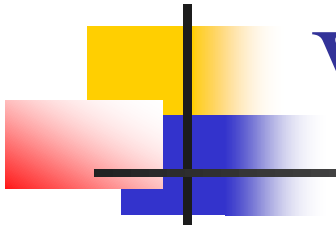
QUÁ KHỚP

Overfitting

TS. Nguyễn Thị Kim Ngân



Ví dụ

- Data set: 50 điểm dữ liệu
  - Trainning set: 30 điểm màu đỏ
  - Test set: 20 điểm màu vàng
- Tìm một mô hình tốt để mô tả quan hệ giữa đầu vào và đầu ra của dữ liệu?

Giả sử biết rằng mô hình được mô tả bởi một đa thức

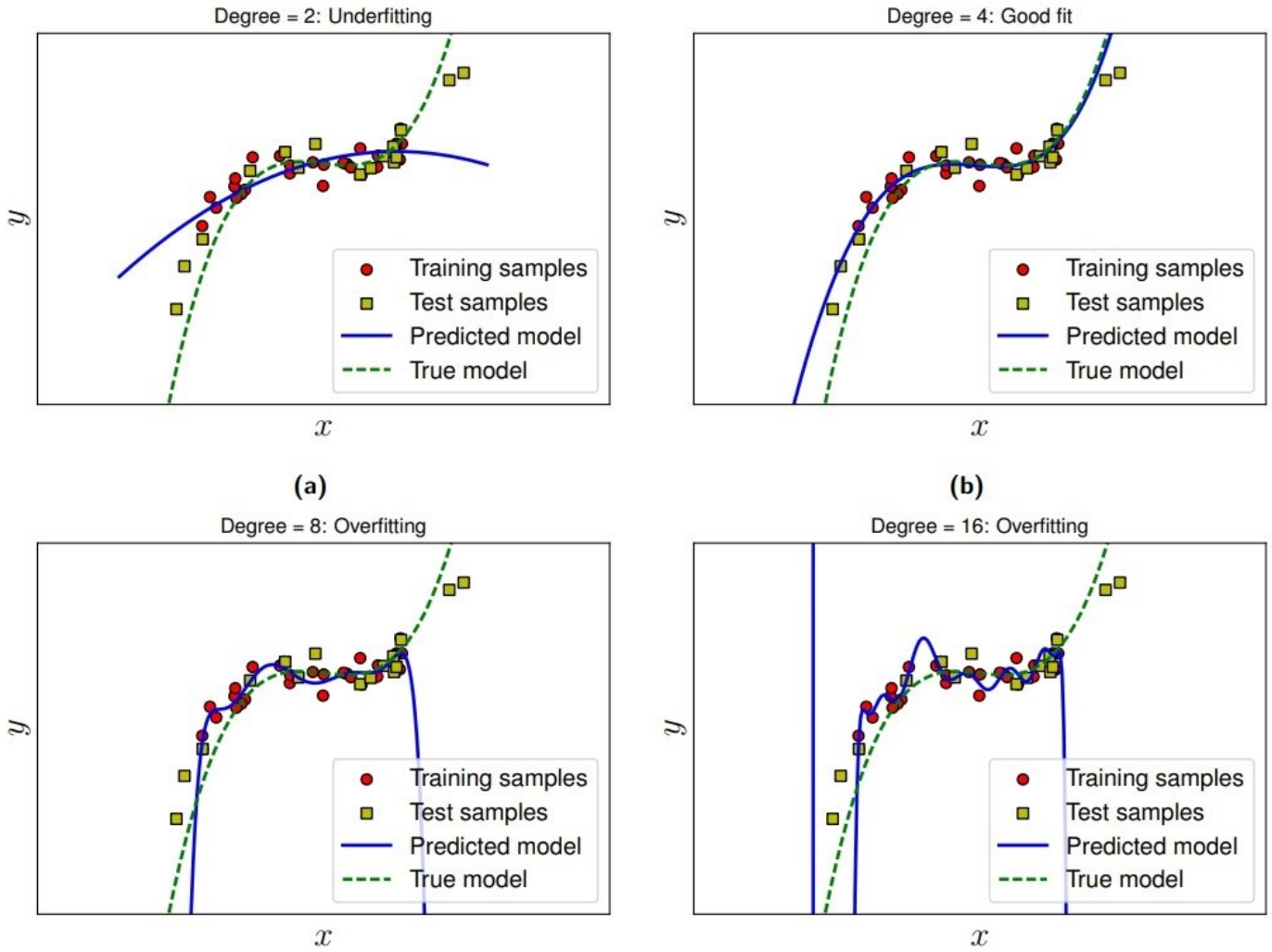
Ví dụ

- Đa thức nội suy Lagrange
  - Cho N cặp điểm dữ liệu $(x_1, y_1),(x_2, y_2), \ldots ,(x_N, y_N)$ với các x_i khác nhau đôi một
  - Luôn tìm được một đa thức P(.) bậc không vượt quá $N - 1$ sao cho $P(x_i) = y_i$, $\forall i=1, 2,\ldots, N$

- Dùng polynomial regression với vector đặc trưng là $x = [1, x, x^2, x^3,\ldots, x^d]^T$ cho đa thức bậc d

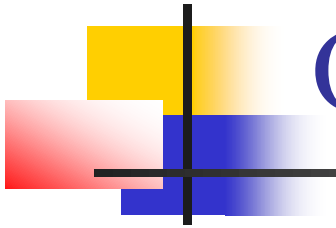
  => Cần xác định bậc d của đa thức?

Ví dụ


Degree = 2: Underfitting
Degree = 4: Good fit
Degree = 8: Overfitting
Degree = 16: Overfitting
Training samples
Test samples
Predicted model
True model
y
x
(a)
(b)

Giới thiệu

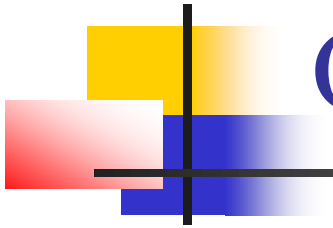
- Nếu một mô hình quá fit với dữ liệu huấn luyện, nhưng không biểu diễn tốt dữ liệu không nhìn thấy => phản tác dụng. Hiện tượng quá fit này được gọi là overfitting, cần tránh
- Một mô hình tốt là mô hình có tính tổng quát. Nghĩa là, nó biểu diễn tốt cả dữ liệu huấn luyện và dữ liệu không nhìn thấy



Overfitting

- Overfitting là hiện tượng mô hình tìm được quá khớp với dữ liệu training set, nhưng không thực sự mô tả tốt dữ liệu ngoài training set
- Việc quá chú trọng vào xấp xỉ training set làm mất tính tổng quá của mô hình

=> Nếu trong training set có nhiễu, thì mô hình tìm được không tốt



Overfitting

- Overfitting đặc biệt xảy ra khi lượng dữ liệu huấn luyện quá nhỏ hoặc độ phức tạp của mô hình quá cao
- Trong ví dụ trên, độ phức tạp của mô hình có thể được coi là bậc của đa thức cần tìm

Vậy, có những kỹ thuật nào giúp tránh overfitting?

Đánh giá chất lượng mô hình

Giả sử y là đầu ra thực sự, và $\hat{y}$ là đầu ra dự đoán bởi mô hình

Train error: Mức độ sai khác giữa đầu ra thực và đầu ra dự đoán của mô hình, thường là giá trị của hàm mất mát áp dụng lên **training set**

$$\text{train error} = \frac{1}{N_{\text{train}}} \sum_{\text{training set}} \|\mathbf{y} - \hat{\mathbf{y}}\|_p^2$$

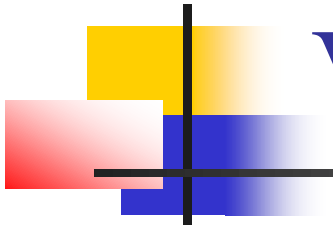
Test error: Mức độ sai khác giữa đầu ra thực và đầu ra dự đoán của mô hình, thường là giá trị của hàm mất mát áp dụng lên **test set**

$$\text{test error} = \frac{1}{N_{\text{test}}} \sum_{\text{test set}} \|\mathbf{y} - \hat{\mathbf{y}}\|_p^2$$

Đánh gia chất lượng mô hình

- Một mô hình được coi là tốt (fit) nếu cả *train error* và *test error* đều thấp
- Nếu *train error* thấp nhưng *test error* cao, ta nói mô hình bị overfitting
- Nếu *train error* cao và *test error* cao, ta nói mô hình bị underfitting
- Nếu *train error* cao nhưng *test error* thấp, hiếm khi xảy ra

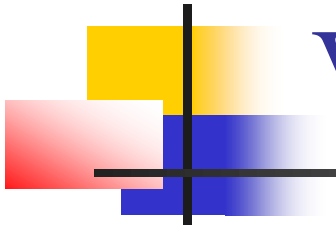
Khi xây dựng một mô hình chỉ dựa trên training set, làm thế nào để biết được chất lượng của nó trên test set?



Validation

Chia tập training set ban đầu (X) thành 2 tập:

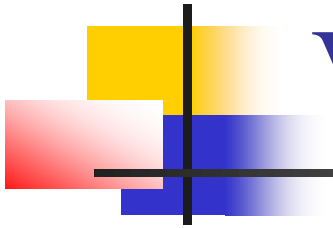
- **Validation set** (Y): Y là một tập con nhỏ của X, để thực hiện đánh giá mô hình
- **Training set mới** (Z): Z=X\Y



Validation

- **Train error** được tính trên training set mới (Z)
- **Validation error** được tính trên tập validation (Y)
- **Test error** được tính trên test set

Tìm mô hình sao cho ***cả train eror và validation error đều nhỏ***, qua đó có thể dự đoán được rằng test error cũng nhỏ



Validation

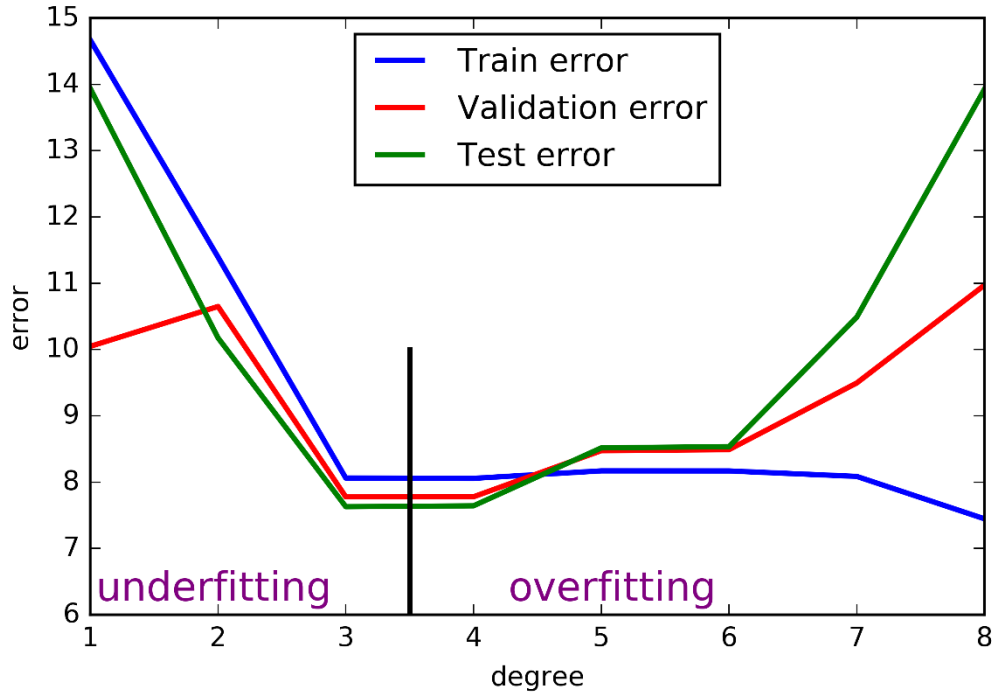
- Huấn luyện nhiều mô hình khác nhau dựa trên training set mới (Z), và tính validation error trên validation set (Y). Mô hình cho validation error nhỏ nhất sẽ là một mô hình tốt
- Bắt đầu từ mô hình đơn giản, sau đó tăng dần độ phức tạp của mô hình. Khi độ phức tạp tăng lên, training error sẽ có xu hướng nhỏ dần. Tuy nhiên, validation error ban đầu thường giảm dần và đến một lúc sẽ tăng lên do overfitting xảy ra
- Để chọn một mô hình tốt, ta quan sát validation error. Khi validation error có chiều hướng tăng lên thì ta chọn mô hình trước đó

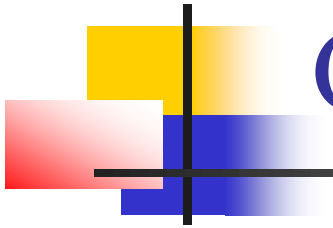
Tìm đa thức mô tả các điểm dữ liệu

Training set (X) gồm 30 điểm được chia thành 2 phần:

- Validation set (Y): 10 điểm được lấy ngẫu nhiên từ X, để tìm bậc của đa thức
- Training set mới Z: Z=X\Y, để tìm hệ số của đa thức với bậc đã biết

Mô hình mà cả *train error* và *validation error* đều nhỏ, thì *test error* cũng nhỏ





Cross-validation

Có ít dữ liệu để xây dựng mô hình

- Nếu lấy quá nhiều dữ liệu trong training set làm validation set, thì phần dữ liệu còn lại của training set không đủ để xây dựng mô hình
- Validation phải thật nhỏ, để giữ lượng dữ liệu đủ lớn cho xây dựng mô hình
- Khi tập validation quá nhỏ, hiện tượng overfitting có thể xảy ra với tập dữ liệu xây dựng mô hình

Có giải pháp nào cho tình huống này không?

Cross-validation

K-fold cross validation: Chia training set thành K tập con không giao nhau, có kích thước gần bằng nhau. Tại mỗi lần kiểm thử:

- 1 tập con được lấy ra làm validation set
- K-1 tập còn lại được dùng để xây dựng mô hình
- Mô hình cuối được xác định dựa trên trung bình của các train error và validation error (***CV(w) là nhỏ nhất***)

$$CV(\mathrm{w}) = \frac{1}{K}\sum_{i=1}^{K} E_i(\mathrm{w}) =$$

Cross-validation

Nhược điểm

- Số lượng mô hình cần huấn luyện tỉ lệ thuận với k
- Nếu trong bài toán, có nhiều tham số cần xác định, khoảng giá trị của mỗi tham số rộng, thì việc huấn luyện nhiều mô hình là khó khả thi

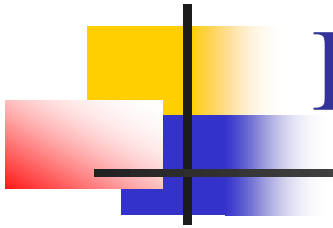
Regularization

Mục đích

- Giảm số mô hình cần huấn luyện
- Giảm độ phức tạp của mô hình, tránh overfitting nhưng giữ được tính tổng quát

Kỹ thuật

- Thay đổi mô hình một chút
- Chấp nhận hy sinh độ chính xác trong training set

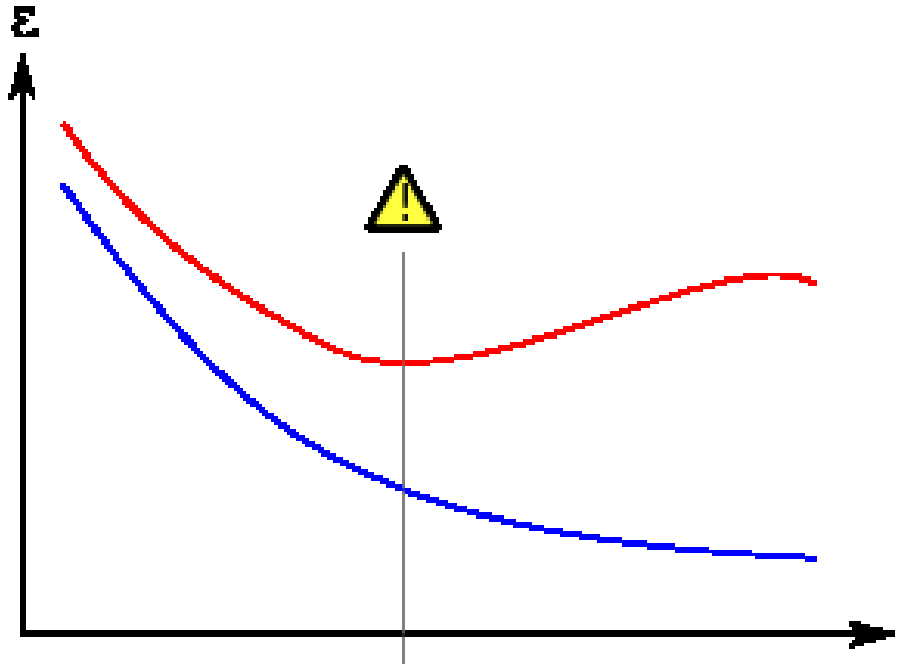


Early Stopping

- **Early Stopping**: Dừng thuật toán trước khi nó hội tụ
- Vậy dừng khi nào là phù hợp?
  - Tách từ training set ra một validation set. Trong khi huấn luyện, ***nếu training error vẫn có xu hướng giảm nhưng validation error có xu hướng tăng lên thì ta dừng thuật toán***
- Sử dụng mô hình tương ứng với điểm và *validation error* đạt giá trị nhỏ

Early Stopping

- Đường màu xanh là training error, màu đỏ là validation error
- Trục hoành (phương nằm ngang) thể hiện số lượng vòng lặp, trục tung (phương thẳng đứng) là giá trị error
- Thuật toán huấn luyện dừng tại vòng lặp mà validation error đạt giá trị nhỏ nhất



Thêm số hạng vào hàm mất mát

Thêm vào hàm mất mát một số hạng nữa:

- Số hạng này thường dùng để đánh giá độ phức tạp của mô hình
- Số hạng này càng lớn, thì mô hình càng phức tạp

Hàm mất mát mới này thường được gọi là **regularized loss function**:

$$f_{reg}(w)=f(w) + \lambda R(w)$$

- w: tham số trong mô hình
- f(w) là hàm mất mát (loss function) phụ thuộc vào training set và w
- R(w) là số hạng regularization chỉ phụ thuộc vào w
- λ: tham số regularization là số vô hướng, thường là một số dương nhỏ để đảm bảo nghiệm của bài toán tối ưu $f_{reg}(w)$ không quá xa nghiệm của bài toán tối ưu f(w)

Thêm số hạng vào hàm mất mát

Tối thiểu *regularized loss function* đồng nghĩa với việc tối thiểu cả *loss function* và số hạng *regularization*.

$$f_{reg}(w) = f(w) + \lambda R(w)$$

Tối thiểu Tối thiểu

LASSO regularization

Norm 1 là tổng các trị tuyệt đối của tất cả các phần tử. Đã chứng minh được rằng tối thiểu norm 1 sẽ dẫn tới nghiệm có nhiều phần tử bằng 0

Khi chọn

$$R(\mathbf{W}) = \|\mathbf{w}\|_1 = \sum_{i=0}^{d} |w_i|$$

Nghiệm w tìm được của bài toán tối thiểu $f_{reg}(w)$ có xu hướng rất nhiều phần tử bằng không. Hàm LASSO regression

$$\mathbf{f_{reg}}(\mathbf{w}) = \mathbf{f}(\mathbf{w}) + \lambda\|\mathbf{w}\|_1$$

- Các thành phần khác không của w tương ứng với các đặc trưng quan trọng trong dự đoán đầu ra
- Các thành phần bằng không của w tương ứng với các đặc trưng được coi là ít quan trọng

=> LASSO regression giúp lựa chọn đặc trưng hữu ích cho mô hình

Ridge regularization

Khi chọn R(w) là Norm 2

$$\mathbf{R(w)} = \|\mathbf{w}\|_\mathbf{2}^\mathbf{2}$$

Ta có hàm Ridge regularization

$$\mathbf{f_{reg}(w)} = \mathbf{f(w)} + \lambda\|\mathbf{w}\|_\mathbf{2}^\mathbf{2}$$

Hàm $\mathbf{R(w)} = \|\mathbf{w}\|_\mathbf{2}^\mathbf{2}$ khiến các hệ số trong w không quá lớn, giúp tránh việc đầu ra phụ thuộc quá nhiều vào một đặc trưng nào đó