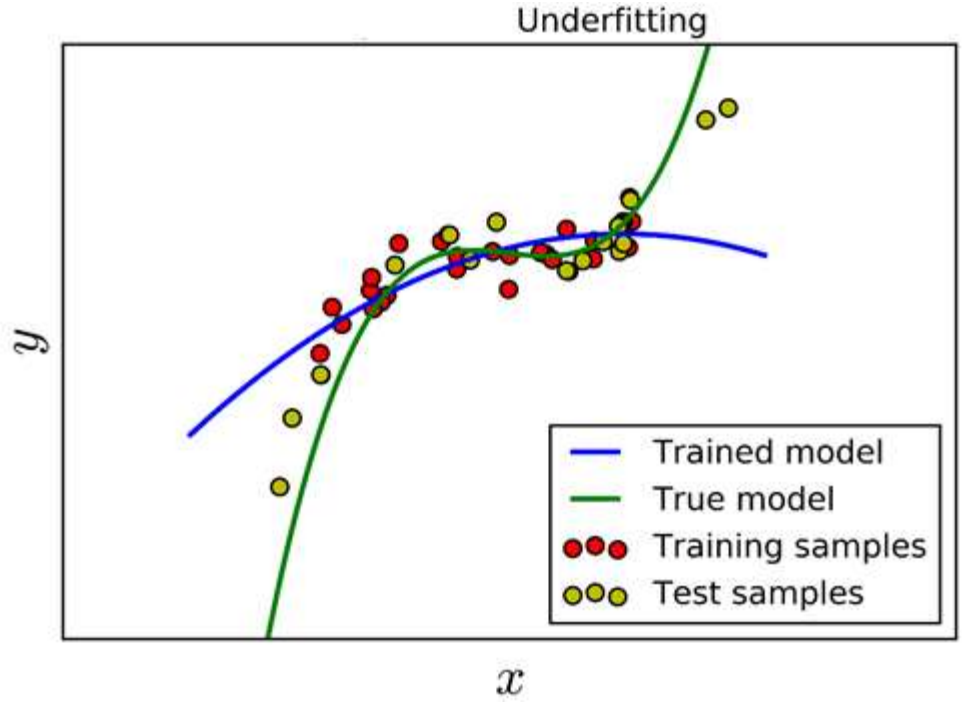
CSE: Faculty of Computer Science and Engineering
Thuyloi University

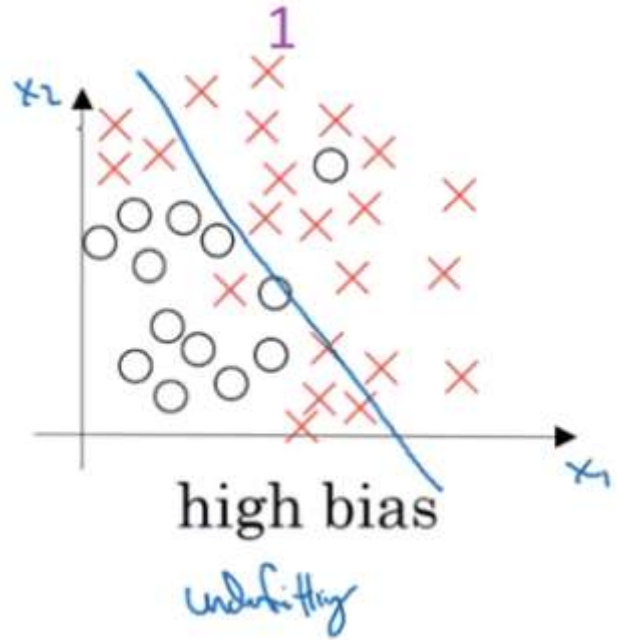
Quá khớp (Overfitting)

Trình bày: PGS.TS Nguyễn Hữu Quỳnh

Giới thiệu

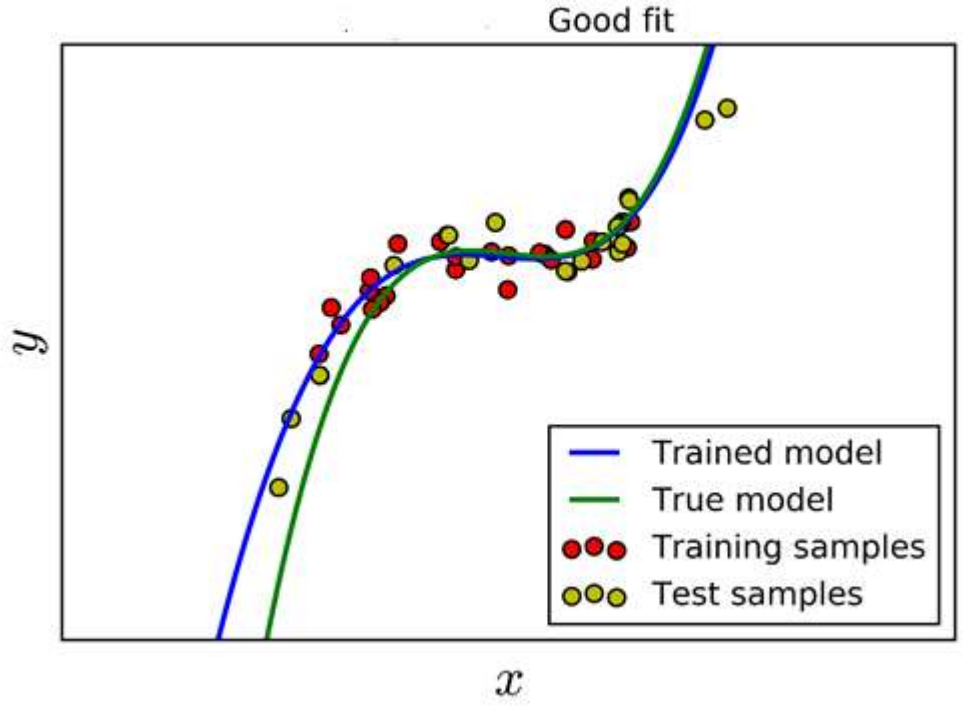
- Nếu một mô hình *quá fit* với dữ liệu thì dự đoán sẽ có sai số lớn (*overfitting)*

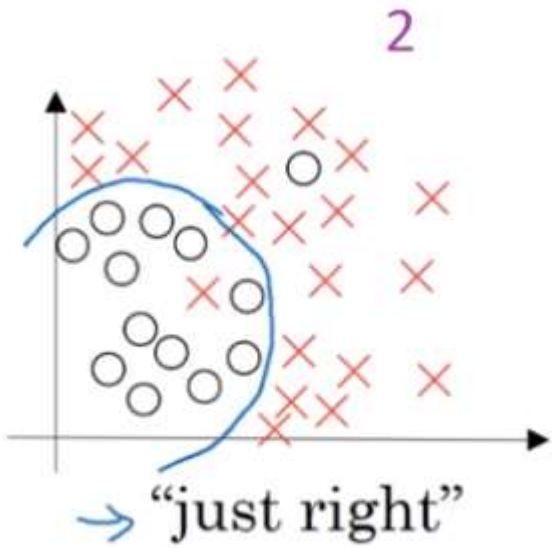




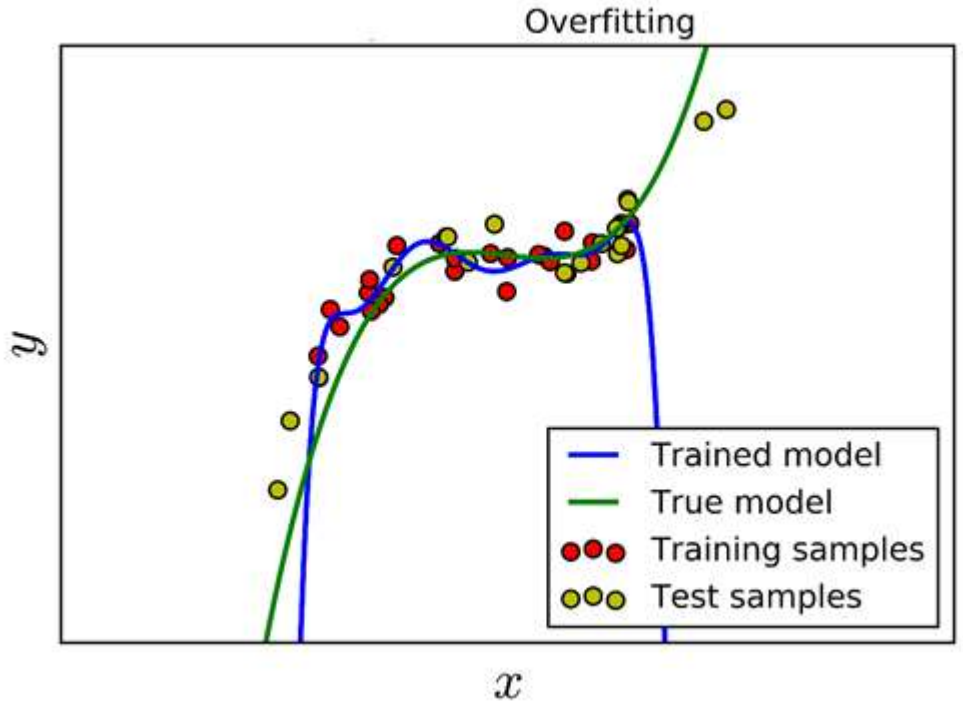
Giới thiệu

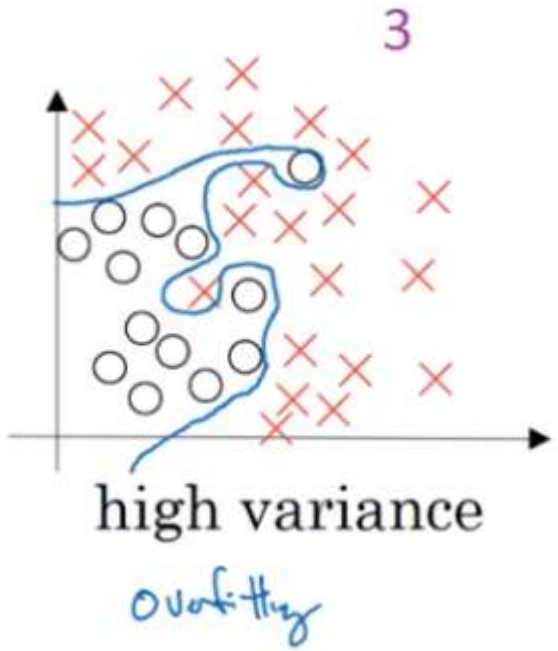
- Nếu một mô hình *quá fit* với dữ liệu thì dự đoán sẽ có sai số lớn (*overfitting)*





Giới thiệu


Overfitting
Trained model
True model
Training samples
Test samples
y
x


3
high variance
Overfitting

Giới thiệu

- Overfitting: là hiện tượng mô hình tìm được *quá khớp* với dữ liệu training nhưng không tốt trên dữ liệu dự đoán (hoặc test).
- Dữ liệu test được giả sử không được sử dụng để xây dựng các mô hình Machine Learning.

Giới thiệu

Đại lượng để đánh giá chất lượng của mô hình trên training data và test data.

Với Regression, đại lượng này thường được định nghĩa:

- **Train error:**

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

- **Test error:**

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

Giới thiệu

- 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.

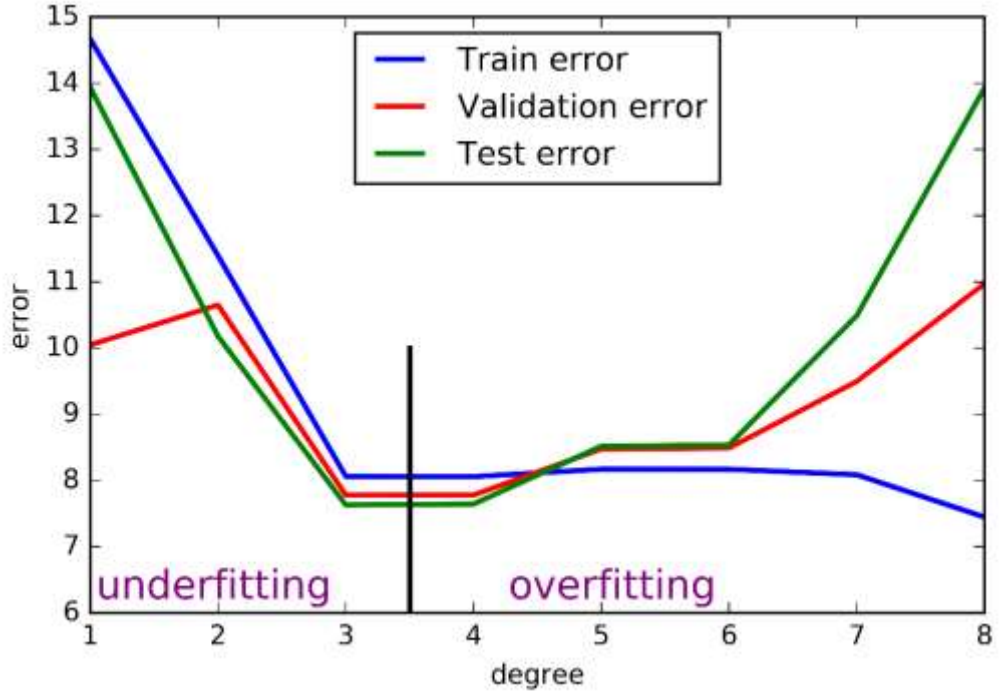
Validation

Validation

- Chúng ta thường chia tập dữ liệu ra thành hai tập nhỏ: training data và test data.
- Khi xây dựng mô hình, ta không được sử dụng test data. Vậy làm cách nào để biết được chất lượng của mô hình với unseen data?
- Cách đơn giản nhất là trích từ tập training data ra một tập con nhỏ (validation set) và thực hiện việc đánh giá mô hình trên tập con nhỏ này.
- Training set là phần còn lại của training set ban đầu.
  - Train error được tính trên training set mới này
  - validation error được tính trên tập validation.

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.



Validation

Cross-validation

- Ta thường có ít 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 sẽ không đủ để xây dựng mô hình.
  - Dẫn đến tập validation phải thật nhỏ để giữ được lượng dữ liệu cho training đủ lớn.
  - Tuy nhiên, khi tập validation quá nhỏ, hiện tượng overfitting có thể xảy ra với tập training còn lại.

Validation

- Có giải pháp nào cho tình huống này: Câu trả lời là cross-validation:
  - 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 validata 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.

Regularization

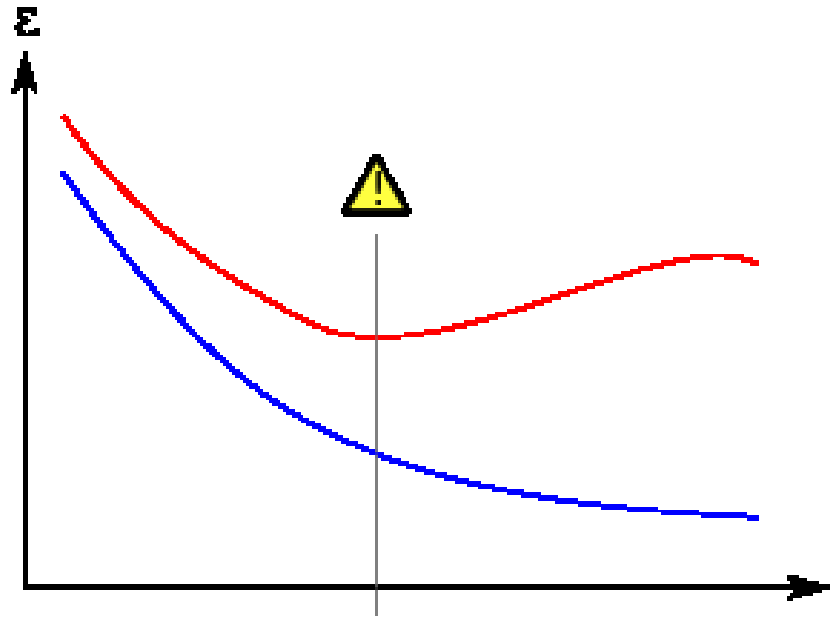
- *Regularization*: là thay đổi mô hình một chút để tránh overfitting trong khi vẫn giữ được tính tổng quát của nó (tính tổng quát là tính mô tả được nhiều dữ liệu, trong cả tập training và test).

Kỹ thuật Early Stopping

- Trong nhiều bài toán Machine Learning, chúng ta cần sử dụng các thuật toán lặp để tìm ra nghiệm, ví dụ như Gradient Descent.:
  - Nhìn chung, hàm mất mát giảm dần khi số vòng lặp tăng lên.
  - Early stopping dừng thuật toán trước khi hàm mất mát đạt giá trị quá nhỏ, giúp tránh overfitting.

Regularization

- Vậy dừng khi nào: Một kỹ thuật thường được sử dụng là:
  - Tách từ training set ra một tập validation set.
  - Sau một số vòng lặp, ta tính cả *train error* và *validation error*, đến khi *validation error* có chiều hướng tăng lên thì dừng lại,
  - Quay lại sử dụng mô hình tương ứng với điểm và *validation error* đạt giá trị nhỏ.



Regularization

Thêm số hạng vào hàm mất mát

- Kỹ thuật regularization 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**:

$$J_{\text{reg}}(\theta) = J(\theta) + \lambda R(\theta)$$

Regularization

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

$$J_{\text{reg}}(\theta) = J(\theta) + \lambda R(\theta)$$

Tối thiểu Tối thiểu

- Nghiệm của bài toán tối ưu *loss function* và **regularized loss function** là khác nhau:
  - Ta mong sự khác nhau này là nhỏ, vì vậy tham số regularization (*regularizaton parameter*) λ thường được chọn là một số nhỏ để biểu thức regularization không làm giảm quá nhiều chất lượng của nghiệm.

Regularization

l_2 regularization

- Trong kỹ thuật này:

$$R(\mathbf{w}) = \|\mathbf{w}\|_2^2$$

- Trong Xác suất thống kê, Linear Regression với l_2 regularization được gọi là **Ridge Regression**. Hàm mất mát của *Ridge Regression* có dạng:

$$J(\mathbf{w}) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{w}\|_2^2 + \lambda\|\mathbf{w}\|_2^2$$

Regularization

Regularizers for sparsity

- Ta muốn các hệ số *thực sự* bằng 0 chứ không phải là *nhỏ gần 0* như l_2 regularization đã làm phía trên. Lúc đó, có một regularization khác được sử dụng, đó là l_0 regularization:

$$R(\mathbf{W}) = \|\mathbf{w}\|_0$$

- Norm 0 của một vector là số các phần tử khác không của vector đó. Khi norm 0 nhỏ, tức rất nhiều phần tử trong vector đó bằng 0, ta nói vector đó là *sparse*.

Regularization

- Việc giải bài toán tối thiểu norm 0 nhìn chung là khó vì hàm số này không *convex*, không liên tục. Thay vào đó, norm 1 thường được sử dụng:

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

- Norm 1 là tổng các trị tuyệt đối của tất cả các phần tử.
- Người ta đã chứng minh được rằng tối thiểu norm 1 sẽ dẫn tới nghiệm có nhiều phần tử bằng 0.
- Ngoài ra, vì norm 1 là một *norm thực sự* (proper norm) nên hàm số này là *convex*, và hiển nhiên là liên tục, việc giải bài toán này dễ hơn việc giải bài toán tối thiểu norm 0.

Regularization

- Trong Thống Kê, việc sử dụng l_1 regularization còn được gọi là LASSO (Least Absolute Shrinkage and Selection Operator).