

Generalization

☑ Trắc nghiệm 29 ☑ Thực hành 3 ⬇ PDF bài

Một model có thể đúng hoàn toàn trên dữ liệu đã học mà vẫn sai trên dữ liệu mới, ví dụ khi nó chỉ lưu lại các buổi cũ rồi tra bảng. Điều ta cần là đoán tốt trên mẫu chưa thấy; khả năng đó gọi là tổng quát hóa (generalization). Bài này dùng một model đơn giản để thấy underfitting (model quá đơn giản), overfitting (model học thuộc dữ liệu train) và cách đọc learning curve để quyết định nên làm gì tiếp.

MỤC TIÊU

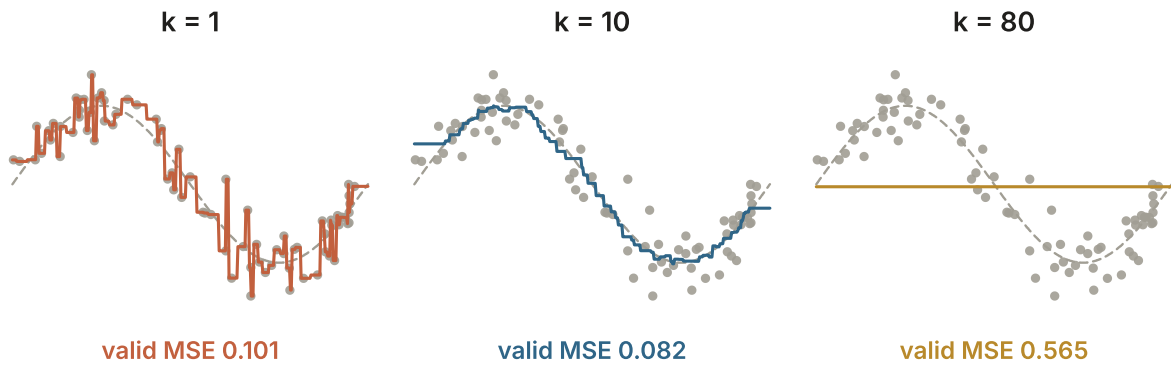
- Phân biệt lỗi trên train với lỗi trên dữ liệu mới.
- Nhận ra underfitting và overfitting.
- Đọc learning curve.
- Chọn việc làm tiếp dựa trên các dấu hiệu đó.

Nhớ một ví dụ khác với học một quy luật

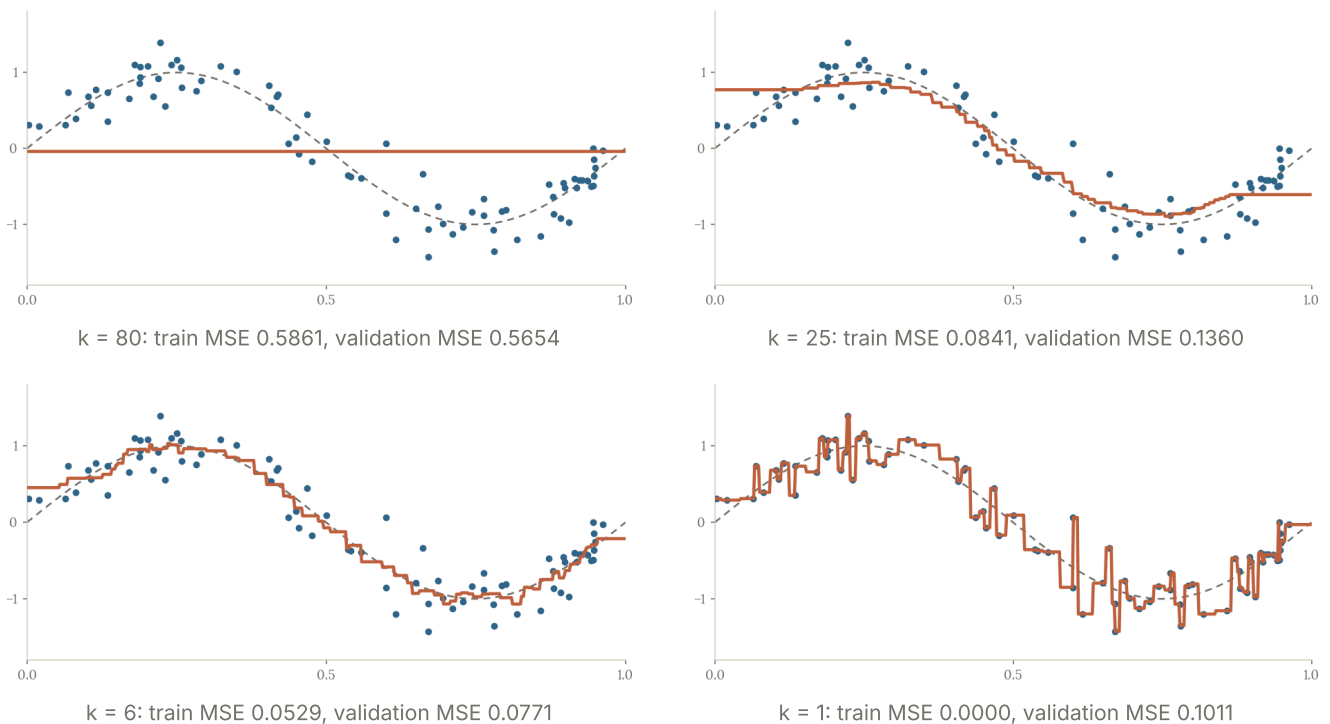
Ta tạm thu bài toán xuống một feature x trong khoảng 0–1: vị trí của một buổi trong lịch hoạt động. Mức tham gia đã chuẩn hóa là

$$y = \sin(2\pi x) + \varepsilon.$$

$\sin$ tạo một xu hướng lên rồi xuống để nhìn được bằng hình. ε là nhiễu, ở đây sinh ngẫu nhiên với trung bình 0 và độ lệch chuẩn 0.25. Các giá trị âm có nghĩa là thấp hơn mức tham gia chuẩn, không phải số người âm. Dạng sóng này do ta tự đặt ra để thí nghiệm, nhờ đó biết chính xác đâu là quy luật, đâu là nhiễu. Học được là nhờ có quy luật: nếu nhẵn hoàn toàn ngẫu nhiên như kết quả xổ số thì không mô hình nào đoán được, dù dữ liệu nhiều đến đâu.



$k = 1$ bám nhiều, $k = 80$ san phẳng, $k = 10$ vừa đủ.



kNN hồi quy trên 80 điểm train của bài khi giảm k từ 80 về 1: đường cam đi từ quá phẳng tới bám từng điểm nhiều; nét đứt là quy luật thật.

Mô hình trong hình là k -Nearest Neighbors regression, viết ngắn là kNN. Để đoán tại x , nó tìm k điểm train gần nhất rồi lấy trung bình nhân của chúng. Bài 2.4 sẽ học thuật toán kỹ hơn; hôm nay chỉ dùng đặc tính: k nhỏ cho đường linh hoạt hơn, k lớn cho đường trơn hơn.

Với $k = 1$, tại một điểm train, hàng xóm gần nhất chính là nó, nên mô hình trả đúng nhãn đã quan sát. Điều ấy bao gồm cả phần nhiễu của riêng điểm đó. Với k bằng toàn bộ số mẫu train, mọi vị trí nhận cùng một giá trị trung bình, dù xu hướng thật lên xuống rõ rệt.

HỎI NHANH

Mô hình đoán đúng mọi nhãn trên tập train. Điều đó chứng minh được gì?

- A Nó khớp các mẫu đã học, còn mẫu mới thì chưa biết
- B Nó chắc chắn đã tìm ra quy luật thật
- C Dữ liệu train chắc chắn không có nhãn nhiễu
- D Nó sẽ đúng ít nhất 90% trên dữ liệu mới

Viết rõ đang đo lỗi ở đâu

Giả sử có n mẫu train. Sai số train với MSE là

$$\hat{R}_{\text{train}}(f) = \frac{1}{n} \sum_{i=1}^n (f(x_i) - y_i)^2.$$

Ví dụ ba mẫu có nhãn thật (1, 3, 5) và dự đoán (2, 3, 4) cho $\text{MSE} = (1 + 0 + 1)/3 = 2/3$.

Nhưng điều ta thật sự cần là lỗi trên các mẫu **mới**, tức trung bình lỗi nếu đem model dùng cho rất nhiều buổi sau này:

$$R(f) = \mathbb{E}[(f(x) - y)^2].$$

$\mathbb{E}$ là kỳ vọng (bài 0.7), lấy trên mọi mẫu có thể gặp lúc dùng thật. Ta không có các mẫu đó, nên dùng tập validation để ước lượng $R(f)$. Muốn ước lượng đúng, validation phải giống dữ liệu lúc dùng thật; chia sai thì điểm có thể đẹp mà không nói lên điều ta cần. Khả năng đoán tốt trên mẫu chưa gặp lúc học gọi là tổng quát hóa, hay khái quát hóa (generalization); đó là thứ ta cần, không phải điểm trên train.

Trong thí nghiệm này, train và validation được sinh riêng từ cùng một công thức, nên validation giống hệt dữ liệu mới. Code tạo 80 mẫu train và 160 mẫu validation, rồi so ba giá trị k .

Python

```
import numpy as np
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_squared_error

rng = np.random.default_rng(12)

def make_data(n):
    X = rng.uniform(0, 1, size=(n, 1))
    y = np.sin(2 * np.pi * X[:, 0]) + rng.normal(0, 0.25, size=n)
    return X, y

X_train, y_train = make_data(80)
X_valid, y_valid = make_data(160)
print(X_train.shape, y_train.shape)
for k in [80, 10, 1]:
    model = KNeighborsRegressor(n_neighbors=k).fit(X_train, y_train)
    train_mse = mean_squared_error(y_train, model.predict(X_train))
    valid_mse = mean_squared_error(y_valid, model.predict(X_valid))
    print(f"k={k:2d}: train={train_mse:.4f}, valid={valid_mse:.4f}")
```

KẾT QUẢ

```
(80, 1) (80,)
k=80: train=0.5861, valid=0.5654
k=10: train=0.0574, valid=0.0820
k= 1: train=0.0000, valid=0.1011
```

Đọc cả hai cột. Khi $k = 80$, train và validation cùng có lỗi lớn: mô hình quá đơn giản để giữ xu hướng. Khi $k = 1$, train có lỗi bằng 0, nhưng validation không vì thế thấp nhất. $k = 10$ lấy trung bình cục bộ, giảm tác động của nhiễu mà còn giữ được dạng sóng.

Không chọn mô hình từ cột train. Cũng không biến ví dụ thành quy tắc “luôn dùng $k = 10$ ”: số mẫu, mật độ điểm, độ nhiễu và số chiều thay đổi thì k phù hợp cũng đổi.

HỎI NHANH

Phương án A có MSE 0.01 trên train và 0.30 trên validation. Phương án B có MSE 0.08 trên train và 0.12 trên validation. Chọn theo validation thì ta chọn phương án nào?

- A A, vì MSE trên train nhỏ hơn
- B A, vì chênh lệch lớn cho thấy nó học được nhiều hơn
- C B, vì MSE trên validation nhỏ hơn
- D Chưa chọn được, phải xem điểm test của cả hai

Underfit và overfit là chẩn đoán có điều kiện

Underfitting (học chưa đủ): model quá đơn giản, không bắt được quy luật. Ví dụ $k = 80$ san phẳng cả đường cong. Dấu hiệu: lỗi train và lỗi validation **cùng cao** và gần nhau. Thêm dữ liệu cùng loại không giúp, vì model còn chưa khớp nổi train; cần model linh hoạt hơn hoặc thêm đặc trưng tốt hơn.

Overfitting (học quá sát): model học thuộc cả nhiễu của train. Ví dụ $k = 1$: đổi một nhãn nhiều là cả một đoạn dự đoán đổi theo. Dấu hiệu: lỗi train **thấp** mà lỗi validation **cao hơn hẳn**. Cách chữa: giảm độ linh hoạt (ở đây là tăng k), dùng regularization (chương 2), hoặc thêm dữ liệu.

Overfitting dễ xảy ra khi model có rất nhiều tham số mà dữ liệu ít, hay số đặc trưng nhiều hơn số mẫu, như 100 chỉ số xét nghiệm cho 50 bệnh nhân. Một mạng neural đủ lớn còn học thuộc được cả nhãn bị xáo ngẫu nhiên, đúng 100% trên train; vì vậy điểm train cao không chứng minh model đã học được quy luật.

Tuy vậy, hai dấu hiệu trên chỉ là gợi ý. Train tốt mà validation kém cũng có thể do dữ liệu đổi theo thời gian, một cột bị sai đơn vị, hay hai tập được xử lý khác nhau. Train và validation cùng kém cũng có thể do nhãn bị lệch hàng hay metric viết sai. Trước khi sửa model, nên kiểm lại dữ liệu và tính tay một ví dụ nhỏ.

HỎI NHANH

Một mô hình phân loại review phim là khen hay chê, có 2.4 triệu tham số, học trên 5000 review. Nó đoán đúng 98.5% trên train, 72.3% trên validation và 71.8% trên test. Số liệu này chỉ ra vấn đề gì?

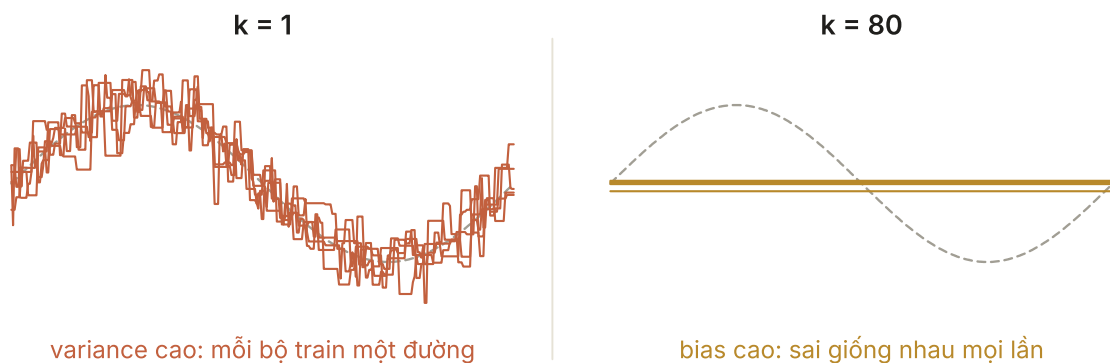
- A Lỗi dữ liệu, vì validation và test gần bằng nhau
- B Underfitting, vì 72% là thấp với bài này
- C Overfitting, vì train gần hoàn hảo còn validation kém xa
- D Dữ liệu thay đổi, vì train cao còn test thấp

ĐỀ THI NOAI 2026 Singapore · Vòng sơ loại · Câu 272

BIAS VÀ VARIANCE Ở MỨC TRỰC GIÁC

Bias là phần sai do model quá cứng nhắc, sai cùng một kiểu dù học từ bộ train nào, như $k = 80$ lần nào cũng đoán một đường phẳng. **Variance** là mức model thay đổi khi đổi sang bộ train khác, như $k = 1$ đổi theo từng điểm nhiễu. Model linh hoạt hơn thường giảm bias nhưng tăng variance. Hình dưới học năm lần trên năm bộ train khác nhau để thấy điều đó.

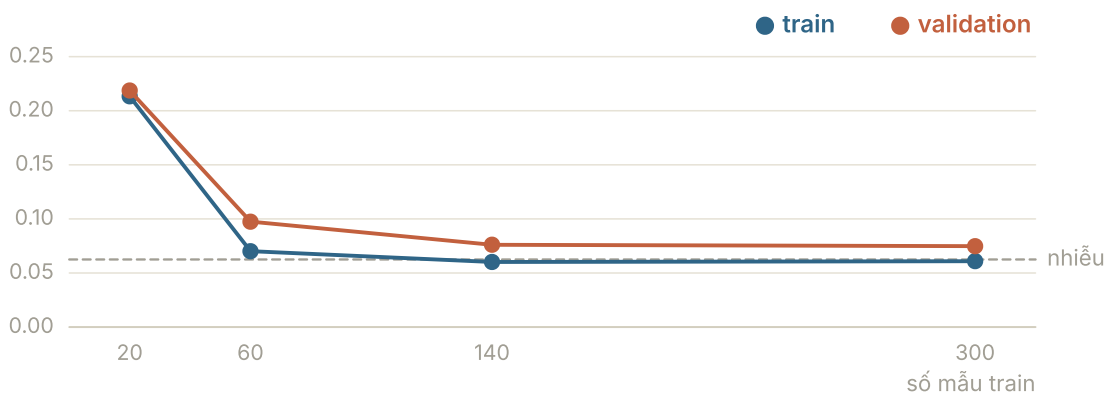
Với MSE, lỗi trên mẫu mới tách được thành ba phần: $\mathbb{E}[(y - \hat{f}(x))^2] = \sigma^2 + \text{bias}^2 + \text{variance}$. Ở đây bias là độ lệch giữa dự đoán trung bình (lấy trên nhiều bộ train) và giá trị thật; variance là phương sai của dự đoán quanh trung bình ấy; σ^2 là phương sai của nhiễu, phần không model nào giảm được (irreducible error). Bias và variance thường đối ngược chiều nhau khi chỉnh độ phức tạp, nhưng không phải luôn luôn: đoán bừa thì cả hai đều cao.



Năm bộ train: $k = 1$ đổi theo từng bộ, $k = 80$ lần nào cũng sai như nhau.

Learning curve: thêm dữ liệu có đáng làm không?

Câu hỏi thực tế: có nên thu thêm dữ liệu không? **Learning curve** trả lời câu đó: giữ $k = 10$, tăng dần số mẫu train, và đo lỗi train cùng lỗi validation ở mỗi mức. Trục ngang là **số mẫu train**. (Đừng nhầm với đường loss theo số bước huấn luyện, sẽ gặp ở chương 3.)



Thêm mẫu, hai đường lỗi gần nhau và sát mức nhiều.

Hàm `learning_curve` của `scikit-learn` làm việc này. Ở mỗi kích thước, nó chia dữ liệu thành 4 phần (gọi là 4 fold), lần lượt giữ một phần để chấm và học trên phần còn lại, rồi lấy trung bình; bài 1.3 sẽ học kỹ cách chia này.

Python

```
from sklearn.model_selection import KFold, learning_curve

X_curve, y_curve = make_data(400)
cv = KFold(n_splits=4, shuffle=True, random_state=12)
```

```

sizes, train_scores, valid_scores = learning_curve(
    KNeighborsRegressor(n_neighbors=10),
    X_curve, y_curve, cv=cv,
    train_sizes=[20, 60, 140, 300],
    scoring="neg_mean_squared_error",
    shuffle=True, random_state=12, n_jobs=1,
)
for size, train_score, valid_score in zip(
    sizes, train_scores.mean(axis=1), valid_scores.mean(axis=1)
):
    print(f"n={size:3d}: train MSE={-train_score:.4f}, valid MSE={-valid_score:.4f}")

```

KẾT QUẢ

```

n= 20: train MSE=0.2133, valid MSE=0.2187
n= 60: train MSE=0.0701, valid MSE=0.0974
n=140: train MSE=0.0602, valid MSE=0.0761
n=300: train MSE=0.0610, valid MSE=0.0748

```

Các score có dấu âm vì công cụ chọn mô hình của scikit-learn quy ước “lớn hơn là tốt hơn”, nên MSE được đổi dấu thành neg-MSE; ta đổi dấu lại để đọc. Mỗi hàng là trung bình của bốn fold.

Với 20 mẫu, mười hàng xóm phủ một vùng rộng nên đường dự đoán bị san phẳng, cả hai lỗi đều cao. Thêm mẫu thì vùng ấy hẹp lại, đường giữ được dạng sóng, và lỗi validation giảm. Từ 140 lên 300 mẫu, lỗi validation gần như không giảm nữa.

Cách đọc một learning curve:

- **Hai đường sát nhau và cùng đi ngang ở mức lỗi cao:** underfitting. Thêm dữ liệu không giúp; cần model mạnh hơn hoặc đặc trưng tốt hơn.
- **Hai đường cách xa, đường validation còn đang giảm:** thêm dữ liệu còn có ích.
- **Hai đường cách xa, đường validation đã đi ngang:** overfitting; nên giảm độ phức tạp hoặc thêm regularization.

Đổi tỉ lệ chia, chẳng hạn từ 70% train và 30% test sang 90% và 10%, cũng theo cách đọc này: train nhiều hơn thường giúp nhưng không chắc, còn tập chấm nhỏ đi thì con số đo được dao động nhiều hơn.

Còn một mốc để so: kể cả khi biết chính xác $\sin(2\pi x)$, lỗi vẫn còn bằng phương sai của nhiễu, $0.25^2 = 0.0625$, vì không model nào đoán được nhiễu. Validation MSE 0.0748 ở

300 mẫu đã khá gần mốc ấy, nên không còn cải thiện được bao nhiêu. Với dữ liệu thật ta không biết mức nhiễu; người ta thường lấy kết quả tốt nhất mà con người làm được, như một nhóm bác sĩ hội chẩn, làm ước lượng cho mức lỗi thấp nhất có thể, gọi là sai số Bayes.

HỎI NHANH

Tăng train từ 200 lên 800 rồi 2000 mẫu. MSE trên train lần lượt là 0.42, 0.43, 0.43. MSE trên validation là 0.46, 0.45, 0.45. Cách chia và metric đều đã kiểm. Ta nên thử gì trước?

- A Giữ nguyên mọi thứ, chỉ thu thêm mẫu
- B Thêm feature hoặc dùng mô hình mạnh hơn
- C Giảm độ phức tạp để thu hẹp khoảng cách
- D Đổi seed chia dữ liệu tới khi validation thấp hơn

PHÒNG THEO CMU 10-601 · Practice Problems, Exam 1 · Q3.1

Trở lại câu chuyện câu lạc bộ

Model học trên các buổi trong năm học rồi đem dùng vào dịp hè có thể đoán kém, vì số đăng ký, lịch học và thói quen tham gia đều đổi. Hiện tượng dữ liệu lúc dùng khác dữ liệu lúc học gọi là **distribution shift** (thay đổi phân phối). Model chỉ đáng tin trong vùng dữ liệu đã học. Vài ví dụ:

- Học giá nhà 40–120 m² rồi đoán căn 600 m²: đoán ra ngoài vùng đã học (ngoại suy).
- Nhận diện khuôn mặt học trên người lớn, dùng cho trẻ em.
- Học trên ảnh chụp ban ngày, dùng cho ảnh ban đêm.
- Học trên dữ liệu năm 2023, dùng cho năm 2025.

Cách sửa gốc là đưa đúng loại dữ liệu sẽ gặp lúc dùng vào train, và chấm trên dữ liệu gần thời điểm dùng nhất. Vì vậy “model tổng quát hóa tốt” luôn phải nói rõ: tốt cho trường hợp nào. Đoán các buổi sau của cùng câu lạc bộ khác với đoán cho câu lạc bộ ở một trường mới; bài 1.3 sẽ chia dữ liệu theo thời gian và theo nhóm để đo từng trường hợp.

HỎI NHANH

Ta cần dự báo cho tháng sau. Chia dữ liệu ngẫu nhiên thì MAE là 2. Giữ riêng tháng cuối làm validation thì MAE là 5. Nên tin con số nào?

- A Bỏ tháng cuối khỏi dữ liệu cho phép đánh giá ổn hơn
- B Tin số 2, vì MAE thấp hơn là tốt hơn
- C Lấy trung bình 3.5 làm MAE cuối cùng
- D Tin số 5, rồi tìm hiểu vì sao tháng cuối khó hơn

Khi đi thi

- Ghi cả lỗi train lẫn lỗi validation cho mỗi lần thử; một cột không đủ để chẩn đoán.
- Train tốt, validation kém: giảm độ linh hoạt hoặc thêm regularization. Cả hai cùng kém: thêm feature hoặc dùng model mạnh hơn.
- Chia validation giống lúc dùng thật: đoán tương lai thì giữ riêng đoạn thời gian cuối.

Tóm tắt

- Generalization là đoán tốt trên mẫu mới, trong đúng trường hợp sẽ dùng model.
- Lỗi train thấp chưa đủ; phải xem lỗi validation, và validation phải chia giống lúc dùng thật.

- Cả hai lỗi cùng cao: underfitting. Train thấp, validation cao hẳn: overfitting. Nhưng kiểm dữ liệu trước khi sửa model.
- Learning curve vẽ lỗi theo số mẫu train, cho biết thu thêm dữ liệu có đáng không.

Kiểm tra cuối bài

CÂU 1

Một mô hình khớp train rất tốt. Bằng chứng nào cho thấy nó cũng sẽ đoán tốt khi đem dùng thật?

- A File mô hình nhỏ và huấn luyện nhanh
- B Lỗi thấp trên dữ liệu mới, chưa dùng để học hay để chọn
- C Lỗi train còn giảm tiếp khi tăng số tham số
- D Chạy lại với cùng seed thì dự đoán trên train không đổi

CÂU 2

kNN regression dùng trọng số đều, có 80 mẫu train, và ta chọn $k = 80$. Hai điểm cần đoán khác nhau sẽ nhận dự đoán thế nào?

- A Cả hai đều bằng trung bình của 80 nhãn train
- B Khác nhau, vì khoảng cách tới 80 mẫu khác nhau
- C Cả hai đều bằng median của 80 nhãn train
- D Mỗi điểm nhận nhãn của mẫu gần nó nhất

CÂU 3

Bài phân loại văn bản 10 lớp, có 5000 mẫu. Ba mô hình cho tỉ lệ đúng trên train và validation như sau. A là một mô hình rất đơn giản: 68% và 67%. B là một mô hình rất lớn, 2 triệu tham số: 99% và 71%. C là mô hình vừa phải, có regularization: 85% và 83%. Chẩn đoán nào đúng?

- A A underfit, cần mô hình mạnh hơn. B overfit, cần regularization. C cân bằng.
- B A khớp tốt, nên giữ. B bias cao, cần train lâu hơn. C variance cao, nên bớt regularization.
- C A variance cao, cần thêm dữ liệu. B khớp tốt, đem dùng được. C bias cao, cần tăng sức chứa.
- D Cả ba đều do dữ liệu bẩn, phải làm sạch trước khi chỉnh mô hình.

ĐỀ THI NOAI 2026 Singapore · Vòng sơ loại · Câu 277

CÂU 4

kNN regression với $k = 1$, chấm trên chính tập train. Khi nào thì lỗi train chắc chắn bằng 0?

- A Lúc nào cũng vậy, vì mỗi điểm là hàng xóm gần nhất của chính nó
- B Khi không có hai điểm train trùng input mà khác nhãn
- C Khi nhiễu nhỏ
- D Khi đã chuẩn hóa feature

CÂU 5

Một hàm của scikit-learn báo điểm `neg_mean_squared_error` bằng -0.12 . MSE bằng bao nhiêu?

- A 0.12
- B -0.12
- C 8.33
- D Không đổi được

CÂU 6

Trên cùng một cách chia, kNN cho MSE trên train và validation như sau. $k = 1$: 0 và 0.48. $k = 5$: 0.07 và 0.21. $k = 25$: 0.25 và 0.29. Chỉ chọn trong ba giá trị này thì chọn k nào?

- A $k = 1$, vì lỗi train bằng 0
- B $k = 25$, vì train và validation gần nhau nhất
- C Lấy trung bình ba giá trị, tức khoảng $k = 10$
- D $k = 5$, vì MSE trên validation nhỏ nhất

PHÒNG THEO CMU 10-601 · Practice Problems, Exam 1 · Q3.2

CÂU 7

Khi tăng độ phức tạp của mô hình, chẳng hạn tăng bậc đa thức trong hồi quy, bias và variance thường thay đổi thế nào?

- A Cả hai cùng tăng
- B Cả hai cùng giảm
- C Cả hai không đổi
- D Bias giảm, variance tăng
- E Bias tăng, variance giảm

ĐỀ THI [USA-NA-AIO 2026 · Vòng 1 · Part 1.3](#)

CÂU 8

Giữ nguyên validation, tăng train từ 100 lên 400 rồi 1600 mẫu. MSE trên train là 0.03, 0.07, 0.10. MSE trên validation là 0.50, 0.30, 0.18. Nếu còn kinh phí, ta nên làm gì?

- A Giảm train về 100 mẫu để MSE train xuống 0.03
- B Thu thêm mẫu cùng loại, tiếp tục theo dõi validation
- C Kết luận thêm dữ liệu có hại, vì MSE train tăng
- D Chọn lại validation cho hai lỗi gần nhau nhất

PHÒNG THEO [CMU 10-601 · Practice Problems, Exam 1 · Q3.1](#)

CÂU 9

Một đội chia ngẫu nhiên train và validation 50 lần, rồi chỉ báo MSE nhỏ nhất trong 50 lần ấy. Làm vậy sai ở đâu?

- A Số mẫu độc lập bị tăng lên 50 lần
- B Không sai, thử 50 lần thì số nhỏ nhất là công bằng
- C Đổi seed nên không chạy lại ra cùng kết quả được
- D Chọn lần chia may nhất, nên con số quá lạc quan

CÂU 10

Mô hình đạt điểm tốt khi chia ngẫu nhiên từng hàng. Có chắc nó cũng đoán tốt cho một trường chưa có trong dữ liệu không?

- A Có, nếu train đủ lớn và đa dạng
- B Có, nếu MSE train bằng 0 ở mọi trường
- C Không, phải để riêng trọn vài trường ra mà kiểm
- D Có, nếu chia nhiều lần hơn

Đọc thêm

- [Dive into Deep Learning – Generalization](#). Đọc để nối empirical risk, nhiễu và sự khác nhau giữa fitting với generalization.
- James và cộng sự – [An Introduction to Statistical Learning](#), chương 2 và 5. Đọc trực giác bias-variance và cách ước lượng lỗi dự đoán.
- [scikit-learn – Validation curves và learning curves](#). Xem đúng trục của từng loại đường và quy ước score.