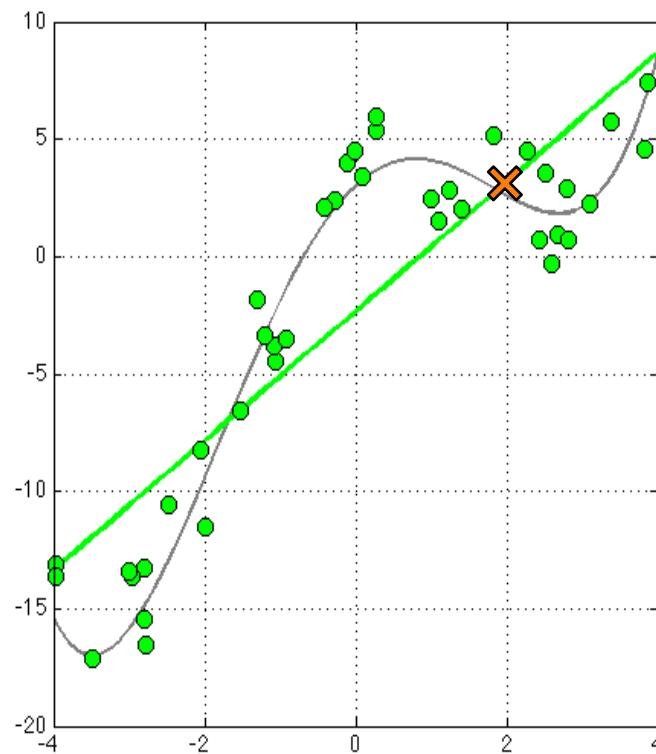
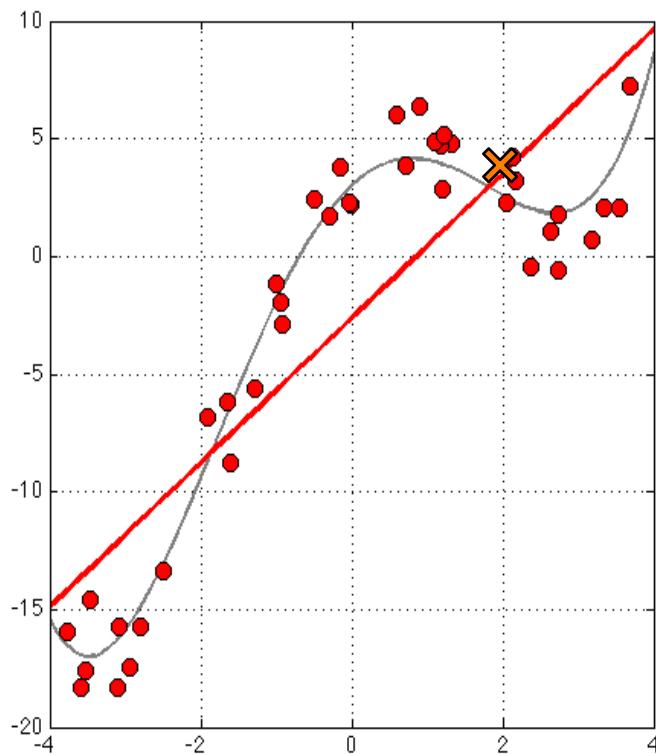
- *Phương sai của hàm ước lượng*
 - Chênh lệch giữa kết quả áp dụng mô hình với các quan sát đầu vào khác nhau.
- Phương sai cao: các thay đổi nhỏ trong tập huấn luyện
→ Các thay đổi lớn trong hàm ước lượng thống kê.
 - Các phương pháp càng linh hoạt → Phương sai càng lớn.



Trade-off: Độ lệch vs. Phương sai



Trade-off: Độ lệch vs. Phương sai



Trade-off: Độ lệch vs. Phương sai

- *Độ lệch (bias) của hàm ước lượng*
 - Bias là độ sai lệch giữa kết quả dự đoán của mô hình và thực tế, sai số xấp xỉ một hàm khi áp dụng một mô hình đơn giản.
 - Vd: Hồi quy tuyến tính giả định các biến phải quan hệ tuyến tính.
 - Lỗi bias xuất hiện khi hệ thống là phi tuyến.
 - Các phương pháp càng linh hoạt → bias nhỏ.

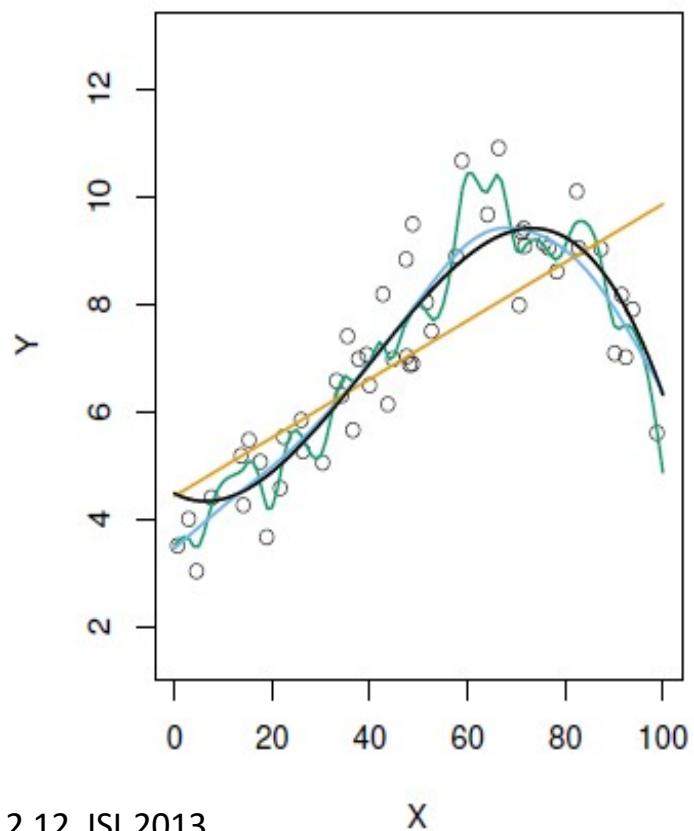


Trade-off: Độ lệch vs. Phương sai

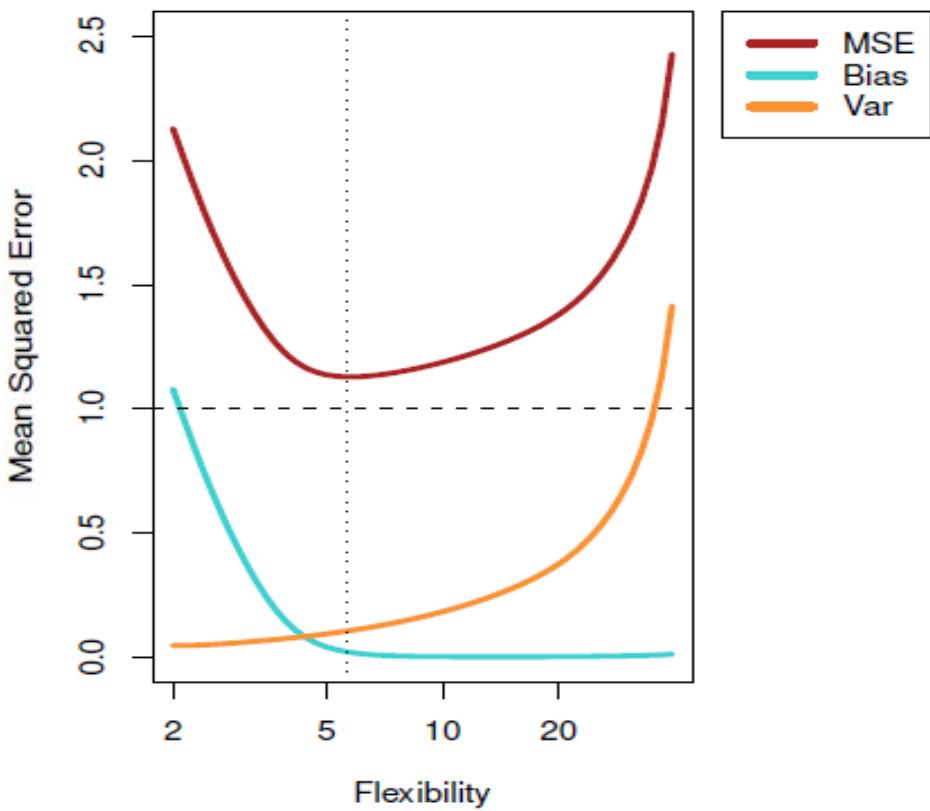
- Phương sai thấp và bias thấp → Lỗi kiểm thử cũng thấp.
- Càng linh hoạt (phức tạp) → Phương sai tăng, bias giảm.
- Lỗi kiểm thử đường cong hình chữ U (U-shaped):
 - Ban đầu độ linh hoạt mô hình tăng, ta thấy bias giảm nhanh hơn tăng phương sai → lỗi kiểm thử MSE giảm.
 - Độ linh hoạt của mô hình có ảnh hưởng nhỏ hơn đến việc giảm bias, tuy nhiên khi tăng độ linh hoạt nó ảnh hưởng lớn đến phương sai → lỗi kiểm thử MSE tăng.



Trade-off: Độ lệch vs. Phương sai



Figures 2.9, 2.12, ISL 2013



Trade-off: Độ lệch vs. Phương sai

- Phương pháp linh hoạt (phức tạp)
 - Có thể xấp xỉ sát hàm ước lượng thống kê (bias thấp),
 - Tuy nhiên các lỗi/rủi ro của mô hình học lại quá phụ thuộc vào dữ liệu huấn luyện (phương sai cao)
- Phương pháp đơn giản hơn
 - Có thể xấp xỉ hàm ước lượng với độ chính xác không cao (bias cao),
 - Tuy nhiên chúng ít phụ thuộc vào dữ liệu huấn luyện (phương sai thấp)
- Tradeloff
 - Dễ đạt được phương sai thấp/bias cao hoặc phương sai cao/bias thấp,
 - Tuy nhiên rất khó để đạt được cả phương sai và bias cùng thấp



Hồi quy: Hồi quy tuyến tính



Hồi quy tuyến tính

- *Hồi quy tuyến tính*: là phương pháp học máy có giám sát đơn giản, được sử dụng để dự đoán giá trị biến đầu ra dạng số (định lượng)
 - Nhiều phương pháp học máy là dạng tổng quát hóa của hồi quy tuyến tính
 - Là ví dụ để minh họa các khái niệm quan trọng trong bài toán học máy có giám sát



Hồi quy tuyến tính

- Tại sao dùng hồi quy tuyến tính?
 - Mỗi quan hệ tuyến tính: là sự biến đổi tuân theo quy luật hàm bậc nhất
 - Tìm một mô hình (phương trình) để mô tả một mối liên quan giữa X và Y
 - Ta có thể biến đổi các biến đầu vào để tạo ra mối quan hệ tuyến tính
 - Diễn giải các mối quan hệ giữa biến đầu vào và đầu ra - sử dụng cho bài toán suy diễn



Hồi quy tuyến tính đơn giản

- Biến đầu ra Y và biến đầu vào X có mối quan hệ tuyến tính giữa X và Y như sau:

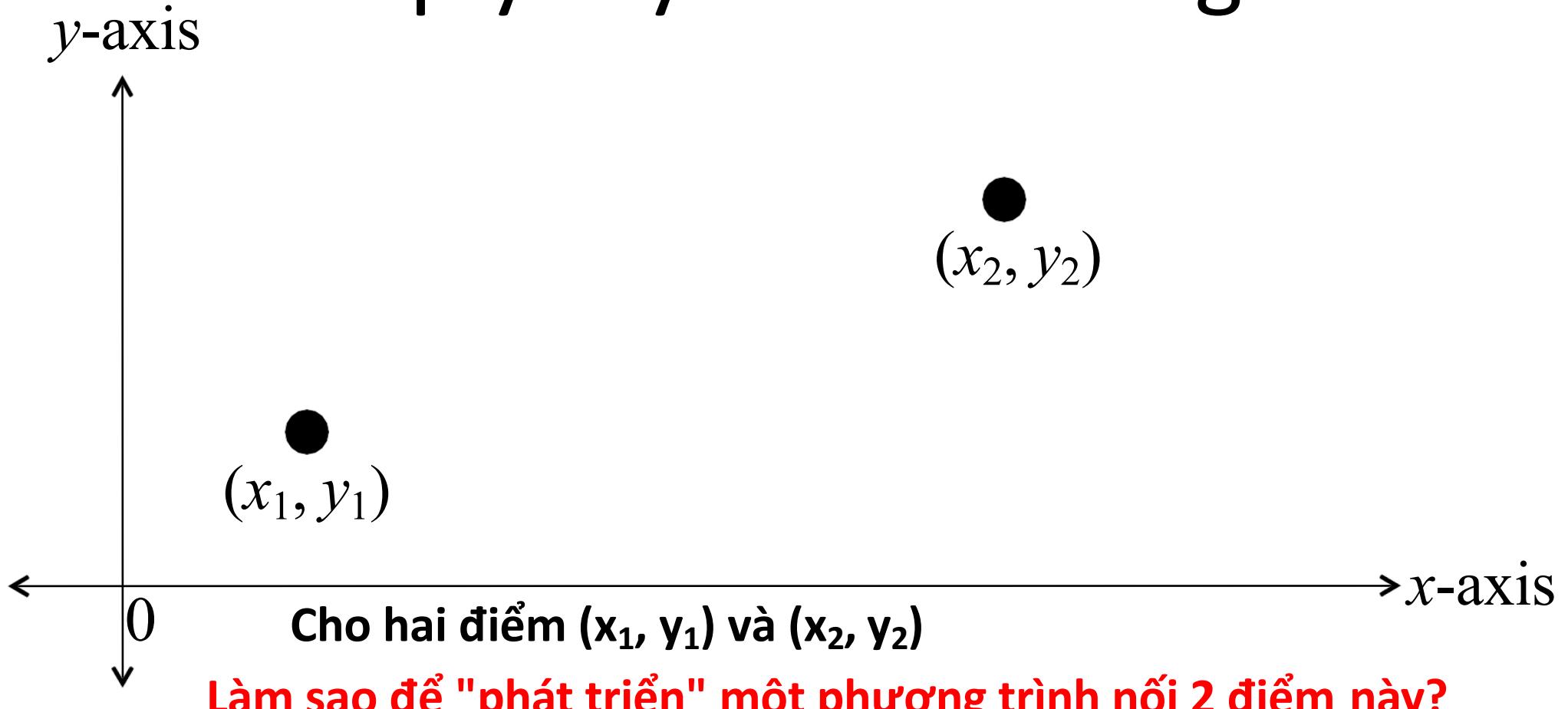
$$Y = \beta_0 + \beta_1 X + \epsilon$$

- Các tham số của mô hình:

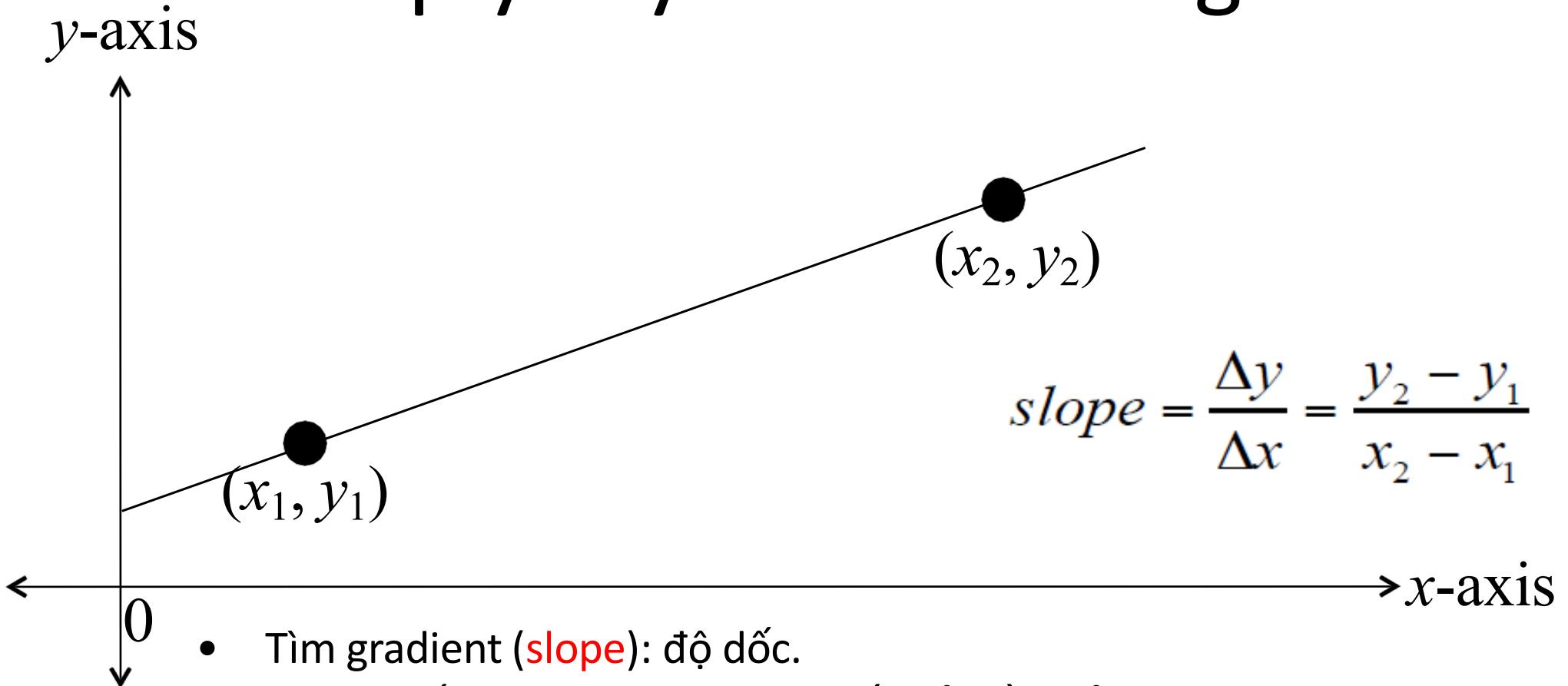
β_0 intercept hệ số chặn (khi các $x_i=0$)
 β_1 slope độ dốc



Hồi quy tuyến tính đơn giản



Hồi quy tuyến tính đơn giản



Hồi quy tuyến tính đơn giản

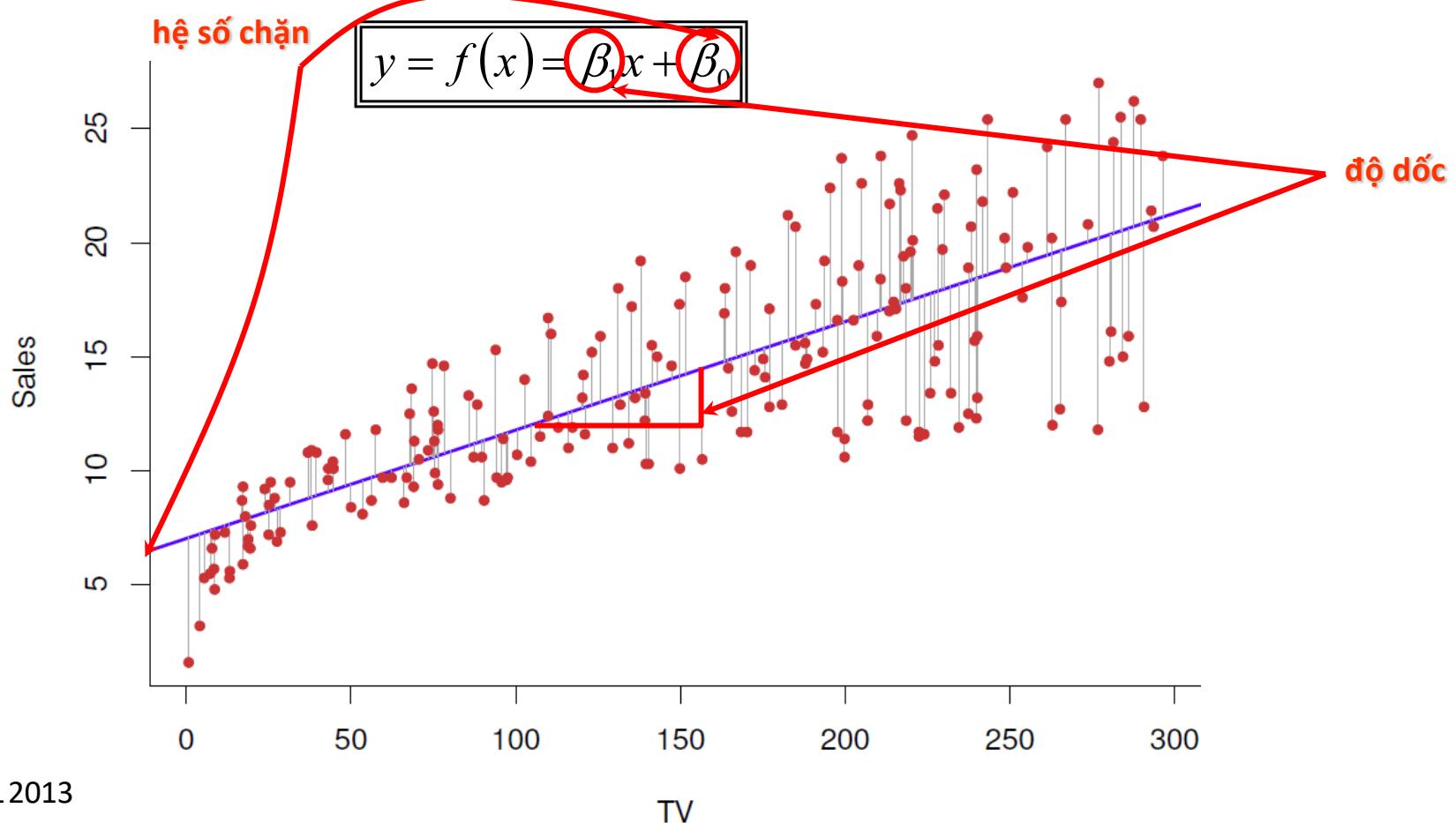


Figure 3.1 , ISL 2013



Hồi quy tuyến tính đơn giản

- β_0 và β_1 chưa biết \rightarrow Ta ước tính giá trị của chúng từ dữ liệu đầu vào

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X$$

- Lấy $\hat{\beta}_0, \hat{\beta}_1$ sao cho mô hình đạt “xấp xỉ tốt nhất” (“good fit”) đối với tập huấn luyện

$$Y^{(i)} \approx \hat{\beta}_0 + \hat{\beta}_1 X^{(i)}, \quad i = 1, \dots, n$$



Các giả định

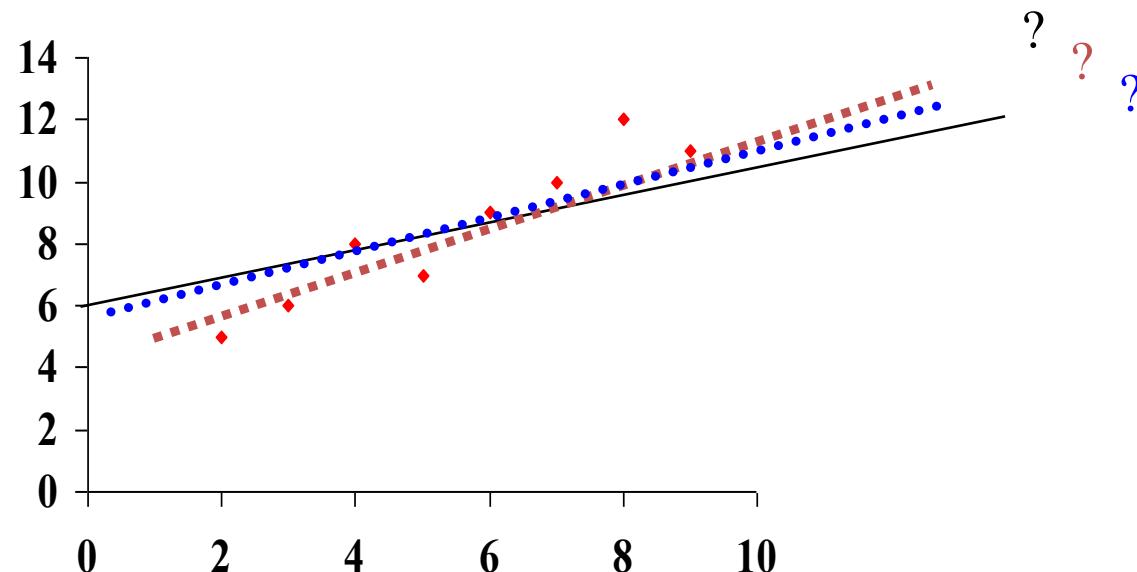
- Mỗi liên quan giữa X và Y là tuyến tính (linear) về *tham số*
- X không có sai số ngẫu nhiên
- Giá trị của Y độc lập với nhau (vd, Y_1 không liên quan với Y_2) ;
- Sai số ngẫu nhiên (ε): phân bố chuẩn, trung bình 0, phương sai bất biến

$$\varepsilon \sim N(0, \sigma^2)$$



Đường thẳng phù hợp nhất

Cho tập dữ liệu đầu vào, ta cần tìm cách tính toán các tham số của phương trình đường thẳng



Bình phương nhỏ nhất

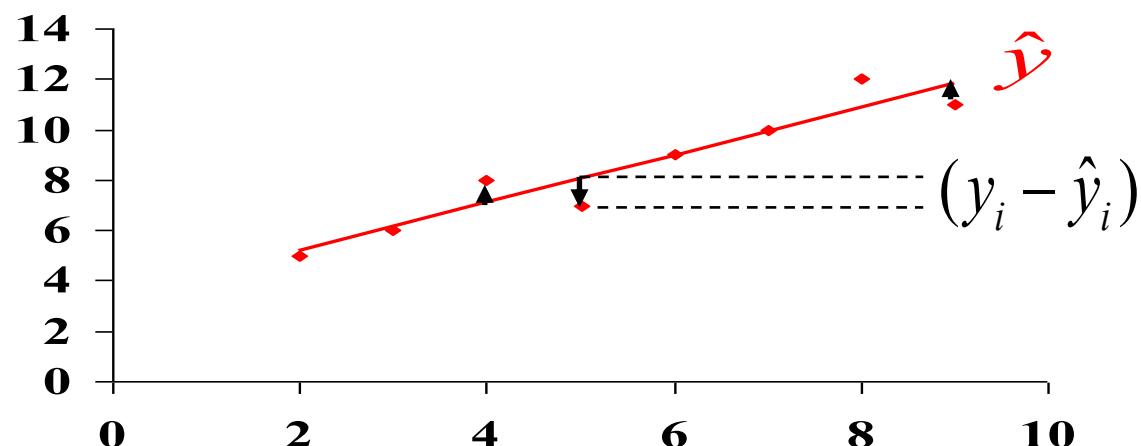
- Thông thường, để đánh giá độ phù hợp của mô hình từ dữ liệu quan sát ta sử dụng phương pháp *bình phương nhỏ nhất* (*least squares*)
- Lỗi bình phương trung bình (Mean squared error):

$$MSE = \frac{1}{n} \sum_{i=1}^n \left(Y^{(i)} - \hat{Y}^{(i)} \right)^2$$



Đường thẳng phù hợp nhất

Rất hiếm để có 1 đường thẳng khớp chính xác với dữ liệu, do vậy luôn tồn tại lỗi gắn liền với đường thẳng
Đường thẳng phù hợp nhất là đường giảm thiểu độ dao động của các lỗi này



Phần dư (lỗi)

Biểu thức ($y_i - \hat{y}$) được gọi là lỗi hoặc *phần dư*

$$\varepsilon_i = (y_i - \hat{y})$$

Đường thẳng phù hợp nhất tìm thấy khi tổng bình phương lỗi là nhỏ nhất

$$SSE = \sum_{i=1}^n (y_i - \hat{y})^2$$



Ước lượng tham số

- Các ước số $\hat{\beta}_0, \hat{\beta}_1$ tính được bằng cách cực tiểu hóa MSE

$$\min_{(\hat{\beta}_0, \hat{\beta}_1)} \left[\frac{1}{n} \sum_{i=1}^n \left(Y^{(i)} - (\hat{\beta}_0 + \hat{\beta}_1 X^{(i)}) \right)^2 \right]$$

- Hệ số chẵn của đường thẳng $\hat{\beta}_1 = \frac{SS_{xy}}{SS_x}$

trong đó: $SS_{xy} = \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})$ và $SS_x = \sum_{i=1}^n (x_i - \bar{x})^2$



Ước lượng tham số

Hệ số chặn của đường thẳng

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

trong đó

$$\bar{y} = \frac{\sum_{i=1}^n y_i}{n} \quad \bar{x} = \frac{\sum_{i=1}^n x_i}{n}$$



Hồi quy tuyến tính đơn giản

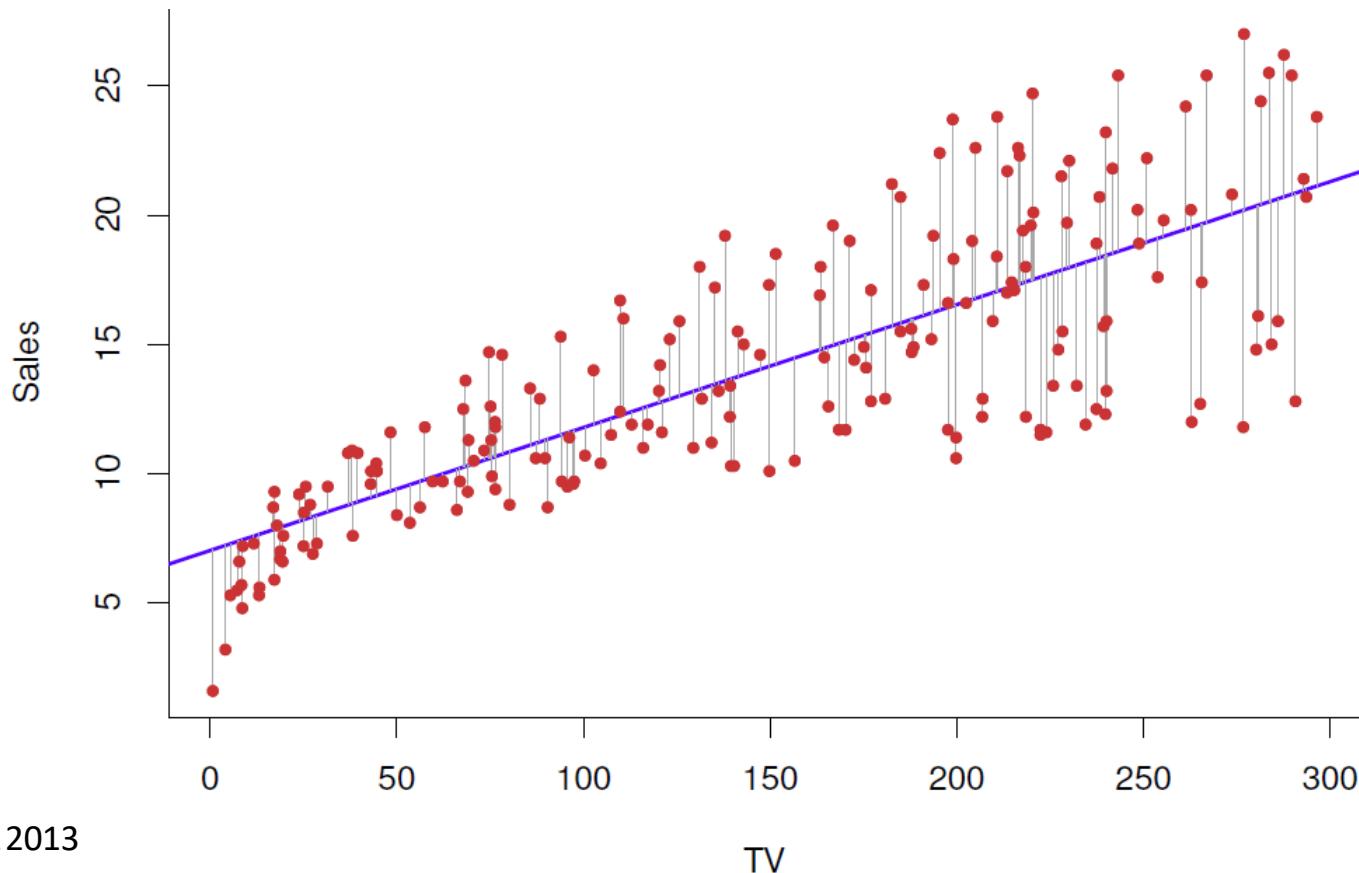
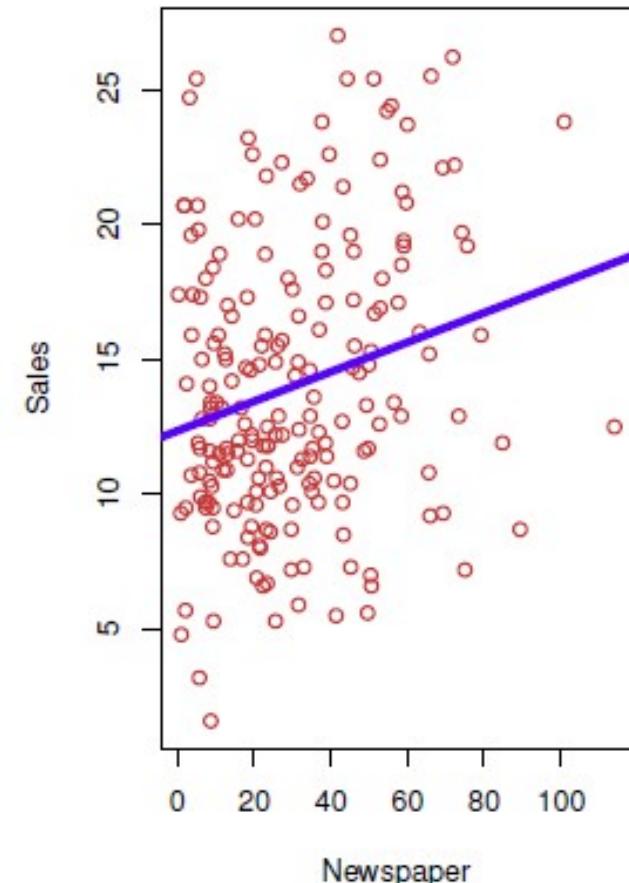
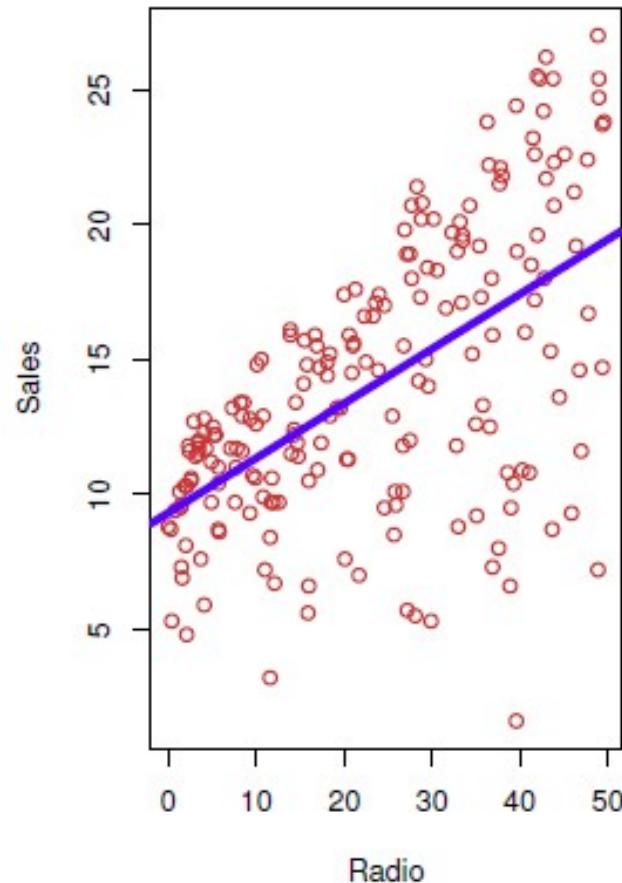
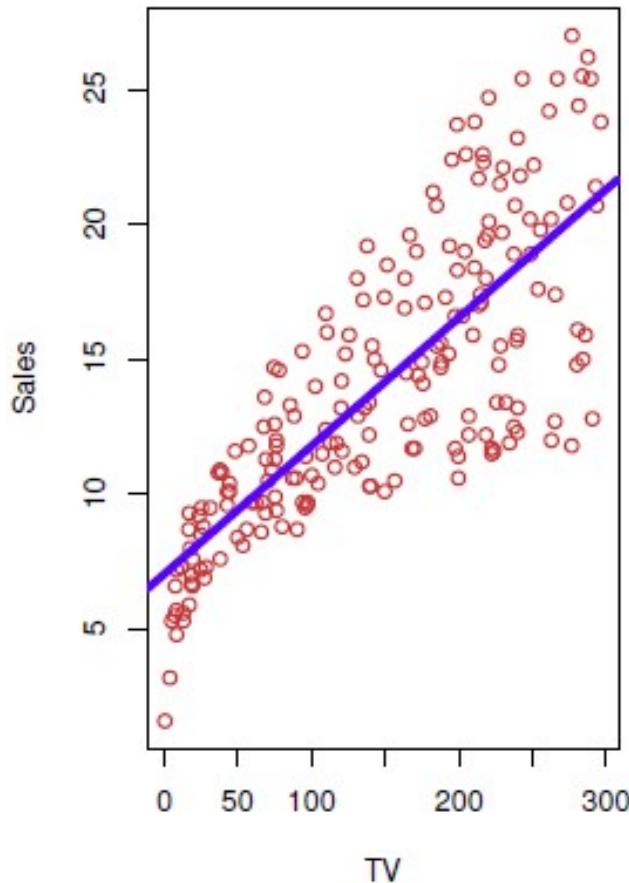


Figure 3.1 , ISL 2013



Hồi quy tuyến tính đơn giản



Phương pháp đánh giá

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (Y_i - \hat{Y}_i)^2}; MAE = \frac{1}{N} \sum_{i=1}^N |Y_i - \hat{Y}_i|$$

và $R^2 = 1 - \sum_{i=1}^N (Y_i - \hat{Y}_i)^2 / \sum_{i=1}^N (Y_i - \bar{Y})^2$.



Ví dụ

X kilos	Y giá \$
17	132
21	150
35	160
39	162
50	149
65	170

$$\bar{x} = 37.83$$

$$\bar{y} = 153.83$$

$$SS_{xy} = 891.83$$

$$SS_x = 1612.83$$

$$\hat{\beta}_1 = \frac{SS_{xy}}{SS_x} = \frac{891.83}{1612.83} = 0.533$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x} = 153.83 - 0.533 \times 37.83 = 132.91$$

phương trình tìm được là

$$Y = 132.91 + 0.533 * X$$



jupyter Linear Reg 1

```
File Edit View Insert Cell Kernel Widgets Help  
Run Code
```

```
In [ ]: import numpy as np
from sklearn.linear_model import LinearRegression

In [ ]: #You should call .reshape() on x because this array is required to be two-dimensional,
#or to be more precise, to have one column and as many rows as necessary.
#That's exactly what the argument (-1, 1) of .reshape() specifies.
x = np.array([17, 21, 35, 39, 50, 65]).reshape((-1, 1))
y = np.array([132, 150, 160, 162, 149, 170])

In [ ]: print(x)

In [ ]: print(y)

In [ ]: lm = LinearRegression()

In [ ]: lm.fit(x, y)

In [ ]: r_sq = lm.score(x, y)

In [ ]: print("Hệ số R^2:", r_sq)

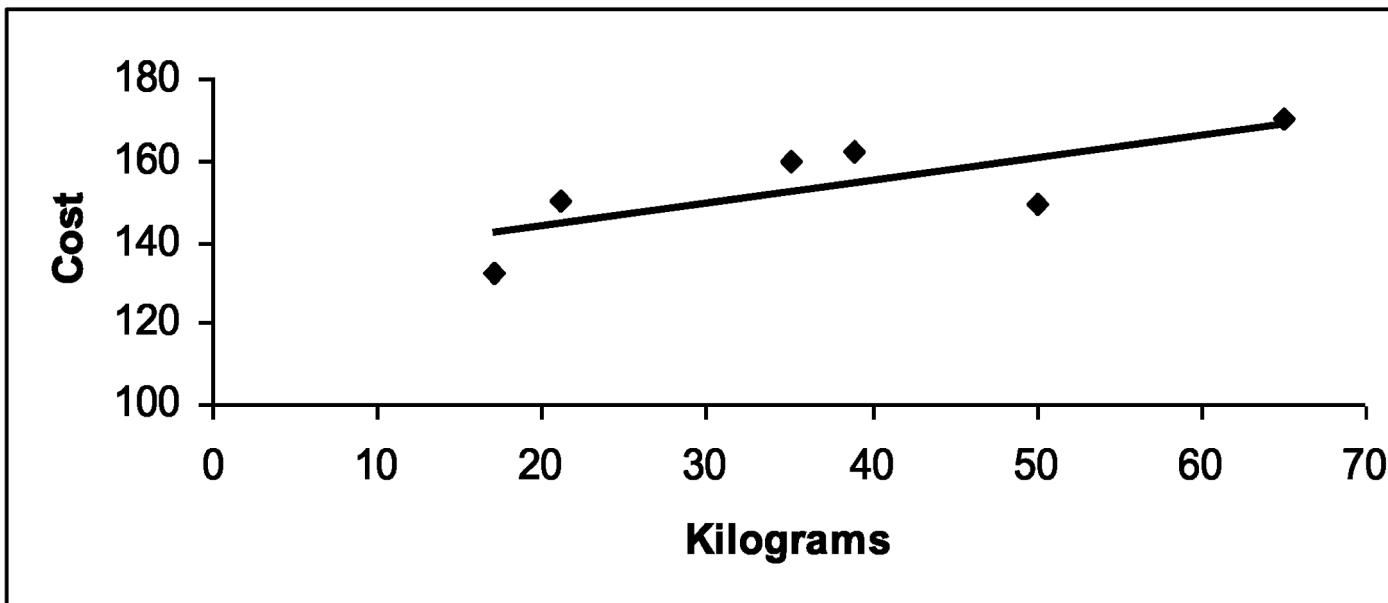
In [ ]: print('Hệ số chặn:', lm.intercept_)
print('Độ dốc:', lm.coef_)

In [ ]: y_pred = lm.predict(x)
print('Kết quả dự đoán:', y_pred)
```



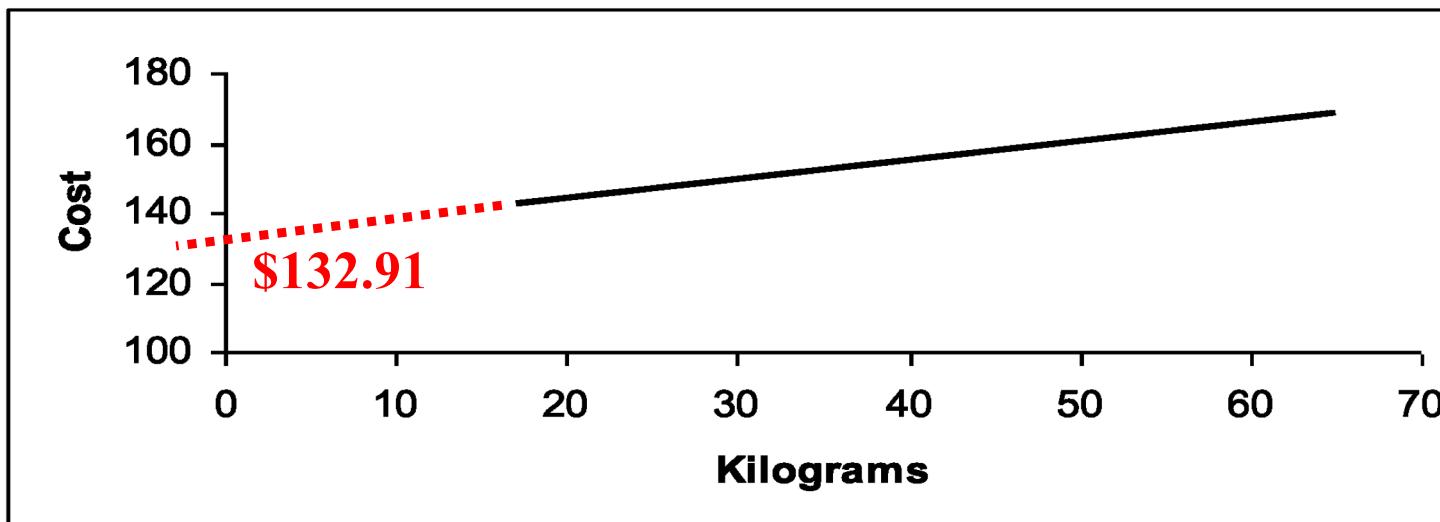
Diễn giải tham số

Trong ví dụ trước, tham số ước lượng $\hat{\beta}_1$ của độ dốc là 0.553. Điều này có nghĩa là khi thay đổi 1 kg của X, giá của Y thay đổi 0.553 \$



Diễn giải tham số

$\hat{\beta}_0$ là hệ số chẵn của Y. Nghĩa là, điểm mà đường thẳng cắt trục tung Y. Trong ví dụ này là \$132.91



Đây là giá trị của Y khi X = 0



Dữ liệu phân tích: Boston

- **Boston data:** liên quan đến giá nhà đất
- Các biến số
 - **crim:** tỉ lệ tội phạm của thị trấn
 - **zn:** tỉ lệ khu đất có diện tích trên 25,000 feet vuông
 - **indus:** tỉ lệ doanh nghiệp tương đối lớn
 - **chas:** gần sông Charles (1=yes, 0=no)
 - **nos:** nồng độ nitric oxides (parts/10 triệu)
 - **rm:** số phòng trung bình mỗi nhà
 - **age:** tỉ lệ căn hộ (unit) xây trước 1940
 - **dis:** khoảng cách đến các trung tâm kĩ nghệ (tìm việc làm)



Dữ liệu phân tích: Boston

- **Boston data:** liên quan đến giá nhà đất
- Các biến số
 - **rad:** chỉ số gần xa lộ radial
 - **tax:** tỉ suất thuế tinh trên \$10,000
 - **ptratio:** tỉ số học trò trên giáo viên của thị trấn
 - **black:** chỉ số về số người da đen trong thị trấn ($Bk - 0.63)^2$
 - **Istat:** tỉ lệ dân số thành phần kinh tế thấp
 - **PRICE:** trị giá nhà (\$1000)



Ước tính bằng Python

- Chúng ta muốn ước tính mối liên quan giữa số phòng (rm) và giá căn nhà
- Mô hình hồi qui tuyến tính đơn giản:

$$\text{PRICE} = \beta_0 + \beta_1 * \text{RM} + \varepsilon$$

- Python

```
#y is the dependent variable.  
y = boston_df['PRICE']  
#X is "RM"  
x = boston_df['RM'].values.reshape(-1, 1)  
  
from sklearn.linear_model import LinearRegression  
lm = LinearRegression()  
#lm.fit( x_train, y_train )  
lm.fit(x, y)  
# regression coefficients  
print('Coefficients: ', lm.coef_)  
print('Intercept: \n', lm.intercept_)
```



Diễn giải kết quả

```
from sklearn.linear_model import LinearRegression  
lm = LinearRegression()  
#lm.fit( X_train, y_train )  
lm.fit(X, y)  
# regression coefficients  
print('Coefficients: ', lm.coef_)  
print('Intercept: \n', lm.intercept_)
```

Coefficients: [9.10210898]
Intercept: -34.67062077643857

- Nhớ rằng mô hình là:

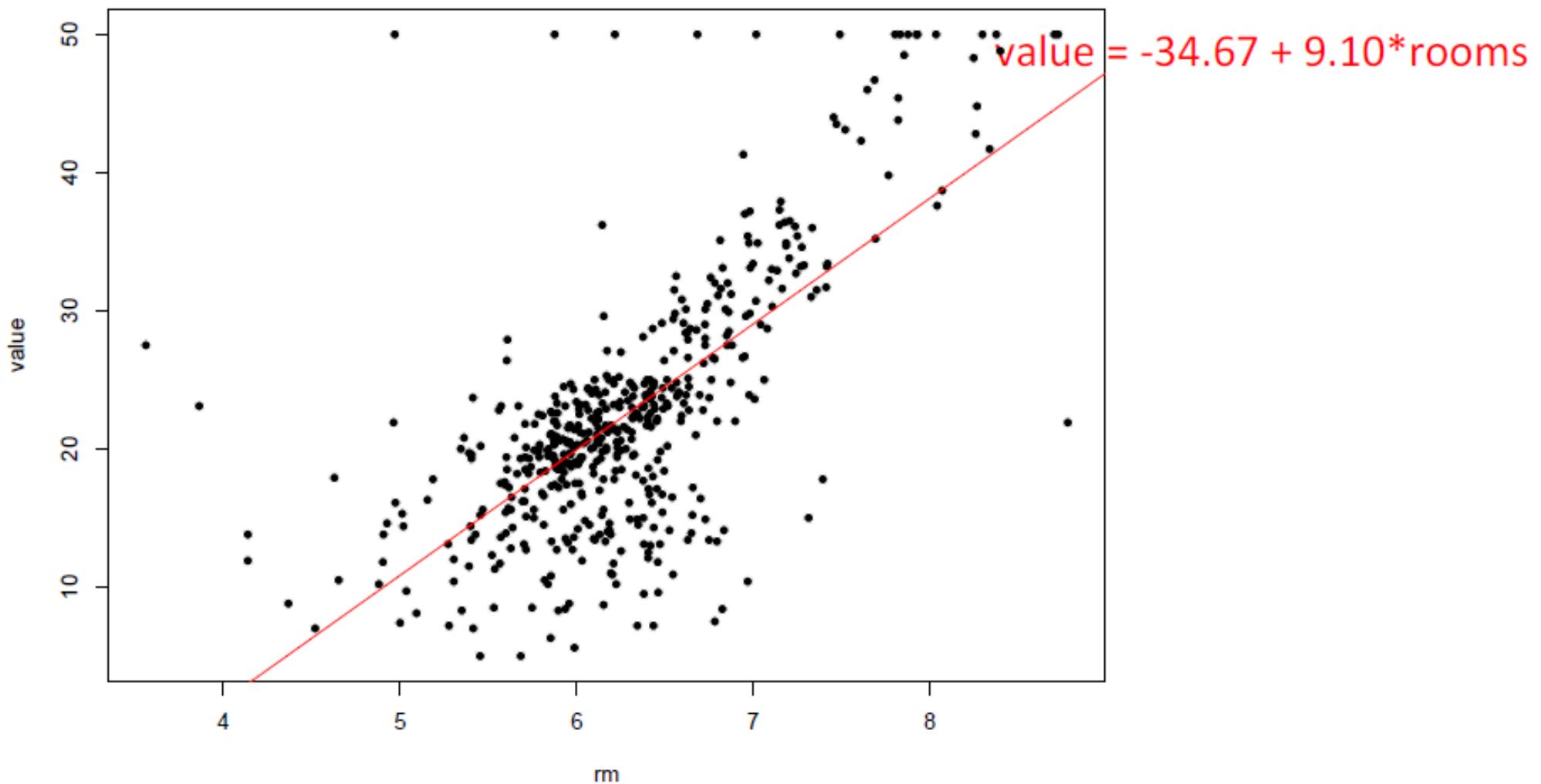
$$\text{PRICE} = a + b * rm$$

- Phương trình:

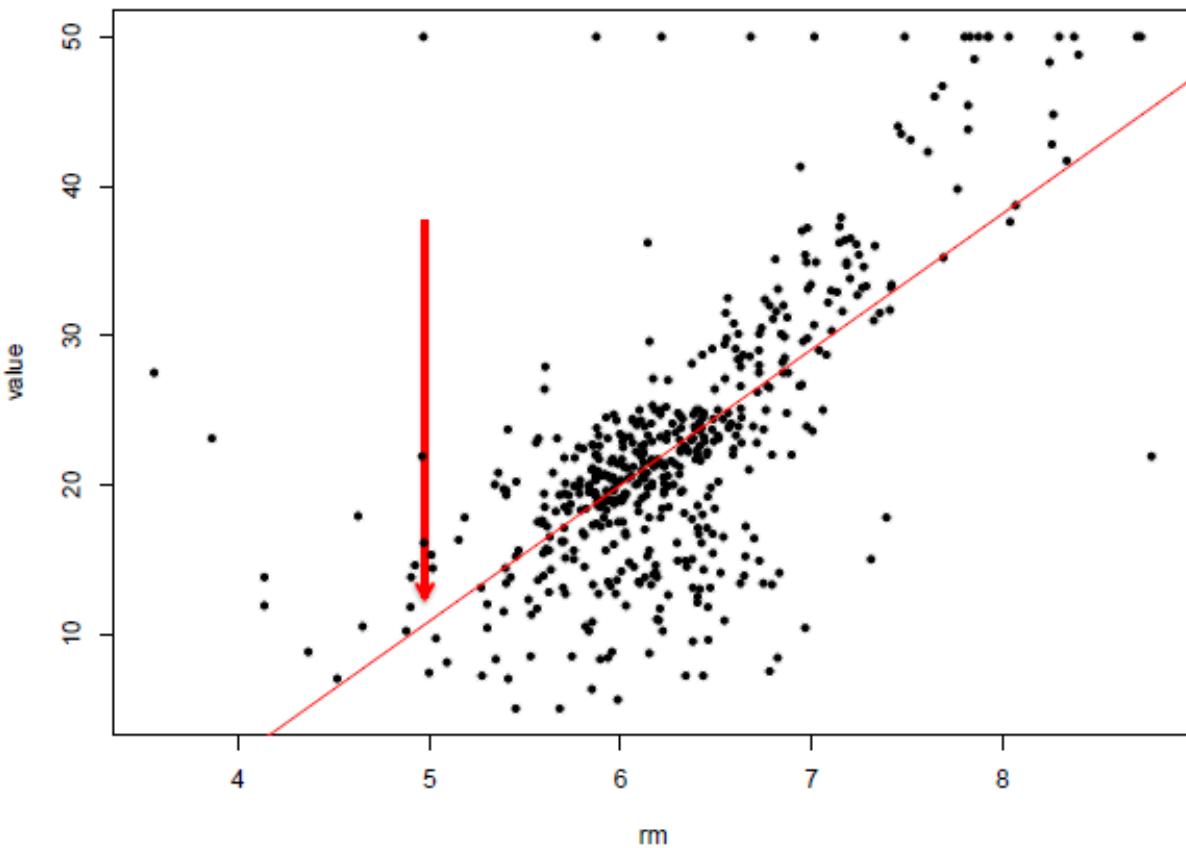
$$\text{PRICE} = -34.67 + 9.10 * \text{rooms}$$

- Ý nghĩa: nhà có thêm 1 phòng tăng 9100 USD cho giá trị căn nhà.





Ý nghĩa của đường biểu diễn



Giá trị trung bình (kì vọng)

$$\text{PRICE} = -34.67 + 9.10 \cdot \text{rooms}$$

Khi room = 5,

$$\text{PRICE} = -34.67 + 9.10 \cdot 5 = 10.83$$

Khi room = 6

$$\text{PRICE} = -34.67 + 9.10 \cdot 6 = 19.93$$

Khi room = 8

$$\text{PRICE} = -34.67 + 9.10 \cdot 8 = 38.13$$



Hồi quy tuyến tính đa biến

- **Hồi quy tuyến tính đa biến:** mô hình có nhiều hơn 1 biến dùng để dự đoán biến đích

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_d X_d + \epsilon$$



Hồi quy tuyến tính đa biến

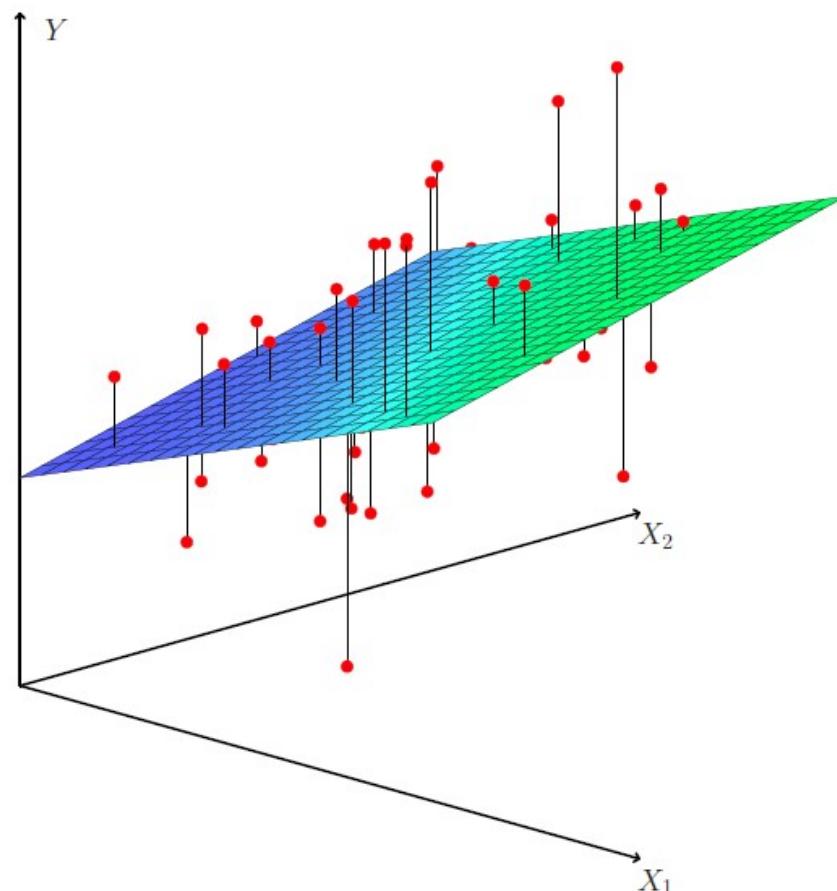


Figure 3.4 , ISL 2013



Hồi quy tuyến tính đa biến

- Diễn giải hệ số β_j :

khi tăng X_j lên một đơn vị \rightarrow Y sẽ tăng trung bình một lượng là β_j

	Coefficient
Intercept	2.939
TV	0.046
radio	0.189
newspaper	-0.001



Bình phương nhỏ nhất

- Tìm các ước số bằng phương pháp bình phương nhỏ nhất

$$\hat{\beta} = \arg \min_{\beta} \|Y - X^T \beta\|^2 \quad X = \begin{bmatrix} 1 & X^{(1)^T} \\ & \cdots \\ 1 & X^{(n)^T} \end{bmatrix} \quad \hat{\beta} = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \vdots \\ \hat{\beta}_d \end{bmatrix}$$
$$Y = \begin{bmatrix} Y^{(1)} \\ \vdots \\ Y^{(n)} \end{bmatrix}$$

- Giải phương trình để tìm $\hat{\beta}$:

$$X^T X \hat{\beta} = X^T Y \quad \rightarrow \quad \hat{\beta} = (X^T X)^{-1} X^T Y$$



Hồi quy tuyến tính đa biến

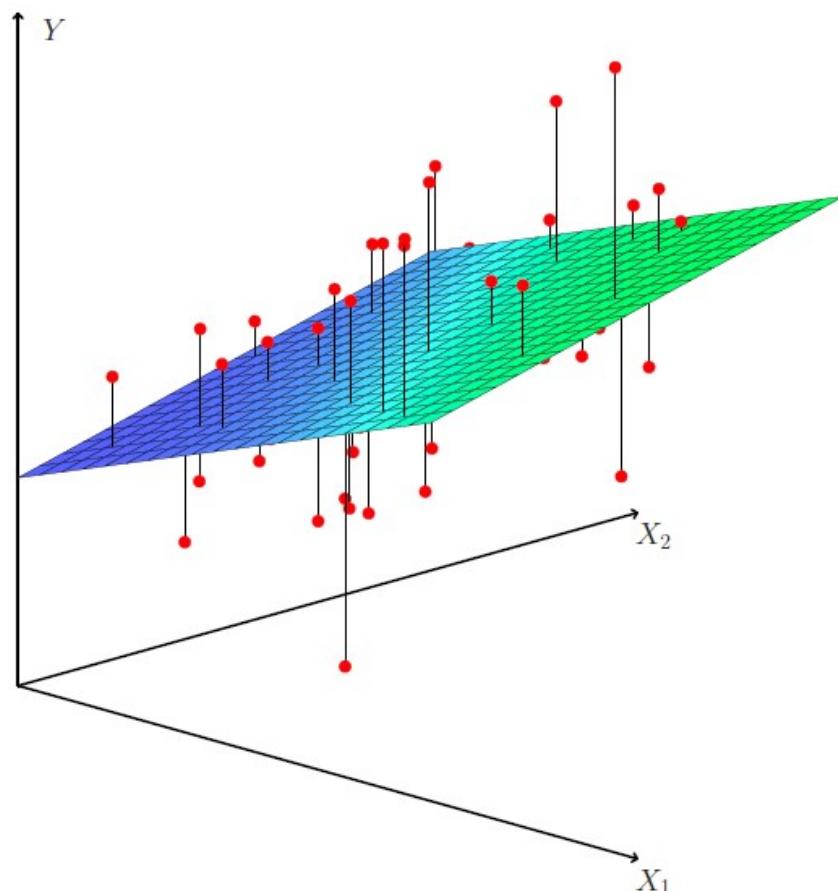


Figure 3.4 , ISL 2013



Ví dụ

Cho

$$\mathbf{y} = \begin{bmatrix} 6 \\ 9 \\ 12 \\ 5 \\ 13 \\ 2 \end{bmatrix}$$

$$\mathbf{X} = \begin{bmatrix} 1 & 3 & 9 & 16 \\ 1 & 6 & 13 & 13 \\ 1 & 4 & 3 & 17 \\ 1 & 8 & 2 & 10 \\ 1 & 3 & 4 & 9 \\ 1 & 2 & 4 & 7 \end{bmatrix} \quad \hat{\boldsymbol{\beta}} = \begin{bmatrix} \hat{\beta}_0 \\ \hat{\beta}_1 \\ \hat{\beta}_2 \\ \hat{\beta}_3 \end{bmatrix}$$



Ví dụ

$$X^\tau = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 3 & 6 & 4 & 8 & 3 & 2 \\ 9 & 13 & 3 & 2 & 4 & 4 \\ 16 & 13 & 17 & 10 & 9 & 7 \end{bmatrix}$$

$$X^\tau X = \begin{bmatrix} 6 & 26 & 35 & 72 \\ 26 & 138 & 153 & 315 \\ 35 & 153 & 295 & 448 \\ 72 & 315 & 448 & 944 \end{bmatrix} \quad X^\tau y = \begin{bmatrix} 47 \\ 203 \\ 277 \\ 598 \end{bmatrix}$$



Ví dụ

$$\hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} = \begin{bmatrix} 2.59578 & -0.15375 & -0.01962 & -0.13737 \\ -0.15375 & 0.03965 & -0.00014 & -0.00144 \\ -0.01962 & -0.00014 & 0.01234 & -0.00431 \\ -0.13737 & -0.00144 & -0.00431 & 0.01406 \end{bmatrix} \begin{bmatrix} 47 \\ 203 \\ 277 \\ 598 \end{bmatrix}$$
$$= \begin{bmatrix} 3.20975 \\ -0.07573 \\ -0.11162 \\ 0.46691 \end{bmatrix}$$

$$\hat{\beta}_0 = 3.20975 \quad \hat{\beta}_1 = -0.07573 \quad \hat{\beta}_2 = -0.11162 \quad \hat{\beta}_3 = 0.46691$$

$$\hat{y} = 3.20975 - 0.07573x_1 - 0.11162x_2 + 0.46691x_3$$



Python code

```
In [8]: import numpy as np
from sklearn.linear_model import LinearRegression

In [6]: x = np.array([[1, 1, 1, 1, 1, 1], [3, 6, 4, 8, 3, 2], [9, 13, 3, 2, 4, 4], [16, 13, 17, 10, 9, 7]])
y = np.array([6, 9, 12, 5, 13, 2])
print(x.T)

[[ 1   3   9  16]
 [ 1   6  13  13]
 [ 1   4   3  17]
 [ 1   8   2  10]
 [ 1   3   4   9]
 [ 1   2   4   7]]
```

```
In [12]: lm = LinearRegression()
lm.fit(x.T, y)
print('intercept:', lm.intercept_)
print('beta_i:', lm.coef_)

intercept: 3.2097464603226875
beta_i: [ 0.         -0.07573263 -0.11162331  0.46690813]
```



```

import matplotlib.pyplot as plt
import numpy as np
from sklearn import datasets, linear_model, metrics

# load the boston dataset
boston = datasets.load_boston(return_X_y=False)

# defining feature matrix(X) and response vector(y)
X = boston.data
y = boston.target

# splitting X and y into training and testing sets
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.4,
                                                    random_state=1)

# create linear regression object
reg = linear_model.LinearRegression()

# train the model using the training sets
reg.fit(X_train, y_train)

# regression coefficients
print('Coefficients: ', reg.coef_)

# variance score: 1 means perfect prediction
print('Variance score: {}'.format(reg.score(X_test, y_test)))

```



Bài tập tại lớp

Cho bảng dữ liệu về chiều cao và cân nặng của 15 người như sau:

Chiều cao (cm)	Cân nặng (kg)	Chiều cao (cm)	Cân nặng (kg)
147	49	168	60
150	50	170	72
153	51	173	63
155	52	175	64
158	54	178	66
160	56	180	67
163	58	183	68
165	59		

Bài toán đặt ra là: liệu có thể dự đoán cân nặng của một người dựa vào chiều cao của họ không?



Dữ liệu định tính

- Xử lý dữ liệu dạng định tính (định danh, hạng mục) trong mô hình hồi quy tuyến tính
 - vd: biến “giới tính”: “male” hoặc “female”
- Nếu chỉ có 2 khả năng trên, ta tạo *biến giả (dummy variable)*

$$X_j = \begin{cases} 1 & \text{if female} \\ 0 & \text{if male} \end{cases}$$



Dữ liệu định tính

- Nếu có nhiều hơn 2 giá trị, ta biểu diễn biến chúng dùng nhiều biến giả
 - vd: biến “màu mắt”: “blue”, “green” or “brown”

$$X_j = \begin{cases} 1 & \text{if blue} \\ 0 & \text{if not blue} \end{cases}$$

$$X_{j+1} = \begin{cases} 1 & \text{if brown} \\ 0 & \text{if not brown} \end{cases}$$



Hồi quy tuyến tính

- **Ưu điểm:**
 - Mô hình đơn giản, dễ hiểu
 - Dễ diễn giải hệ số hồi quy
 - Nhận được kết quả tốt khi dữ liệu quan sát nhỏ
 - Nhiều cải tiến/mở rộng
- **Nhược điểm:**
 - Mô hình hơi đơn giản nên khó dự đoán chính xác với dữ liệu có miền giá trị rộng
 - Khả năng ngoại suy (extrapolation) kém
 - Nhạy cảm với dữ liệu ngoại lai (outliers) – do dung phương pháp bình phương nhỏ nhất



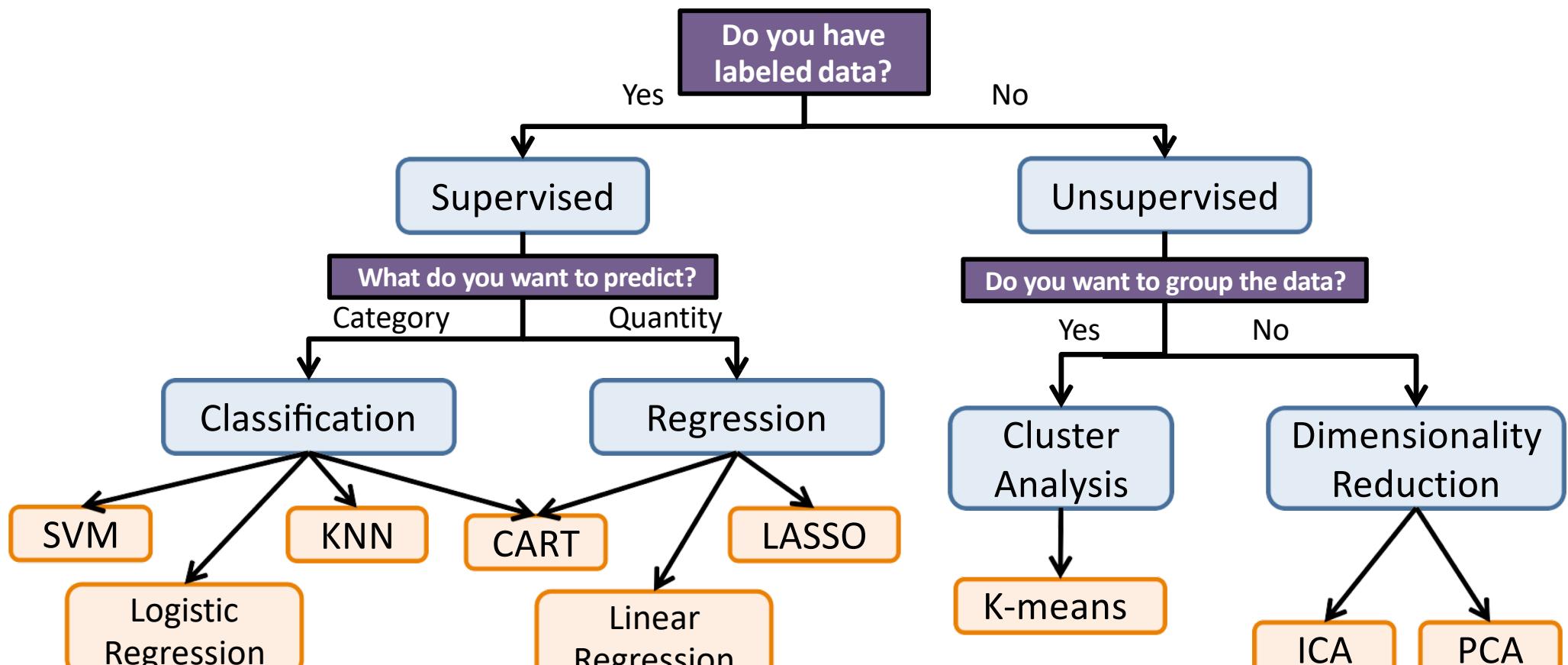
Câu hỏi?



Kỹ thuật đánh giá chéo, hiệu chỉnh mô hình



Các dạng giải thuật Học máy



Hàm tổn thất Loss Functions



Hồi quy tuyến tính đơn giản

Bình phương nhỏ nhất

- Sử dụng phương pháp bình phương nhỏ nhất để đo lường độ xấp xỉ của mô hình áp dụng trên dữ liệu
- *Phần dư (Residual)*: sai số giữa giá trị quan sát được và giá trị dự đoán.

$$r^{(i)} = Y^{(i)} - \hat{Y}^{(i)}$$

- *Tổng phần dư bình phương-Residual sum of squares (RSS)*:

$$RSS = (r^{(1)})^2 + (r^{(2)})^2 + \dots + (r^{(n)})^2$$

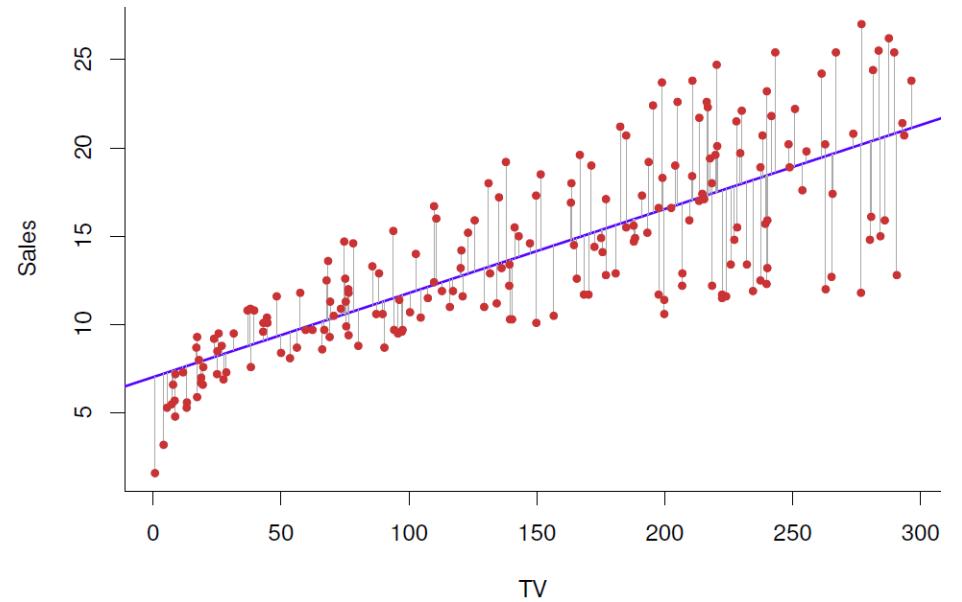
- *Lỗi bình phương trung bình-Mean squared error (MSE)*:

$$MSE = \frac{1}{n} \sum_{i=1}^n (Y^{(i)} - \hat{Y}^{(i)})^2 = \frac{1}{n} (RSS)$$



Loss Functions

$$L(\theta_i, \hat{\theta}_i)$$



Loss Functions

$$L(\theta_i, \hat{\theta}_i)$$

Sai số bình phương (Squared error)

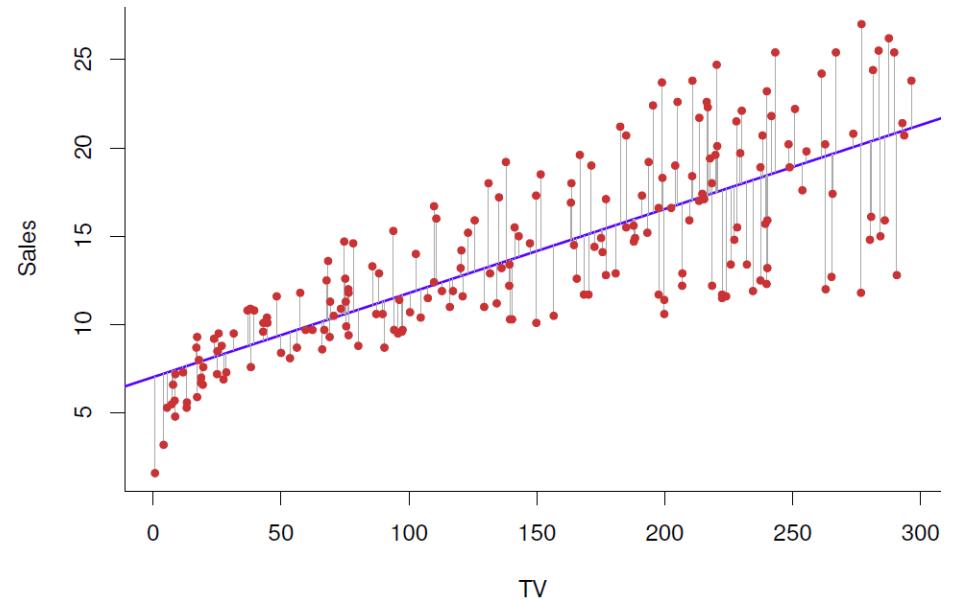
$$\sum_i (\theta_i - \hat{\theta}_i)^2$$

Sai số tuyệt đối (Absolute error)

$$\sum_i |\theta_i - \hat{\theta}_i|$$

Indicator error

$$\sum_i I(\theta_i \neq \hat{\theta}_i)$$



Học máy chỉ để giải 1 vấn đề

$$\hat{f} = \operatorname{argmin}_{\tilde{f}} E[L(Y, \tilde{f}(X))]$$

argument minimum: Cho giá trị nhỏ nhất của 1 hàm số trong miền xác định



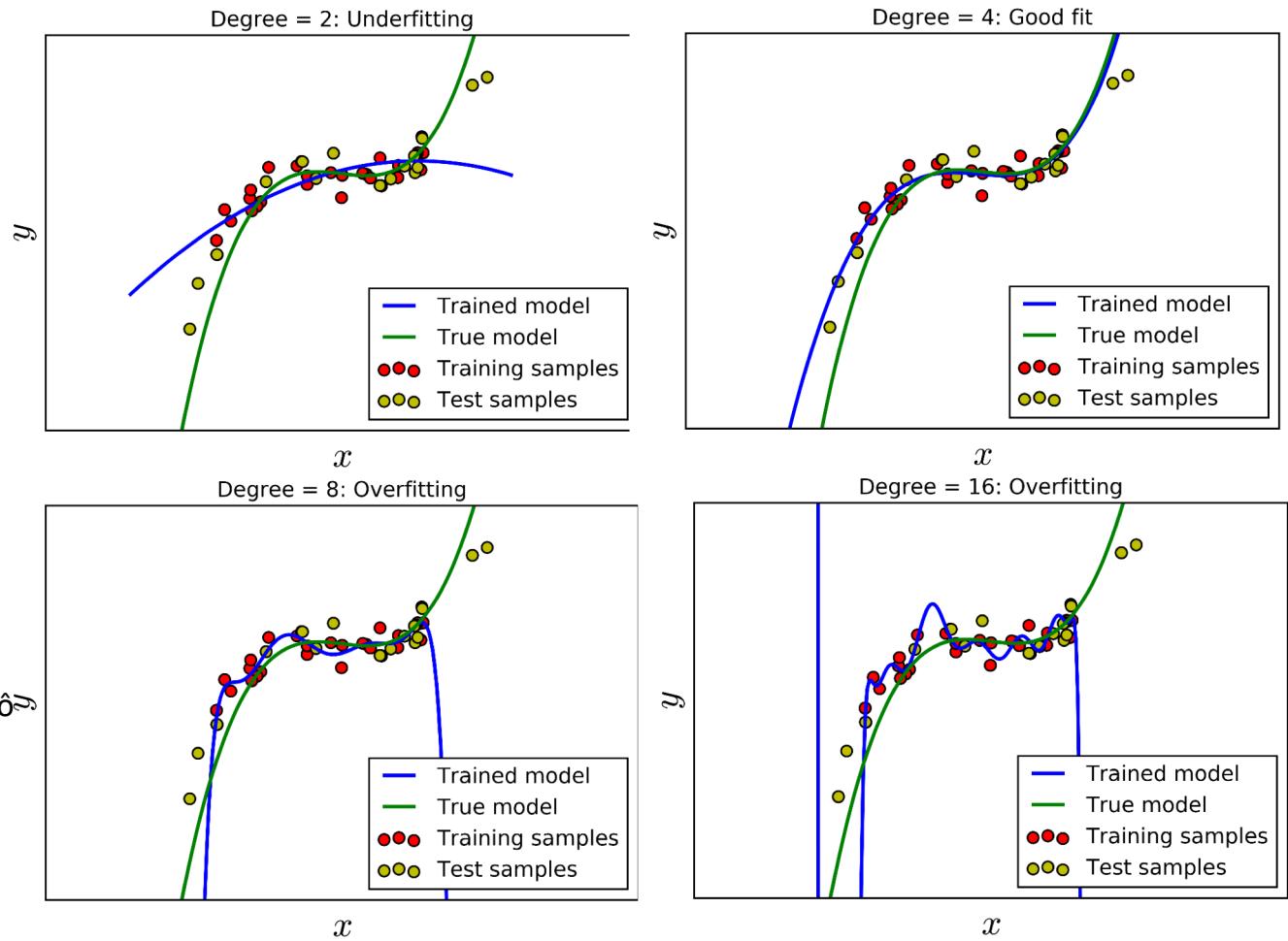
Hiện tượng quá khớp

Overfitting



Underfitting và Overfitting

- Có 50 điểm dữ liệu được tạo bằng một đa thức bậc ba cộng thêm nhiễu.
- Đồ thị của đa thức có màu xanh lục (true model).
- Bài toán: Giả sử ta không biết mô hình ban đầu mà chỉ biết các điểm dữ liệu, hãy tìm một mô hình “tốt” để mô tả dữ liệu đã cho?
- Với $d=2$, mô hình không thực sự tốt vì dự đoán quá khác so với mô hình thực: *underfitting*
- Với $d=8$ và $d=16$, với các điểm dữ liệu trong khoảng của training data, mô hình dự đoán và mô hình thực là khá giống nhau. Tuy nhiên, về phía phải, đa thức bậc 8 và 16 cho kết quả hoàn toàn ngược với xu hướng của dữ liệu: *Overfitting*.
- $d=4$, mô hình tốt nhất.



Kỹ thuật đánh giá chéo

Cross-validation



Kỹ thuật đánh giá chéo

“Dùng lỗi trên tập dữ liệu kiểm thử để ước lượng lỗi dự đoán”

$$err = E[L(Y, \hat{f}(X))]$$



Tập đánh giá (Validation)

- Thường chia tập dữ liệu ra thành training data và test data.
- Chú ý: khi xây dựng mô hình, ta không được sử dụng test data.
- Làm cách nào để biết được chất lượng của mô hình với unseen data (tức dữ liệu chưa nhìn thấy bao giờ)?



Tập đánh giá (Validation)

- Phương pháp: trích từ training data ra một tập con nhỏ và thực hiện việc đánh giá mô hình trên tập con này.
- Tập con nhỏ được trích ra từ training set này được gọi là validation set. Lúc này, training set là phần còn lại của training set ban đầu.
- Train error được tính trên training set mới này.
- Validation error: Lỗi được tính trên tập validation.



Tập đánh giá (Validation)

- Tìm mô hình sao cho cả *train error* và *validation error* đều nhỏ, qua đó có thể dự đoán được rằng *test error* cũng nhỏ.
- Phương pháp thường được sử dụng là sử dụng nhiều mô hình khác nhau. Mô hình nào cho *validation error* nhỏ nhất sẽ là mô hình tốt.

Tuy nhiên, khi ta có rất hạn chế số lượng dữ liệu để xây dựng mô hình. Nếu lấy quá nhiều dữ liệu trong tập training ra làm dữ liệu validation, phần dữ liệu còn lại của tập training là không đủ để xây dựng mô hình. Nếu ta giữ tập validation phải thật nhỏ để có được lượng dữ liệu cho training đủ lớn. Một vấn đề khác nảy sinh, hiện tượng overfitting lại có thể xảy ra với tập training còn lại.

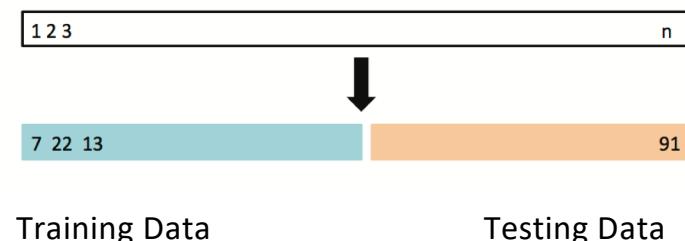
Giải pháp: **Cross-validation (Kỹ thuật đánh giá chéo).**



Kỹ thuật đánh giá chéo

- *Cross validation* là một cải tiến của *validation* với lượng dữ liệu trong tập validation là nhỏ nhưng chất lượng mô hình được đánh giá trên nhiều tập *validation* khác nhau.
 - Chia tập training ra k tập con không có phần tử chung, có kích thước gần bằng nhau.
 - Tại mỗi lần kiểm thử, một trong số k tập con được lấy ra làm *validation set*. Mô hình sẽ được xây dựng dựa vào hợp của $k-1$ tập con còn lại.
 - Mô hình cuối được xác định dựa trên trung bình của các *train error* và *validation error*.
- Cách làm này còn có tên gọi là **k-fold cross validation**.

Tập huấn luyện - Training Set
Tập kiểm thử - Test Set
Tập đánh giá - Validation Set



Kỹ thuật đánh giá chéo K-fold

Ví dụ 5-fold



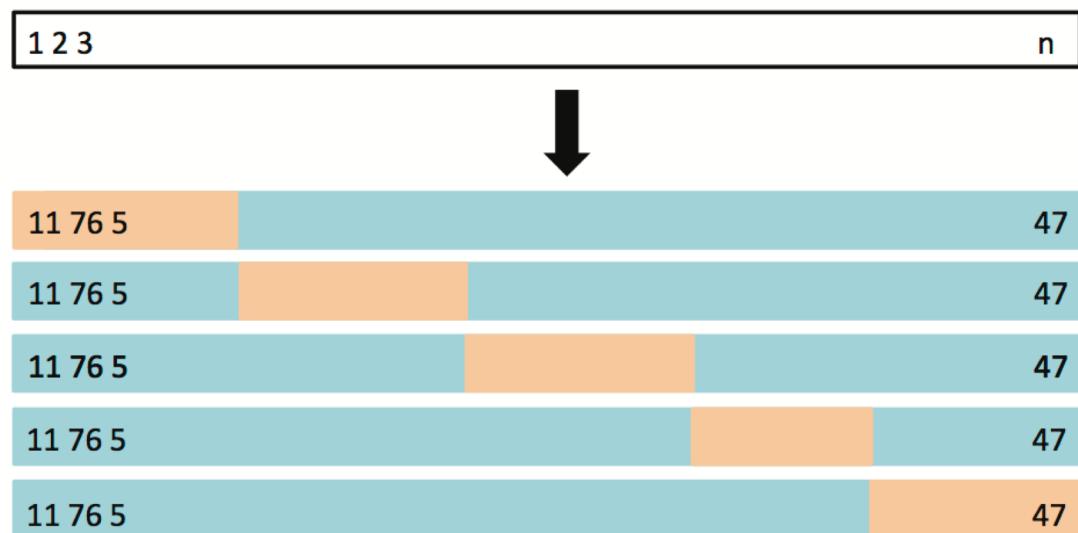
$$\text{CV}(\hat{f}) = \frac{1}{N} \sum_{i=1}^N L(y_i, \hat{f}^{-\kappa(i)}(x_i))$$

Hastie, Trevor, et al. The elements of statistical learning. Vol. 2. No. 1. New York: Springer, 2009.



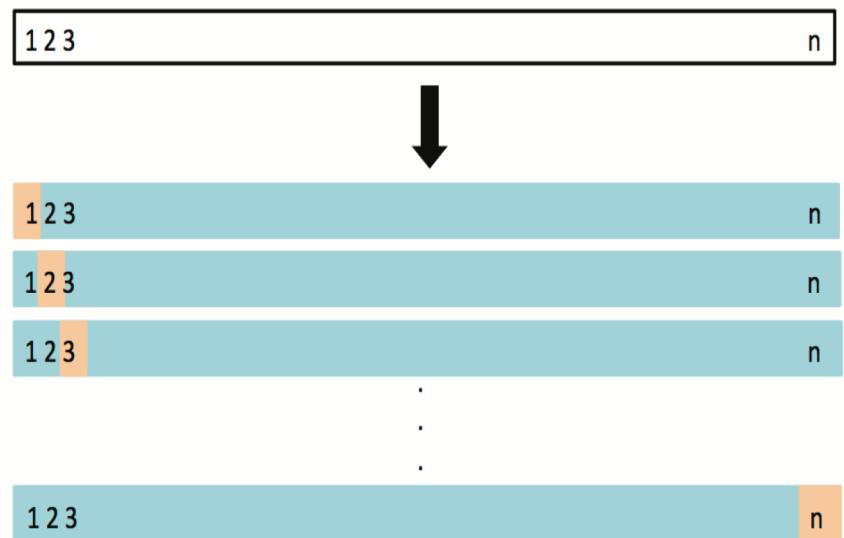
Kỹ thuật đánh giá chéo

5-fold và 10-fold thường được ưa dùng (lỗi bias cao, phương sai thấp)



Kỹ thuật đánh giá chéo

- Khi k bằng với số lượng phần tử N trong tập *training* ban đầu, tức mỗi tập con có đúng 1 phần tử, ta gọi kỹ thuật này là **leave-one-out**.
(lỗi bias thấp, phương sai cao)



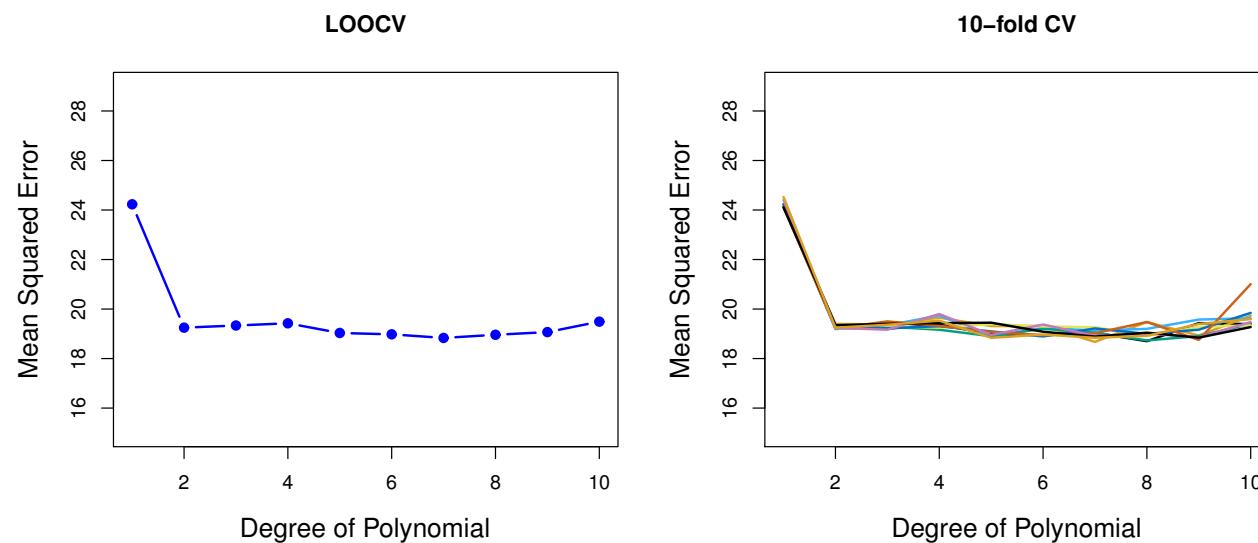
Auto Data: LOOCV vs. K-fold CV

Hình trái: Sai số LOOCV

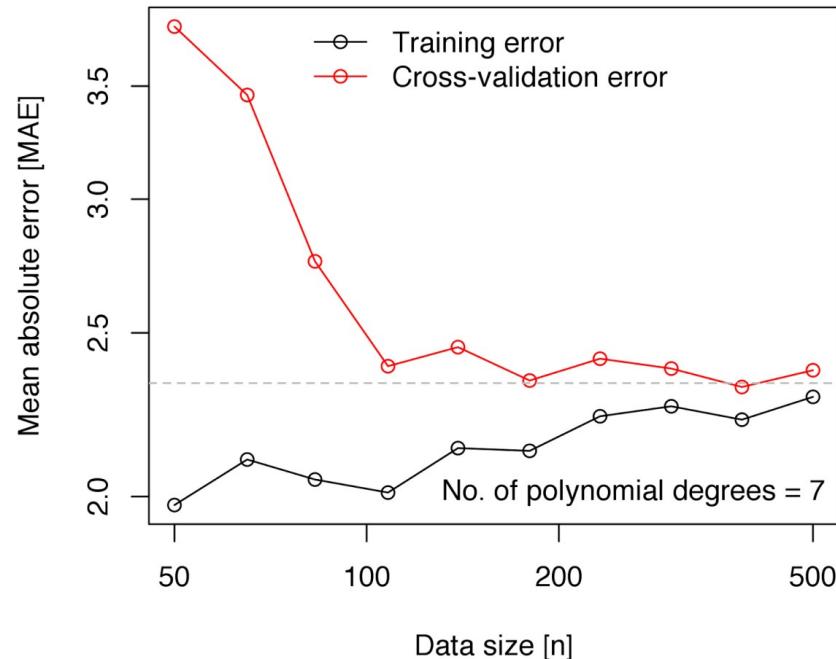
Hình phải: 10-fold CV được chạy nhiều lần, đồ thị biểu diễn sai khác nhõ về lõi CV

LOOCV là trường hợp đặc biệt của k-fold, khi k = N

Cả hai đều ổn định, tuy nhiên LOOCV mất nhiều thời gian tính toán hơn!



Kỹ thuật đánh giá chéo



Ta cần thêm biến (mô hình mới) hoặc thêm dữ liệu?



Kỹ thuật đánh giá chéo

- Nhược điểm lớn của *cross-validation* là số lượng *training runs* tỉ lệ thuận với k . Trong các bài toán Machine Learning, lượng tham số cần xác định thường lớn và khoảng giá trị của mỗi tham số cũng rộng.
- Vậy việc chỉ xây dựng một mô hình thôi đã rất phức tạp.
- Giải pháp giúp số mô hình cần huấn luyện giảm đi nhiều, thậm chí chỉ một mô hình. Cách này có tên gọi chung là *điều chỉnh mô hình (regularization)*.



Điều chỉnh mô hình

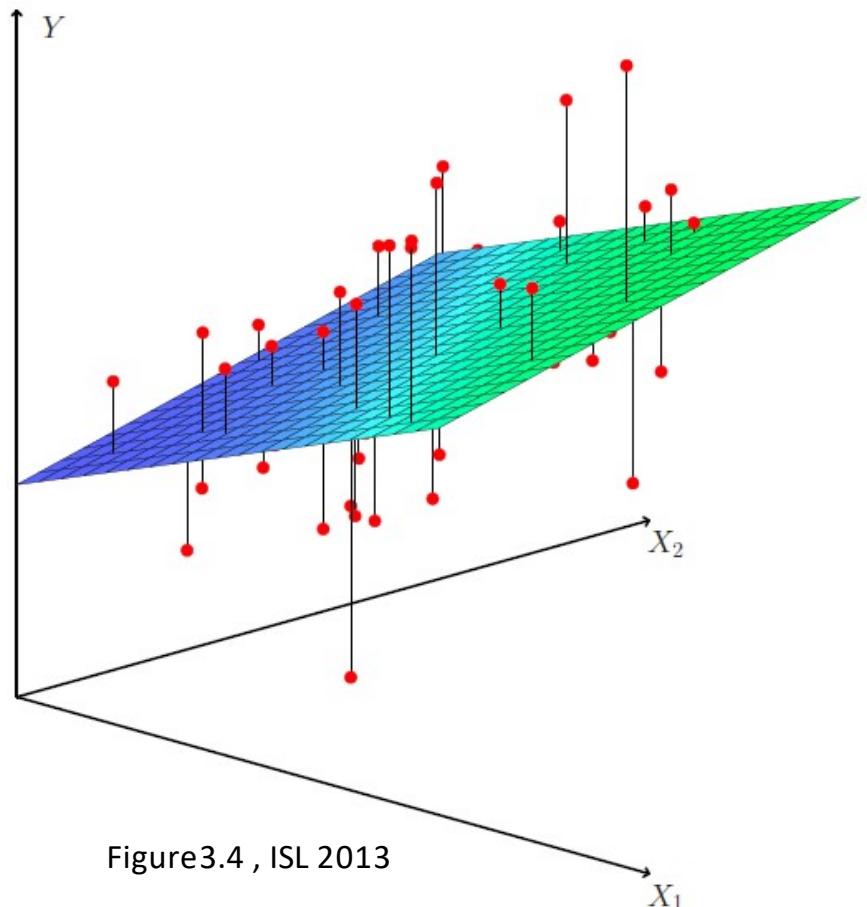
- *Regularization*, một cách cơ bản, là điều chỉnh mô hình một chút để tránh overfitting trong khi vẫn giữ được tính tổng quát của nó (tính tổng quát là tính mô tả được nhiều dữ liệu, trong cả tập training và test).
- Một cách cụ thể hơn, ta sẽ tìm cách *di chuyển* nghiệm của bài toán tối ưu hàm mất mát tới một điểm gần nó. Hướng di chuyển sẽ là hướng làm cho mô hình *ít phức tạp hơn* mặc dù giá trị của hàm mất mát có tăng lên một chút.



Mô hình có điều chỉnh



Nhắc lại: Hồi quy tuyến tính đa biến



$$Y = \beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2$$



Trường hợp quá nhiều biến

khi có quá nhiều biến đầu vào

$$Y = \beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 + \beta_3 \cdot X_3 + \beta_4 \cdot X_4 + \beta_5 \cdot X_5 + \beta_6 \cdot X_6 + \beta_7 \cdot X_7 + \beta_8 \cdot X_8$$

khi có tương tác giữa các biến đầu vào

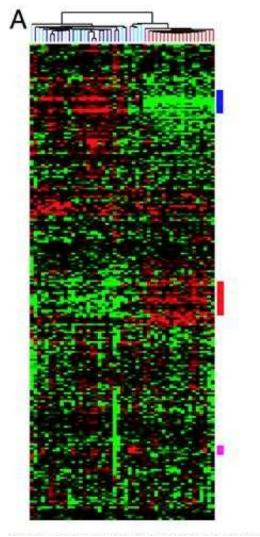
$$Y = \beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 + \beta_3 \cdot (X_1 X_2) + \beta_4 \cdot {X_1}^2 + \beta_5 \cdot {X_2}^2 + \beta_6 \cdot \log(X_1 / X_2) + \beta_7 \cdot \sin(X_1 - X_2)$$



Trường hợp quá nhiều biến

$$Y = \beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 + \beta_3 \cdot X_3 + \beta_4 \cdot X_4 + \beta_5 \cdot X_5 + \beta_6 \cdot X_6 + \beta_7 \cdot X_7 + \beta_8 \cdot X_8$$

Gene expression arrays



Điều gì xảy ra?

$$Y = \beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 + \beta_3 \cdot X_3 + \beta_4 \cdot X_4 + \beta_5 \cdot X_5 + \beta_6 \cdot X_6 + \beta_7 \cdot X_7 + \beta_8 \cdot X_8$$

Câu hỏi: Ta có 8 biến và có hàng trăm mẫu. Hai biến (X_3 và X_4) có tương quan yếu với Y (do đó cũng hữu dụng nhỏ cho dự đoán), tuy nhiên chúng có tương quan cao với các biến khác. Điều gì xảy ra khi diễn giải các hệ số β của hai biến X_3 và X_4 ?



Đa cộng tuyến (Multi-collinearity)

$$Y = \beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 + \beta_3 \cdot X_3 + \beta_4 \cdot X_4 + \beta_5 \cdot X_5 + \beta_6 \cdot X_6 + \beta_7 \cdot X_7 + \beta_8 \cdot X_8$$

- Theo giả thiết của phương pháp Hồi quy tuyến tính thì các biến độc lập không có mối quan hệ tuyến tính.
- Nếu quy tắc này bị vi phạm thì sẽ có hiện tượng đa cộng tuyến: là hiện tượng các biến độc lập trong mô hình phụ thuộc tuyến tính lẫn nhau và thể hiện được dưới dạng hàm số



Hồi quy tuyến tính đa biến

Quay lại hồi quy tuyến tính, ta cố gắng để cực tiểu hóa lỗi bình phương

$$\sum_{các mẫu} [Y - (\beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2)]^2$$



Hồi quy Ridge

Tìm giá trị β để cực tiểu lỗi phạt “penalized”, tương đương với

$$\sum_{samples} [Y - (\beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2)]^2 + \boxed{\lambda \cdot (\beta_0^2 + \beta_1^2 + \beta_2^2)}$$

L2



Hiệu chỉnh mô hình (Regularization)

Hồi quy Ridge

Tìm giá trị β để cực tiểu lỗi phạt “penalized”, tương đương với

$$\sum_{\text{các mẫu}} [Y - (\beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2)]^2 + \lambda \cdot (\beta_0^2 + \beta_1^2 + \beta_2^2)$$

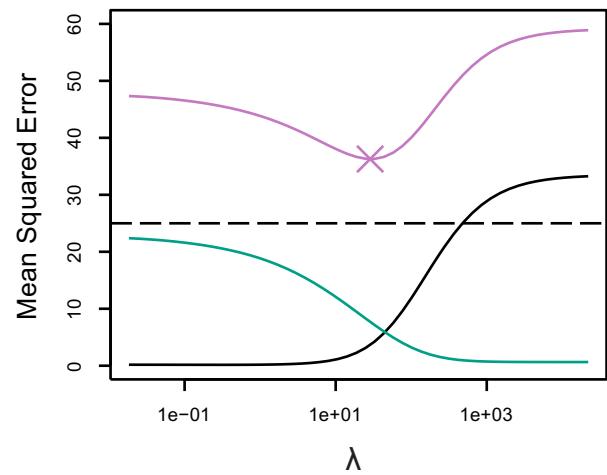
L2

hoặc viết ở dạng khác,

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \left(\|y - X\beta\|_2^2 + \lambda \|\beta\|_2^2 \right)$$



Hồi quy Ridge

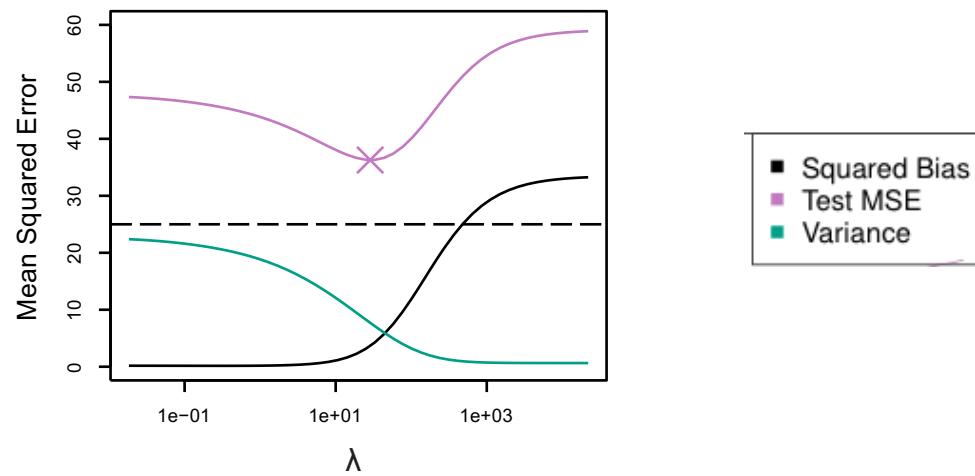


Đường cong nào là lỗi bias, đâu là phương sai, và đâu là lỗi dự đoán trên tập dữ liệu kiểm thử?

Hastie, Trevor, et al. Introduction to statistical learning.



Hồi quy Ridge



Hastie, Trevor, et al. Introduction to statistical learning.



Hiệu chỉnh mô hình

Ta đã xử lý:

- *Underdetermined*
- *Overfitting*
- Đa cộng tuyến (*Multi-collinearity*)

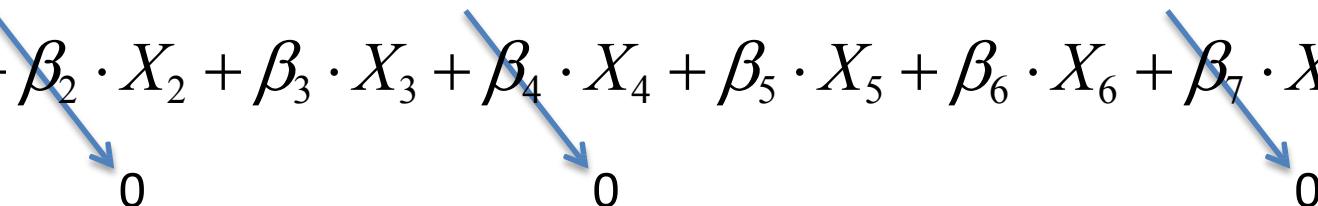
Vậy mô hình thừa là gì (*sparsity*)?

$$Y = \beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2 + \beta_3 \cdot X_3 + \beta_4 \cdot X_4 + \beta_5 \cdot X_5 + \beta_6 \cdot X_6 + \beta_7 \cdot X_7 + \beta_8 \cdot X_8$$

The equation shows a linear regression model. Three terms in the sum, corresponding to coefficients β_2 , β_4 , and β_7 , have arrows pointing down to the number 0, indicating that these coefficients are zero. This visualizes a sparse model where many features have no impact on the outcome.



Mô hình thưa (Sparsity)

$$Y = \beta_0 + \beta_1 \cdot X_1 + \cancel{\beta_2} \cdot X_2 + \beta_3 \cdot X_3 + \cancel{\beta_4} \cdot X_4 + \beta_5 \cdot X_5 + \beta_6 \cdot X_6 + \cancel{\beta_7} \cdot X_7 + \beta_8 \cdot X_8$$


- Dùng cho lựa chọn biến (Feature selection)
- Thời gian tính toán lâu (computational efficiency)



Mô hình thừa (Sparsity)

Lasso

“Least absolute shrinkage and selection operator”

$$\sum_{samples} [Y - (\beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2)]^2 + \boxed{\lambda \cdot |\beta_0| + |\beta_1| + |\beta_2|}$$

L1

Mô hình giống như hồi quy Rigde nhưng khác hàm phạt

Tibshirani, Robert. "Regression shrinkage and selection via the lasso." Journal of the Royal Statistical Society. Series B (Methodological)(1996): 267–288.



Lasso

“Least absolute shrinkage and selection operator”

$$\sum_{samples} [Y - (\beta_0 + \beta_1 \cdot X_1 + \beta_2 \cdot X_2)]^2 + \lambda \cdot (|\beta_0| + |\beta_1| + |\beta_2|)$$

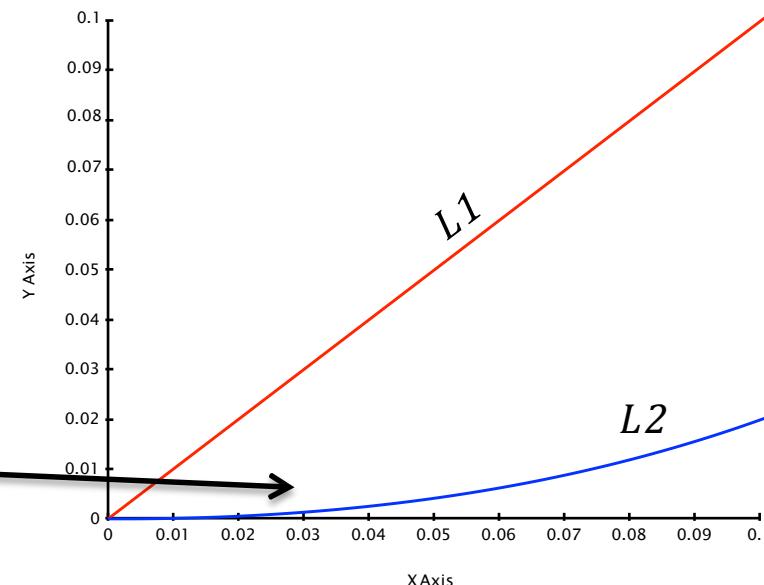
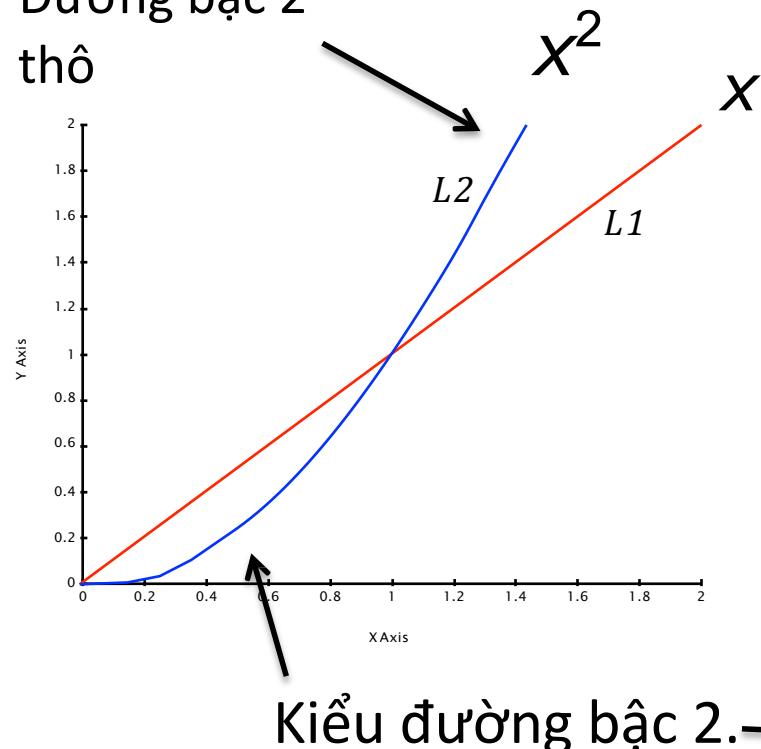
hoặc viết ở dạng khác,

$$\hat{\beta} = \operatorname{argmin}_{\beta} \left(\|y - X\beta\|_2^2 + \lambda \|\beta\|_1 \right)$$



Phương thức phạt (Penalties)

Đường bậc 2
thô



Mục tiêu khác: Mô hình thưa

Lasso

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^N \left(y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^p |\beta_j| \right\}$$

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^N \left(y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j \right)^2 \\ \text{s.t. } \sum_{j=1}^p |\beta_j| \leq t$$

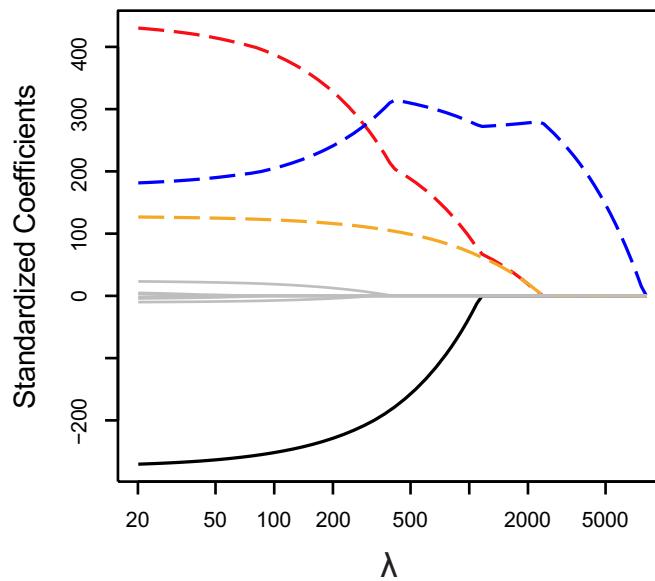
Ridge

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^N \left(y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j \right)^2 + \lambda \sum_{j=1}^p \beta_j^2 \right\}$$

$$\hat{\beta} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^N \left(y_i - \beta_0 - \sum_{j=1}^p x_{ij} \beta_j \right)^2 \\ \text{s.t. } \sum_{j=1}^p \beta_j^2 \leq t$$



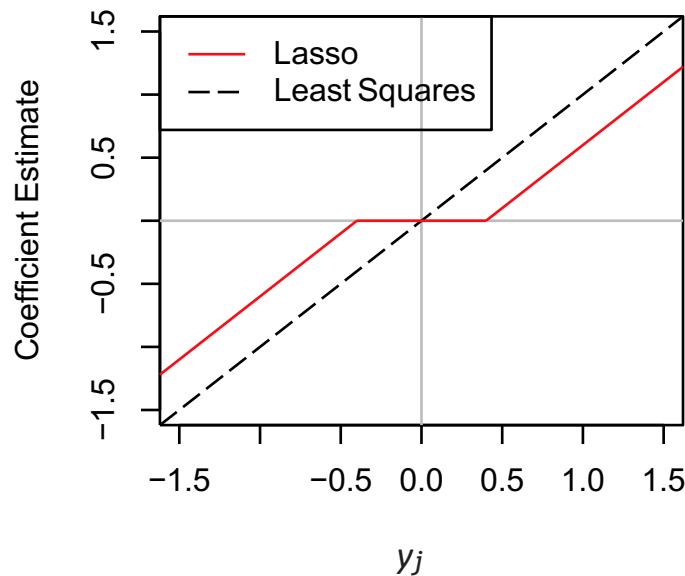
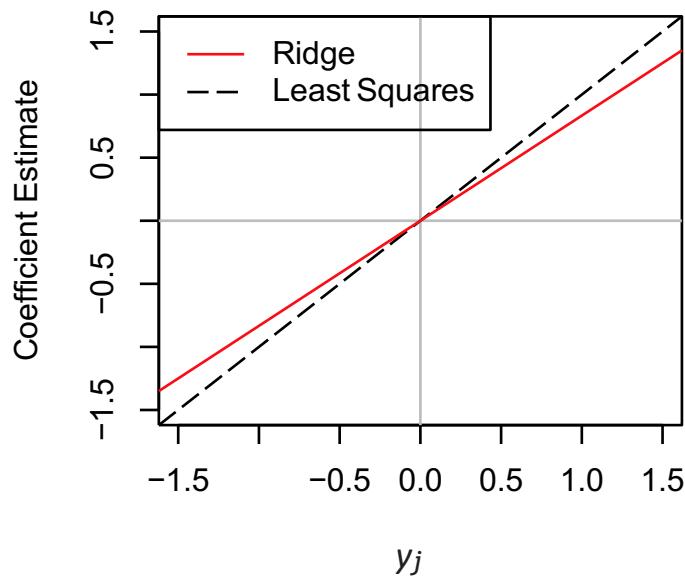
L1 (lasso) tính nhanh hơn và thưa



Hastie, Trevor, et al. Introduction to statistical learning.



Ridge vs. Lasso: Mô hình thưa



Hastie, Trevor, et al. Introduction to statistical learning.



Câu hỏi?

