Nhập môn Học máy và Khai phá dữ liệu

(IT3190)

Nguyễn Nhật Quang

quang.nguyennhat@hust.edu.vn

Trường Đại học Bách Khoa Hà Nội

Viện Công nghệ thông tin và truyền thông

Năm học 2019-2020

Nội dung môn học:

- Giới thiệu về Học máy và Khai phá dữ liệu
- Tiền xử lý dữ liệu
- **Đánh giá hiệu năng của hệ thống**
- Hồi quy
- Phân cụm
- Phân loại
- Phát hiện luật kết hợp

Đánh giá hiệu năng của hệ thống (1)

- Việc đánh giá hiệu năng của hệ thống học máy (hoặc khai phá dữ liệu) thường được thực hiện dựa trên thực nghiệm (**experimentally**), hơn là dựa trên phân tích (analytically)
  - Các đánh giá phân tích (analytical evaluation) nhằm chứng minh một hệ thống là đúng đắn (correct) và hoàn chỉnh (complete) (vd: các bộ chứng minh định lý trong Logics)
  - *Không thể xây dựng một đặc tả (định nghĩa) hình thức của vấn đề mà một hệ thống học máy (hoặc khai phá dữ liệu) giải quyết* (Đối với bài toán học máy hoặc khai phá dữ liệu, thì tính đúng đắn và tính hoàn chỉnh là gì?)

Đánh giá hiệu năng của hệ thống (2)

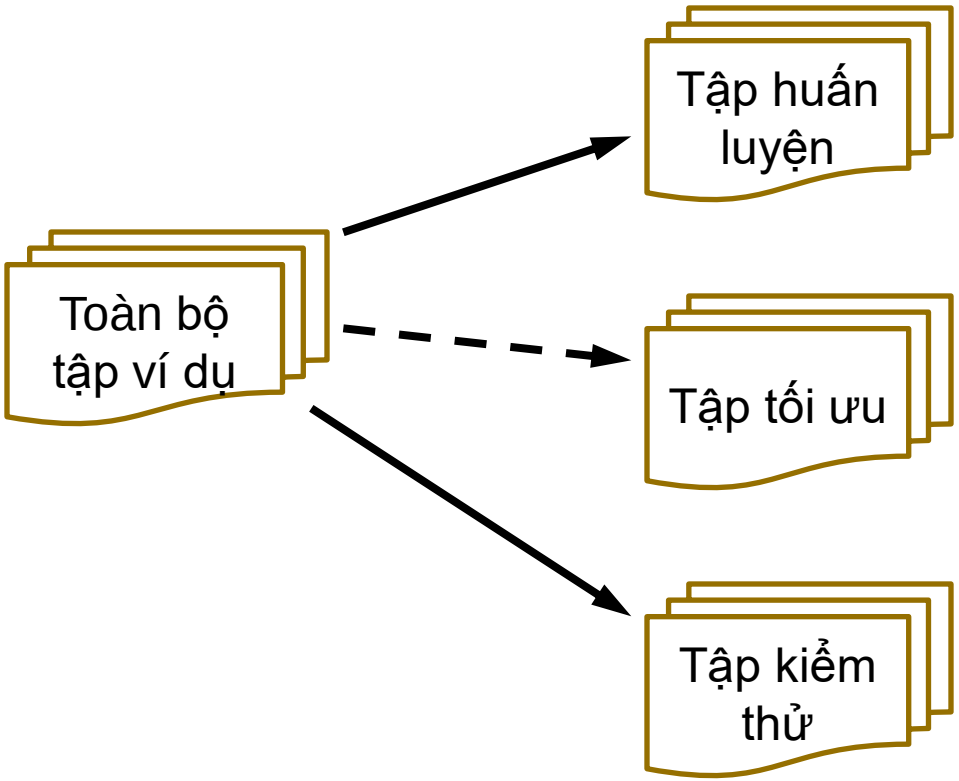
- Tập trung vào việc đánh giá hiệu năng của hệ thống
  - *Thực hiện một cách tự động bởi hệ thống*, sử dụng một tập các ví dụ (tập thử nghiệm – test set)
  - Không cần sự tham gia (can thiệp) của người dùng
- Các **phương pháp** đánh giá (evaluation methods)

  → Làm sao có được một đánh giá đáng tin cậy về hiệu năng của hệ thống?
- Các **tiêu chí** đánh giá (evaluation metrics)

  → Làm sao để đo (tính toán) hiệu năng của hệ thống?

  → Khác nhau đối với các kiểu bài toán (hồi quy, phân cụm, phân loại, phát hiện luật kết hợp)

Các phương pháp đánh giá (1)



Được dùng để huấn luyện hệ thống

Tùy chọn; và được dùng để tối ưu các tham số của hệ thống

Được dùng để đánh giá hệ thống đã (sau khi) được huấn luyện

Các phương pháp đánh giá (2)

- Làm thế nào để thu được một đánh giá đáng tin cậy về hiệu năng của hệ thống?
  - Tập huấn luyện càng lớn, thì hiệu năng của hệ thống càng tốt
  - Tập kiểm thử càng lớn, thì việc đánh giá càng chính xác
  - Vấn đề: Rất khó (ít khi) có thể có được các tập dữ liệu (rất) lớn
- *Hiệu năng của hệ thống không chỉ phụ thuộc vào giải thuật* được sử dụng, mà còn phụ thuộc vào:
  - Phân bố lớp (Class distribution)
  - Chi phí của việc phân lớp sai (Cost of misclassification)
  - Kích thước của tập huấn luyện (Size of the training set)
  - Kích thước của tập kiểm thử (Size of the test set)

Các phương pháp đánh giá (3)

- Hold-out
- Stratified sampling
- Repeated hold-out
- Cross-validation
  - *k*-fold
  - Leave-one-out
- Bootstrap sampling

Hold-out (Splitting)

- Toàn bộ tập ví dụ D được chia thành 2 tập con **không giao nhau**
  - Tập huấn luyện D_train – để huấn luyện hệ thống
  - Tập kiểm thử D_test – để đánh giá hiệu năng của hệ thống

  → $D = D\_train \cup D\_test$, và thường là $|D\_train| >> |D\_test|$
- Các yêu cầu:
  - Bất kỳ ví dụ nào thuộc vào tập kiểm thử D_test đều không được sử dụng trong quá trình huấn luyện hệ thống
  - Bất kỳ ví dụ nào được sử dụng trong giai đoạn huấn luyện hệ thống (i.e., thuộc vào D_train) đều không được sử dụng trong giai đoạn đánh giá hệ thống
  - Các ví dụ kiểm thử trong D_test cho phép một đánh giá không thiên vị đối với hiệu năng của hệ thống
- Các lựa chọn thường gặp: $|D\_train|=(2/3).|D|$, $|D\_test|=(1/3).|D|$
- **Phù hợp khi ta có tập ví dụ D có kích thước lớn**

Stratified sampling

- Đối với các tập ví dụ có kích thước nhỏ hoặc không cân xứng (unbalanced datasets), các ví dụ trong tập huấn luyện và thử nghiệm có thể không phải là đại diện
- Ví dụ: Có (rất) ít, hoặc không có, các ví dụ đối với một số lớp
- Mục tiêu: Phân bố lớp (class distribution) trong tập huấn luyện và tập kiểm thử phải xấp xỉ như trong tập toàn bộ các ví dụ (D)
- Lấy mẫu phân tầng (Stratified sampling)
  - Là một phương pháp để cân xứng (về phân bố lớp)
  - Đảm bảo tỷ lệ phân bố lớp (tỷ lệ các ví dụ giữa các lớp) trong tập huấn luyện và tập kiểm thử là xấp xỉ nhau
- Phương pháp lấy mẫu phân tầng không áp dụng được cho bài toán hồi quy (vì giá trị đầu ra của hệ thống là một giá trị số, không phải là một nhãn lớp)

Repeated hold-out

- Áp dụng phương pháp đánh giá Hold-out nhiều lần, để sinh ra (sử dụng) các tập huấn luyện và thử nghiệm khác nhau
  - Trong mỗi bước lặp, một tỷ lệ nhất định của tập D ***được lựa chọn ngẫu nhiên*** để tạo nên tập huấn luyện (có thể sử dụng kết hợp với phương pháp lấy mẫu phân tầng – stratified sampling)
  - Các giá trị lỗi (hoặc các giá trị đối với các tiêu chí đánh giá khác) thu được trong các bước lặp này được *lấy trung bình cộng (averaged)* để xác định giá trị lỗi tổng thể
- Phương pháp này vẫn không hoàn hảo
  - Mỗi bước lặp sử dụng một tập kiểm thử khác nhau
  - Có một số ví dụ trùng lặp (được sử dụng lại nhiều lần) trong các tập kiểm thử này

Cross-validation

- Để tránh việc trùng lặp giữa các tập kiểm thử (một số ví dụ cùng xuất hiện trong các tập kiểm thử khác nhau)
- k-fold cross-validation
  - Tập toàn bộ các ví dụ D được chia thành k tập con **không giao nhau** (gọi là "*fold*") có kích thước xấp xỉ nhau
  - Mỗi lần (trong số *k* lần) lặp, một tập con được sử dụng làm tập kiểm thử, và (k-1) tập con còn lại được dùng làm tập huấn luyện
  - k giá trị lỗi (mỗi giá trị tương ứng với một *fold*) được tính trung bình cộng để thu được giá trị lỗi tổng thể
- Các lựa chọn thông thường của k: 10, hoặc 5
- Thông thường, mỗi tập con (fold) được lấy mẫu phân tầng (xấp xỉ phân bố lớp) trước khi áp dụng quá trình đánh giá Cross-validation
- Phù hợp khi ta có tập ví dụ D vừa và nhỏ

Leave-one-out cross-validation

- Một trường hợp (kiểu) của phương pháp Cross-validation
  - Số lượng các nhóm (folds) bằng kích thước của tập dữ liệu (k=|D|)
  - Mỗi nhóm (fold) chỉ bao gồm một ví dụ
- Khai thác tối đa (triệt để) tập ví dụ ban đầu
- Không hề có bước lấy mẫu ngẫu nhiên (no random sub-sampling)
- Áp dụng lấy mẫu phân tầng (stratification) không phù hợp

  → Vì ở mỗi bước lặp, tập thử nghiệm chỉ gồm có một ví dụ
- Chi phí tính toán (rất) cao
- Phù hợp khi ta có một tập ví dụ D (rất) nhỏ

Bootstrap sampling (1)

- Phương pháp Cross-validation sử dụng việc lấy mẫu không lặp lại (sampling without replacement)

  → Đối với mỗi ví dụ, *một khi đã được chọn (được sử dụng), thì không thể được chọn (sử dụng) lại* cho tập huấn luyện

- Phương pháp Bootstrap sampling sử dụng việc ***lấy mẫu có lặp lại (sampling with replacement)*** để tạo nên tập huấn luyện
  - Giả sử tập toàn bộ D bao gồm n ví dụ
  - Lấy mẫu có lặp lại n lần đối với tập D, để tạo nên tập huấn luyện D_train gồm n ví dụ
    - Từ tập D, lấy ra ngẫu nhiên một ví dụ x (nhưng **không loại bỏ** x khỏi tập D)
    - Đưa ví dụ x vào trong tập huấn luyện: $D\_train = D\_train \cup x$
    - Lặp lại 2 bước trên n lần
  - Sử dụng tập D_train để huấn luyện hệ thống
  - Sử dụng tất cả các ví dụ thuộc D **nhưng không thuộc** D_train để tạo nên tập thử nghiệm: $D\_test = \{z \in D;\ z \notin D\_train\}$

Bootstrap sampling (2)

- Trong mỗi bước lặp, một ví dụ có xác suất $= \left(1 - \frac{1}{n}\right)$ để không được lựa chọn đưa vào tập huấn luyện
- Vì vậy, xác suất để một ví dụ (sau quá trình lấy mẫu lặp lại – bootstrap sampling) được đưa vào tập kiểm thử là:

$$\left(1 - \frac{1}{n}\right)^n \approx e^{-1} \approx 0.368$$

- Có nghĩa rằng:
  - Tập huấn luyện (có kích thước $=n$) bao gồm xấp xỉ 63.2% các ví dụ trong D (Lưu ý: Một ví dụ thuộc tập D có thể **xuất hiện nhiều lần** trong tập D_train)
  - Tập kiểm thử (có kích thước $<n$) bao gồm xấp xỉ 36.8% các ví dụ trong D (Lưu ý: Một ví dụ thuộc tập D chỉ có thể **xuất hiện tối đa 1 lần** trong tập D_test)
- Phù hợp khi ta có một tập dữ liệu D có kích thước (rất) nhỏ

Tập tối ưu (Validation set)

- Các ví dụ trong tập kiểm thử không thể được sử dụng (theo bất kỳ cách nào!) trong quá trình huấn luyện hệ thống
- Trong một số bài toán, quá trình huấn luyện hệ thống bao gồm 2 giai đoạn
  - Giai đoạn thứ 1: Huấn luyện hệ thống (= Học hàm mục tiêu)
  - Giai đoạn thứ 2: Tối ưu giá trị các tham số của hệ thống
- Tập kiểm thử không thể được sử dụng cho mục đích tối ưu (điều chỉnh) tham số!
- Chia tập toàn bộ các ví dụ D thành 3 tập con không giao nhau: tập *huấn luyện*, tập *tối ưu*, và tập *kiểm thử*
- Tập tối ưu (validation set) được sử dụng để tối ưu giá trị các tham số được sử dụng

  → Đối với một tham số, giá trị tối ưu là giá trị giúp sinh ra *hiệu năng cực đại đối với tập tối ưu*

Các tiêu chí đánh giá (1)

- **Tính chính xác (Accuracy)**
  - →Mức độ dự đoán (phân lớp) chính xác của hệ thống (đã được huấn luyện) đối với các ví dụ kiểm chứng (test instances)
- Tính hiệu quả (Efficiency)
  - →Chi phí về thời gian và tài nguyên (bộ nhớ) cần thiết cho việc huấn luyện và kiểm thử hệ thống
- Khả năng xử lý nhiễu (Robustness)
  - →Khả năng xử lý (chịu được) của hệ thống đối với các ví dụ nhiễu (lỗi) hoặc thiếu giá trị

Các tiêu chí đánh giá (2)

- Khả năng mở rộng (Scalability)
  - →Hiệu năng của hệ thống (vd: tốc độ huấn luyện/phân loại) thay đổi như thế nào đối với kích thước của tập dữ liệu
- Khả năng diễn giải (Interpretability)
  - →Mức độ dễ hiểu (đối với người sử dụng) của các kết quả và hoạt động của hệ thống
- Mức độ phức tạp (Complexity)
  - →Mức độ phức tạp của mô hình hệ thống (hàm mục tiêu) học được

Lựa chọn mô hình

- Việc lựa chọn mô hình cần tìm ra sự thỏa hiệp (compromise) phù hợp giữa
  - Mức độ phức tạp của mô hình hệ thống học được
  - Mức độ chính xác về dự đoán của hệ thống đối với tập huấn luyện
- *Nguyên lý Occam's razor*. Một mô hình tốt là một mô hình **đơn giản** đạt **độ chính xác (về phân loại/dự đoán) cao** đối với tập huấn luyện được sử dụng
- Ví dụ:
  - Bộ phân loại `Sys1`: (Rất) đơn giản, và khá (tương đối) phù hợp với tập huấn luyện
  - Bộ phân loại `Sys2`: Khá phức tạp, và phù hợp hoàn hảo với tập huấn luyện

  →Bộ phân loại `Sys1` được ưa thích hơn bộ phân loại `Sys2`