HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHÓA HỌC

HỌC MÁY CHO AN TOÀN THÔNG TIN

PHẦN 1: CƠ BẢN VỀ HỌC MÁY VÀ HỌC SÂU

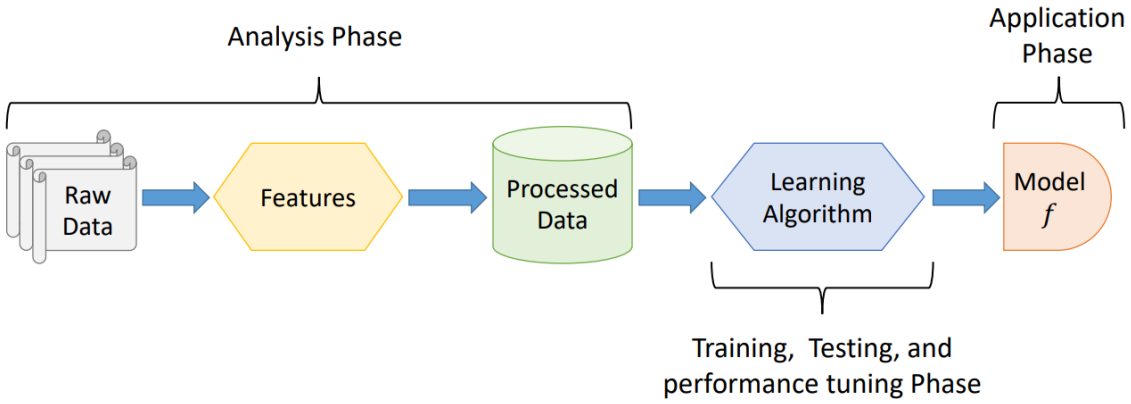
Giảng viên: TS. Nguyễn Ngọc Điệp

E-mail: diepnguyenngoc@ptit.edu.vn

Đơn vị: Khoa An toàn thông tin, Học viện Công nghệ BCVT

Nội dung

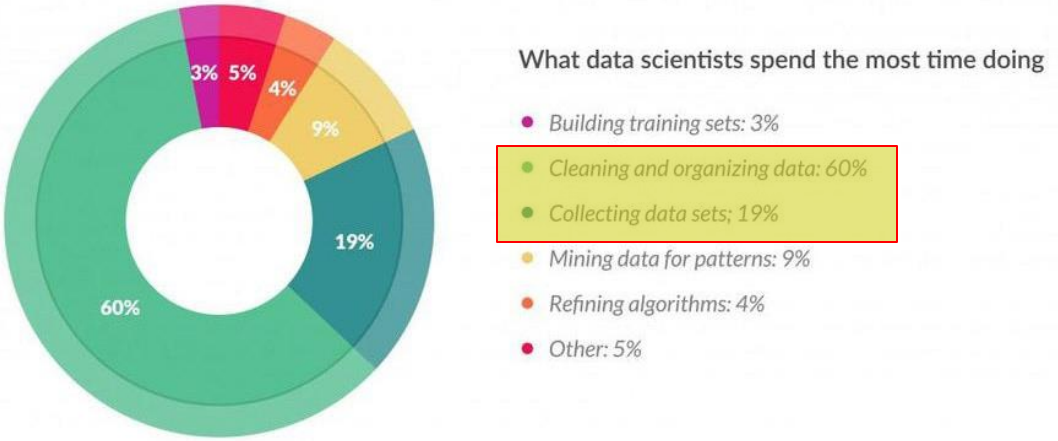
- Tiền xử lý dữ liệu
  - Làm sạch dữ liệu
  - Trích chọn đặc trưng
  - Giảm chiều dữ liệu
  - Xử lý dữ liệu không cân bằng
- Đánh giá mô hình học máy
  - Ma trận nhầm lẫn
  - Các độ đo đánh giá



Nội dung

- Tiền xử lý dữ liệu
  - Làm sạch dữ liệu
  - Trích chọn đặc trưng
  - Giảm chiều dữ liệu
  - Xử lý dữ liệu không cân bằng

Tham khảo các slide từ: Thu thập và Tiền xử lý dữ liệu - Thân Quang Khoát, Nguyễn Minh Phương, Lê Minh Hoà



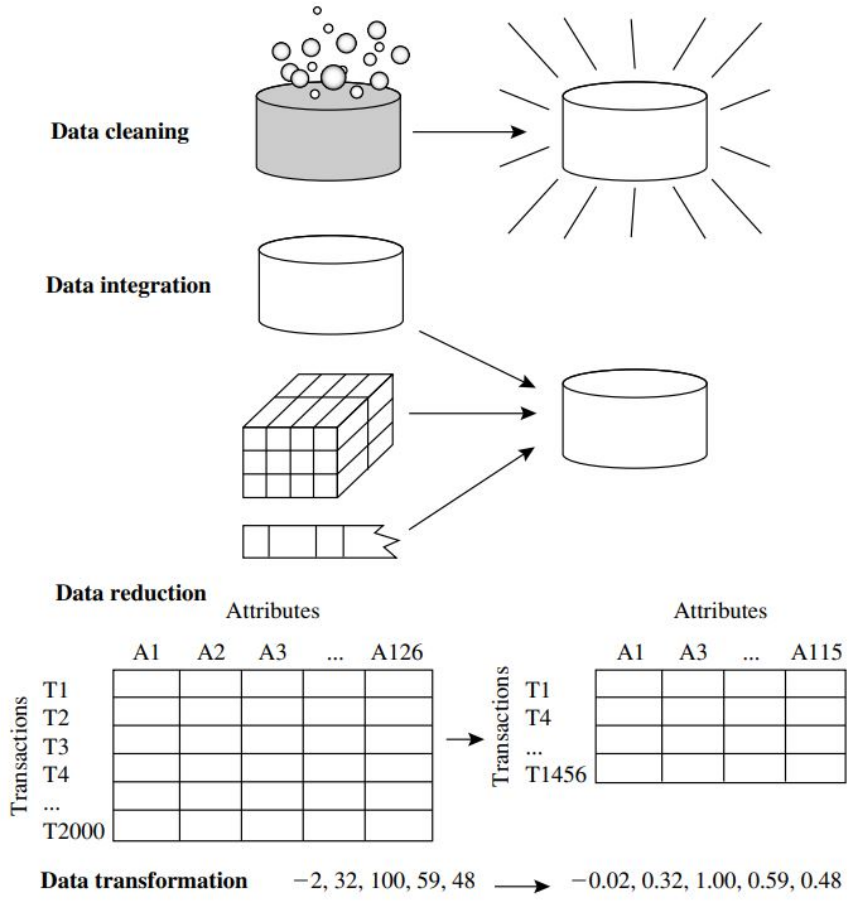
Press Gill, "Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says", Forbes, 2016.

Tiền xử lý dữ liệu

- Mục đích
  - Mô hình xác suất/ mô hình học máy đạt hiệu quả
    - Do dữ liệu thô thường có nhiều sai sót và nhiễu, có thể ảnh hưởng đến hiệu suất của mô hình.
    - Các mô hình Học máy chỉ làm việc với dữ liệu ma trận hoặc vector
  - Dễ lưu trữ / truy vấn
  - Tiền xử lý dữ liệu chuẩn bị dữ liệu để mô hình học máy có thể học và đưa ra dự đoán chính xác. Đảm bảo rằng dữ liệu ở trong trạng thái tốt nhất để mô hình sử dụng.

Tiền xử lý dữ liệu

- Các bước:
  - Thu thập dữ liệu
    - Lấy mẫu
    - Kỹ thuật: crawling, logging, scraping
  - Xử lý dữ liệu:
    - Dữ liệu cần: Lọc nhiễu, số hóa
    - Kĩ thuật: làm sạch, số hóa, lưu trữ, giảm chiều



Tham khảo: Data Mining: Concepts and Techniques

Tiền xử lý dữ liệu

- Lấy mẫu:
  - What: lấy tập mẫu nhỏ, phổ biến để đại diện cho lĩnh vực cần học.
  - How: thu thập các mẫu từ thực tế, hoặc các nguồn chứa dữ liệu web, database, ..
  - Why: không thể học toàn bộ. Giới hạn về thời gian và khả năng tính toán
- Yêu cầu:
  - Đa dạng: tập mẫu thu được đủ đa dạng để phủ hết các ngữ cảnh của lĩnh vực.
  - Tổng quát: dữ liệu cần tổng quát, không bị sai lệch, thiên vị về 1 bộ phận nhỏ nào đó của lĩnh vực.
  - Bảo vệ thông tin nhạy cảm: đối với an toàn thông tin, không tiết lộ thông tin nhạy cảm
  - Vấn đề khác: tuân thủ luật, bản quyền, bảo vệ dữ liệu mẫu, đào tạo cách lấy mẫu cho các vấn đề nhạy cảm về an toàn thông tin

Tiền xử lý dữ liệu

- Kĩ thuật lấy mẫu:
  - Crowd-sourcing: thực hiện các khảo sát
  - Logging: vd lưu lại lịch sử tương tác của người dùng, truy cập sản phẩm,…
  - Scrapping tìm kiếm nguồn dữ liệu trên các website, tải về bóc tách, lọc …

Tiền xử lý dữ liệu

- Dữ liệu thô, cần có:
  - Tính đầy đủ: Từng mẫu thu thập nên đầy đủ thông tin các trường thuộc tính cần thiết
  - Tính rõ ràng: Nguồn thu thập chính thống, đảm bảo mẫu thu được chứa giá trị chính xác trên thực tế.
  - Tính đồng nhất:
    - Rating “1, 2, 3” & “A, B, C”; or Age = “42” & Birthday = “03/07/2010” (inconsistency)

Tiền xử lý dữ liệu

- Kĩ thuật xử lý dữ liệu:
  - Làm sạch, số hóa, lưu trữ, giảm chiều
- Tại sao cần làm sạch:
  - Mẫu dữ liệu cần được thu thập từ các nguồn đáng tin cậy và phản ánh vấn đề cần giải quyết.
  - Mẫu dữ liệu thu thập đôi khi không thể đầy đủ và rõ ràng, cần có chiến lược phù hợp:
    - Bỏ qua, không đưa vào dữ liệu học.
    - Bổ sung các trường còn thiếu cho mẫu: Bằng tay/ Tự động
  - Mẫu dữ liệu thu thập đôi khi không đồng nhất về cách biểu diễn, về ký hiệu
    - Cần đồng nhất
    - Ví dụ: Rating “1, 2, 3” & “A, B, C”; or Age = “42” & Birthday = “03/07/2010”(inconsistency)

Tiền xử lý dữ liệu

- Làm sạch dữ liệu:
  - Điền các giá trị thiếu (missing values)
    - Điền thủ công/ điền tự động với hàm nội suy
  - Làm mịn dữ liệu nhiễu
  - Xác định hoặc loại bỏ các ngoại lai
  - Giải quyết các bất thường (dựa vào phân cụm)

Sorted data for *price* (in dollars): 4, 8, 15, 21, 21, 24, 25, 28, 34

Partition into (equal-frequency) bins:
Bin 1: 4, 8, 15
Bin 2: 21, 21, 24
Bin 3: 25, 28, 34

Smoothing by bin means:
Bin 1: 9, 9, 9
Bin 2: 22, 22, 22
Bin 3: 29, 29, 29

Smoothing by bin boundaries:
Bin 1: 4, 4, 15
Bin 2: 21, 21, 24
Bin 3: 25, 25, 34

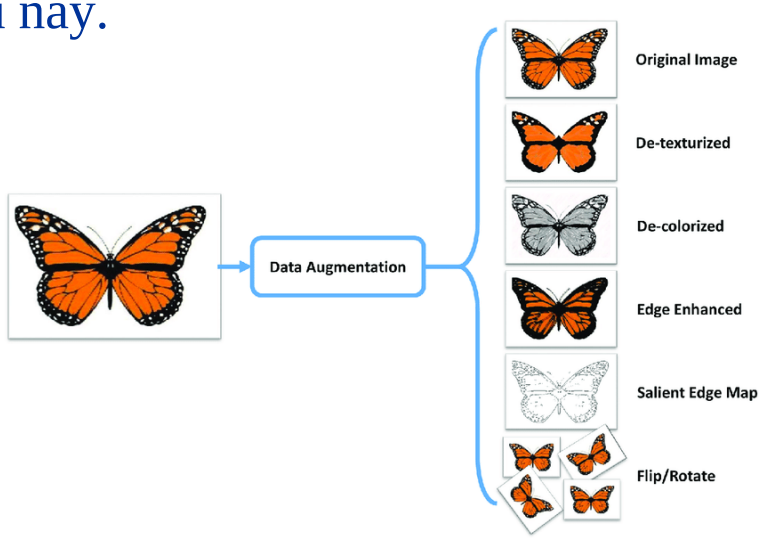
Tiền xử lý dữ liệu

❖ Tích hợp dữ liệu:

- Tích hợp dữ liệu (integration) là quá trình hợp nhất dữ liệu từ nhiều kho dữ liệu khác nhau.
- Giúp thu giảm (reduce), tránh dư thừa dữ liệu (redundant data) và tránh mất đi tính toàn vẹn dữ liệu (inconsistent data) trong tập dữ liệu.
- Đồng thời, tích hợp dữ liệu còn giúp ta cải thiện độ chính xác (accuracy) và tốc độ cho quá trình khai phá dữ liệu sau này.

❖ Bổ sung thêm dữ liệu (data augmentation)

- Giúp mở rộng tập dữ liệu bằng cách tạo ra các mẫu dữ liệu mới từ các mẫu dữ liệu hiện có



Tiền xử lý dữ liệu

- Bổ sung thêm dữ liệu (data augmentation)
  - Giúp mở rộng tập dữ liệu bằng cách tạo ra các mẫu dữ liệu mới từ các mẫu dữ liệu hiện có
    - tạo ra nhiều biến thể của một mẫu dữ liệu gốc mà không cần thu thập thêm dữ liệu mới.
    - Giảm overfitting và chống nhiễu
- Cách thức
  - Dữ liệu số nói chung: thêm các dữ liệu thống kê cơ bản
    - trung bình, trung vị, mốt, phạm vi, phương sai, độ lệch chuẩn, …
  - Đối với ảnh:
    - Xoay, lật, thay đổi kích thước, tương phản, Thay đổi độ sáng, độ bão hòa; Thêm nhiễu;
  - Đối với văn bản:
    - Thêm nhiễu, dịch sang ngôn ngữ khác, thứ tự từ, xóa từ, …; Thêm các mô tả khái niệm, dữ liệu hỗ trợ trong knowledge base, …
  - Đối với âm thanh: thêm nhiễu, thay đổi tốc độ, âm lượng, nhiễu nền, …

Tiền xử lý dữ liệu

❖ Chuyển đổi và chuẩn hóa dữ liệu:

- Dữ liệu có cấu trúc (dạng bảng) và không có cấu trúc/bán cấu trúc

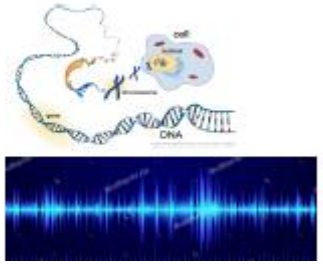
	A	B	C	D	E	F	G
1	Country	Region	Population	Under15	Over60	Fertil	LifeExp
2	Zimbabwe	Africa	13724	40.24	5.68	3.64	54
3	Zambia	Africa	14075	46.73	3.95	5.77	55
4	Yemen	Eastern M	23852	40.72	4.54	4.35	64
5	Viet Nam	Western P	90796	22.87	9.32	1.79	75
6	Venezuela (Bo	Americas	29955	28.84	9.17	2.44	75
7	Vanuatu	Western P	247	37.37	6.02	3.46	72
8	Uzbekistan	Europe	28541	28.9	6.38	2.38	68
9	Uruguay	Americas	3395	22.05	18.59	2.07	77

```
<?xml version="1.0"?>
- <birds>
  - <owl id="1201">
      <species>Bubo bubo</species>
      <name>Eagle Owl</name>
      <region>Eurasia</region>
    </owl>
  - <owl id="1202">
      <species>Strix occidentalis</species>
      <name>Spotted Owl</name>
      <region>North America</region>
    </owl>
</birds>
```

texts in websites, emails, articles, tweets

2D/3D images, videos + meta

spectrograms, DNAs, ...



Tiền xử lý dữ liệu

❖ Chuyển đổi và chuẩn hóa dữ liệu:

- Chuẩn hóa:
  - Feature discretization (rời rạc hóa) – một số thuộc tính tỏ ra hiệu quả hơn khi được phân nhóm, sắp xếp trước. VD: các giá trị liên tục ➔ các khoảng (bins). Phân tích histogram/cluster/ decision tree/phân tích tương quan
  - Feature normalization - chuẩn hóa giá trị thuộc tính, về cùng một miền giá trị, dễ dàng trong tính toán.
- Trích xuất đặc trưng ngữ nghĩa:
  - Từng lĩnh vực cụ thể, từng loại dữ liệu sử dụng các kỹ thuật xuất đặc trưng ngữ nghĩa khác nhau (dữ liệu text, hình ảnh, …)
  - Cần mức ngữ nghĩa tối thiểu để có thể hiểu
    - » Nhằm phân loại văn bản, Phân tích cảm xúc, Phân loại hình ảnh, …

B	C	D	E	F	G
Region	Populat	Under1	Over60	Fertil	LifeExp
Africa	-0.416	0.748	-0.483	0.299	54
Africa	-0.403	1.464	-0.850	1.881	55
Eastern M	-0.060	0.801	-0.725	0.826	64
Western P	2.287	-1.169	0.289	-1.075	75
Americas	0.154	-0.511	0.257	-0.592	75
Western P	-0.888	0.431	-0.411	0.165	72
Europe	0.104	-0.504	-0.334	-0.637	68
Americas	-0.778	-1.260	2.256	-0.867	77

One-hot encoding

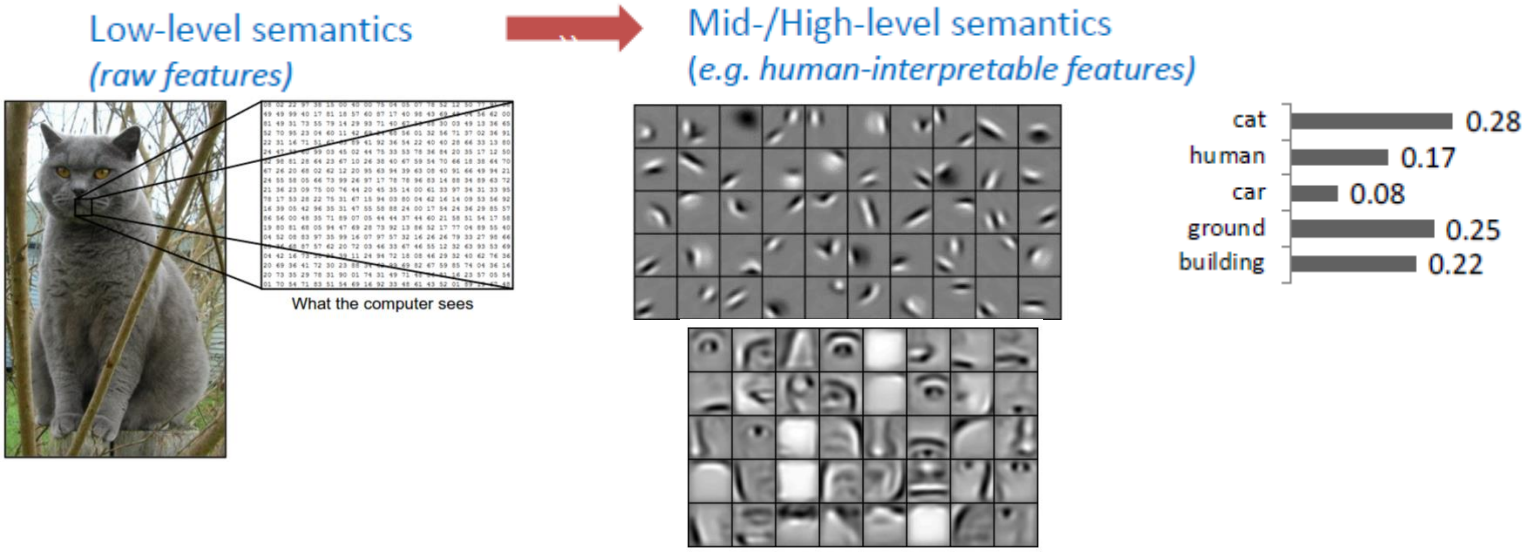
$1 = [1\,0\,0\,0\,0]$
$3 = [0\,0\,1\,0\,0]$

$$\frac{x - \bar{x}}{s}$$

```
USD điều_chỉnh trái chiều , vàng SJC quay đầu tăng
  (0, 24506)    0.2077168092100841
  (0, 23857)    0.34468369118902636
  (0, 22309)    0.31713411814089415
  (0, 21894)    0.3025597601047669
  (0, 21265)    0.2449372095782497
  (0, 20409)    0.3276089788346888
  (0, 17739)    0.515839529548281
  (0, 16499)    0.33820735665113805
  (0, 4648)     0.3132633187744836
```

Tiền xử lý dữ liệu

❖ Chuyển đổi và chuẩn hóa dữ liệu:

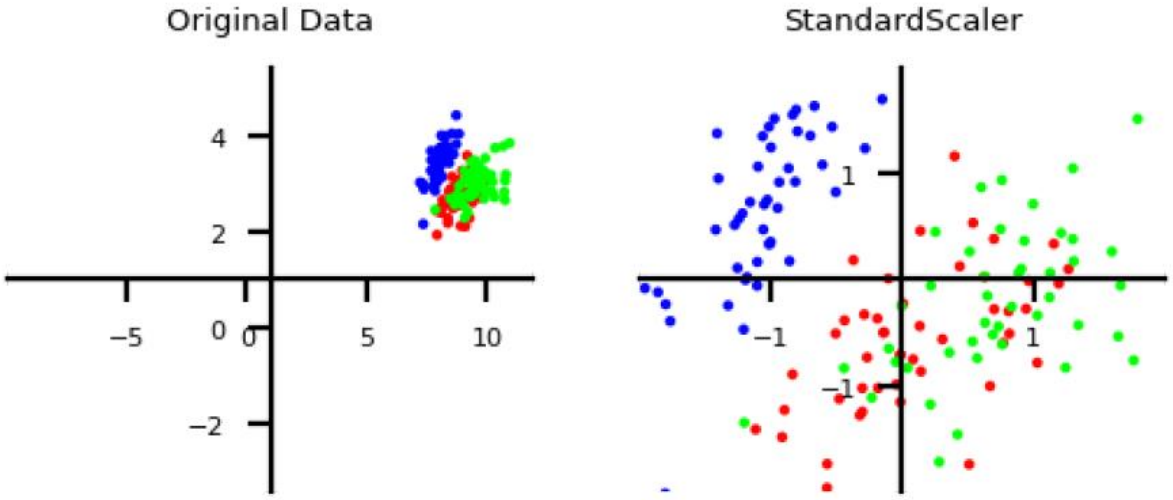


Tiền xử lý dữ liệu

- Giảm chiều dữ liệu:
  - Là quá trình giảm số lượng biến ngẫu nhiên hoặc thuộc tính cần xét.
  - Nhằm thu được một tập dữ liệu nhỏ hơn nhưng vẫn giữ được (hoặc gần như giữ được) kết quả phân tích tương tự như tập dữ liệu gốc. Điều này giúp việc phân tích trên tập dữ liệu giảm thiểu trở nên hiệu quả hơn.
- Kĩ thuật:
  - Biến đổi wavelet (Wavelet Transforms): Trong đó có Chuyển đổi wavelet rời rạc (DWT)
  - Phân tích thành phần chính (Principal Components Analysis - PCA): Tìm kiếm k vector trực giao n chiều có thể được sử dụng tốt nhất để biểu diễn dữ liệu, trong đó $k \leq n$. Dữ liệu gốc được chiếu lên một không gian nhỏ hơn nhiều, dẫn đến giảm chiều.
  - Giảm kích thước của tập dữ liệu bằng cách loại bỏ các thuộc tính không liên quan hoặc trùng lặp, hoặc không quan trọng (Sử dụng Decision Tree/Random Forest).

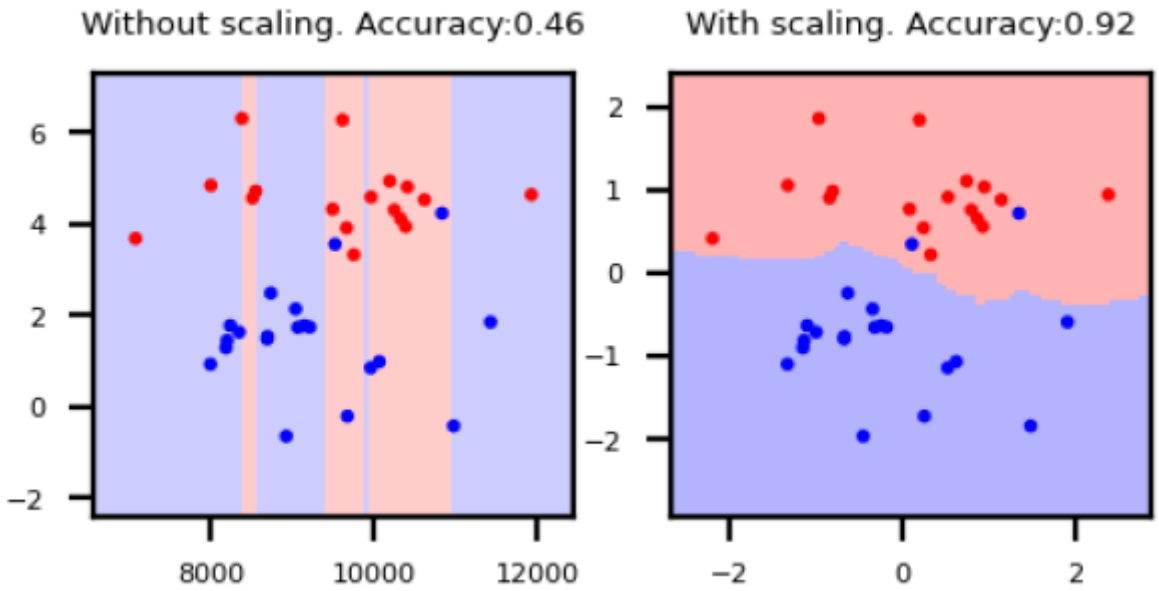
Một số kĩ thuật khác trong Trích xuất đặc trưng và ví dụ

- Feature Scaling
- Chuyển đổi dữ liệu:
  - Liên tục
  - Rời rạc
- Xử lý ngôn ngữ tự nhiên cơ bản



Một số kĩ thuật khác trong Trích chọn đặc trưng

- Feature Scaling:
  - KNN: thuộc tính có giá trị lớn hơn sẽ chiếm ưu thế trong việc tính toán khoảng cách
  - SVM:kết quả nhân vẫn phụ thuộc vào khoảng cách được tính toán trong không gian đặc trưng được ánh xạ bởi kernel.
  - Mô hình tuyến tính: ảnh hưởng đến hiệu quả của các kỹ thuật chính tắc hóa (regularization).



Một số kĩ thuật khác trong Trích chọn đặc trưng

- Feature Scaling:
  - Rescaling: đưa tất cả các thành phần về cùng một khoảng, [0 , 1] hoặc [− 1 , 1]. Sử dụng bộ chuẩn hóa MinMaxScaler trong scikit-learn
    - VD: x' thuộc khoảng [0 , 1]

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

  - Standardization: giả sử mỗi thành phần đều có phân phối chuẩn với kỳ vọng là 0 và phương sai là 1. Chuẩn hóa với:

$$x' = \frac{x - \bar{x}}{\sigma}$$

với x hat , σ lần lượt là kỳ vọng và phương sai (standard deviation) của thành phần đó trên toàn bộ tập dữ liệu huấn luyện.

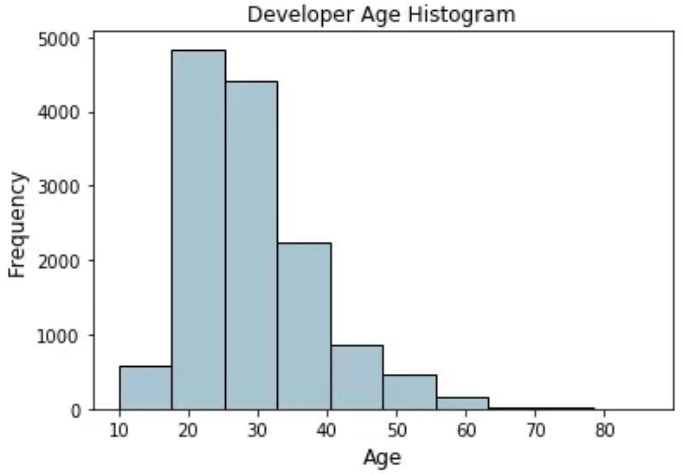
  - Chuẩn hóa với vector có độ lớn (Euclid, tức norm 2) = 1

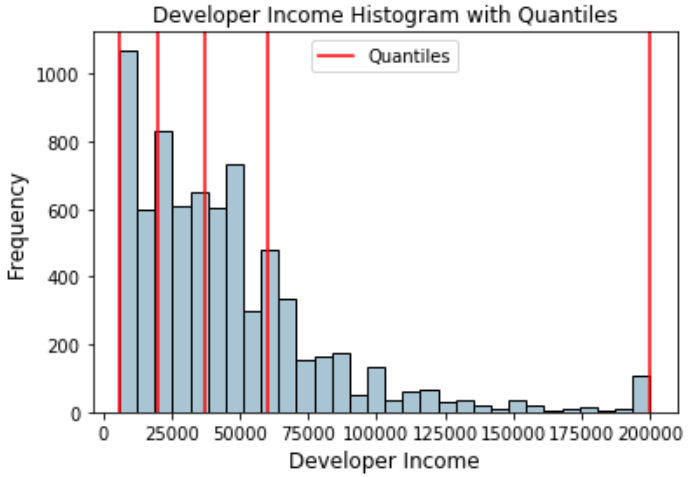
$$\mathbf{x}' = \frac{\mathbf{x}}{||\mathbf{x}||_2}$$

  - Nhị phân hóa: Binarization (thư viện Binarizer của scikit-learn):
    - Chuyển từ tần số, số lượng đếm được sang trạng thái: có/không ~ 1/0

Một số kĩ thuật khác trong Trích chọn đặc trưng

- Biến đổi Binning/ quantization (lượng tử hóa)
  - biến đổi các đặc trưng số liên tục thành dạng các đặc trưng phân loại (categorical) riêng biệt thay vì tính toán trực tiếp trên các giá trị raw có thể gây ra nhiễu mô hình
  - Có thể tạo bin theo độ rộng thích nghi
    - Quantile-based: Quantiles là các giá trị cụ thể hoặc các điểm cắt chia phân phối có giá trị liên tục của một trường thành các phân vùng là các khoảng liền kề rời rạc.
    - q-Quantiles sẽ chia một đặc trưng thành q phân vùng bằng nhau
    - VD: Biểu đồ phân thối thu nhập của các dev với phân chia 4-quantiles. Đường màu đỏ trong biểu đồ là đường phân chia các bin. Chúng ta sẽ sử dụng các phân chia này để tạo bin dựa trên 4-quantiles.





Một số kĩ thuật khác trong Trích chọn đặc trưng

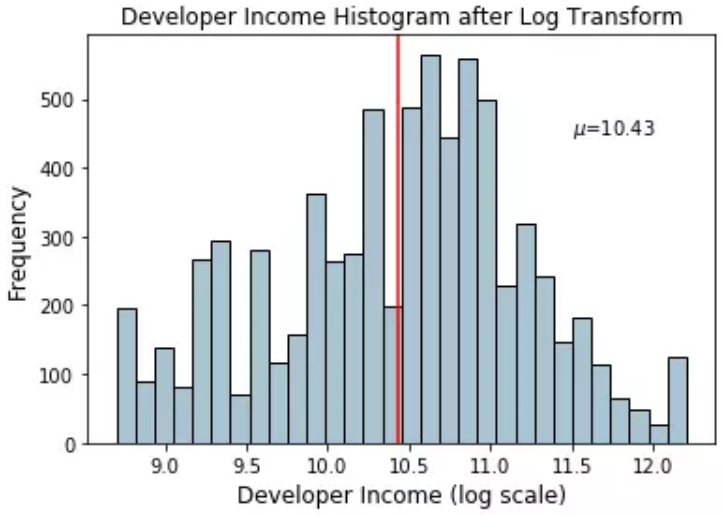
❖ Biến đổi Log

- Giảm tác động bất lợi của phân phối dữ liệu không chuẩn đến các mô hình học máy
- Biến đổi Log là phép biến đổi dữ liệu đơn điệu.

$$y = \log_b(x)$$

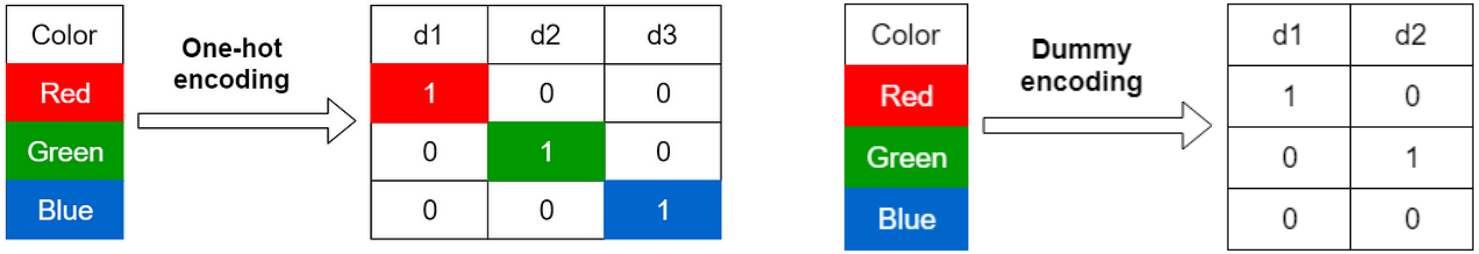
- Ý nghĩa chính của chúng là giúp ổn định phương sai, tuân thủ chặt chẽ phân phối chuẩn và làm cho các dữ liệu độc lập với giá trị trung bình dựa trên phân phối của nó.

Từ biểu đồ có thể thấy sau khi biến đối thì giá trị của Income đã tuân theo phân phối chuẩn (Gaussian) hơn so với dữ liệu gốc.



Một số kĩ thuật khác trong Trích chọn đặc trưng

- Xử lý dữ liệu phân loại (categorial)
  - Có hai dạng chính: danh nghĩa (nominal) và thứ tự (ordinal)
    - Danh nghĩa: các loại thời tiết, loại âm nhạc
    - Thứ tự: kích cỡ áo, trình độ học vấn, …
  - Ví dụ:
    - Chuyển đổi thể loại âm nhạc sang dạng số, sử dụng LabelEncoder trong scikit-learn: 0: 'Action', 1: 'Adventure', 2: 'Fighting', 3: 'Misc', 4: 'Platform', 5: 'Puzzle', 6: 'Racing', 7: 'Role-Playing', 8: 'Shooter', 9: 'Simulation', 10: 'Sports', 11: 'Strategy'
    - Chuyển đổi dữ liệu thứ tự sang dạng số: tương tự như trên
  - Mã hóa dữ liệu phân loại
    - Onehot encoding và dummy encoding



Một số kĩ thuật khác trong Trích chọn đặc trưng

- Xử lý dữ liệu phân loại (categorial)
  - Bin-counting:
    - Vấn đề: số danh mục lớn (ví dụ như địa chỉ IP) ➔ dữ liệu lớn
    - Bin-counting sử dụng thông tin thống kê dựa trên xác suất về giá trị và mục tiêu thực tế hoặc nhãn chúng ta nhắm đến để dự đoán trong các mô hình của mình
    - Ví dụ: xây dựng các giá trị xác suất cho một cuộc tấn công tương tự được gây ra bởi bất cứ địa chỉ IP nào đã có trong lịch sử. Từ đó encode một địa chỉ IP xuất hiện trong tương lai có giá trị xác suất tấn công DDOS là bao nhiêu.
    - Cần giá trị thống kê đủ lớn, công phu
  - Feature hashing:
    - Sử dụng hàm băm để nhóm các đặc trưng thành một tập hợp n bins hữu hạn sao cho khi hàm băm được áp dụng trên cùng một giá trị nhóm/danh mục chúng sẽ được gán vào cùng một bin (hoặc một tập hợp các bins) trong số n bins được tạo ra trước đó.
      ➔ 1000 danh mục khác nhau ➔ nhóm vào 10 bin (10 nhãn)
    - Hàm FeatureHasher trong scikit-learn

Một số kĩ thuật khác trong Trích chọn đặc trưng

- Xử lý dữ liệu văn bản trong NLP
  - Dữ liệu văn bản: chữ cái thường, chữ cái hoa, dấu câu, các kí tự đặc biệt,… Các ngôn ngữ khác nhau cũng có mẫu kí tự khác nhau và cấu trúc ngữ pháp khác nhau.
  - Làm thế nào mã hoá được kí tự về dạng số (véc tơ)?

    ➔ Kĩ thuật tokenization chia văn bản theo đơn vị nhỏ nhất và xây dựng một từ điển đánh dấu index cho những đơn vị này.
  - Có hai kiểu mã hoá chính là mã hoá theo từ và mã hoá theo kí tự.
- Phương pháp túi từ - bag-of-words – BoW
  - Mã hoá các từ trong câu thành một véc tơ có độ dài bằng số lượng các từ trong từ điển và đếm tần suất xuất hiện của các từ.
  - Tần xuất của từ thứ *i* trong từ điển sẽ chính bằng phần tử thứ *i* trong véc tơ.

Một số kĩ thuật khác trong Trích chọn đặc trưng

- Xử lý dữ liệu văn bản trong NLP
  - Dữ liệu văn bản: chữ cái thường, chữ cái hoa, dấu câu, các kí tự đặc biệt,… Các ngôn ngữ khác nhau cũng có mẫu kí tự khác nhau và cấu trúc ngữ pháp khác nhau.
  - Làm thế nào mã hoá được kí tự về dạng số (véc tơ)?

    ➔ Kĩ thuật tokenization chi

điển đánh dấu index cho những c
  - Có hai kiểu mã hoá chính là m
- Phương pháp túi từ - bag-

  - Mã hoá các từ trong câu thành
điển và đếm tần suất xuất hiệr
  - Tần xuất của từ thứ trong từ đ

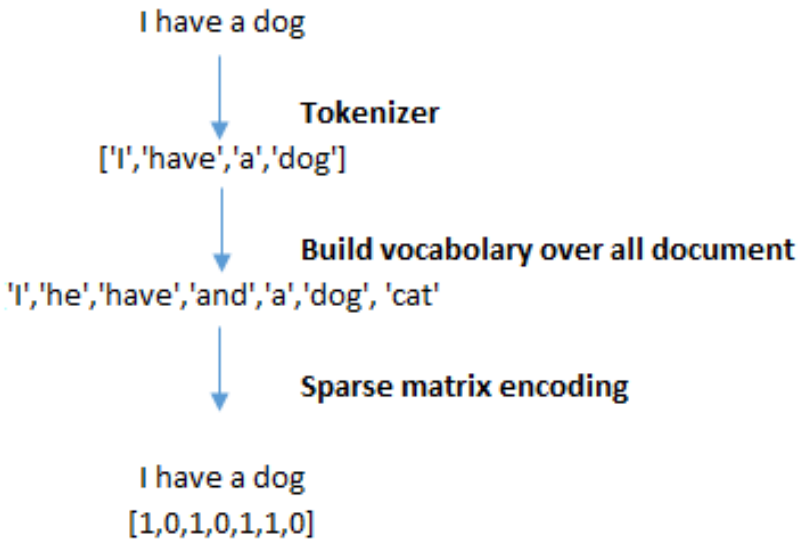
I have a greate AI book,
I have to read twice times

I	2
AI	1
a	1
about	1
book	1
deep	0
greate	1
is	0
this	0
machine	0
learning	0
have	2
to	1
read	1
twice	1
times	1

Một số kĩ thuật khác trong Trích chọn đặc trưng

- Phương pháp túi từ - bag-of-words – BoW
  - Mã hoá các từ trong câu thành một véc tơ có độ dài bằng số lượng các từ trong từ điển và đếm tần suất xuất hiện của các từ.
  - Tần xuất của từ thứ trong từ điển sẽ chính bằng phần tử thứ trong véc tơ.
  - Mã hóa câu: Ví dụ: I have a dog ➔ [1,0,1,0,1,1,0]

  - Hạn chế: không phân biệt được 2 câu văn có cùng các từ do không phân biệt thứ tự trước sau của các từ trong một câu.
    - Vd: ‘you have no dog’ = ‘no, you have dog’



Một số kĩ thuật khác trong Trích chọn đặc trưng

- Phương pháp túi ngram - bag-of-ngram
  - Mở rộng của bag-of-words
  - Một n-grams là một chuỗi bao gồm n tokens.
    - n = 1 ➔ unigram
    - n = 2 ➔ bigram
    - n = 3 ➔ trigram
  - Ví dụ: bag of bigram
    - I have a dog ➔ I have, have a, a dog
  - Hạn chế: số lượng các từ trong từ điển sẽ gia tăng một cách đáng kể. Véc tơ biểu diễn của câu trong bigram là một véc tơ rất thưa và có số chiều lớn. Điều này dẫn tới tốn kém về chi phí tính toán và lưu trữ
  - Thực hiện:
    - Thường kết hợp unigram, bigram, …
    - Sử dụng hàm CountVectorizer trong scikit-learn
  - Vấn đề: các từ hiếm không có trong từ điển, đôi khi rất quan trọng? ➔ sử dụng TF/IDF

Một số kĩ thuật khác trong Trích chọn đặc trưng

❖ Phương pháp TF-IDF

- Vấn đề: các từ không quan trọng xuất hiện nhiều, gây nhiễu. Ví dụ các từ dạng stopwords trong tiếng Anh: "the, a, an". Một số từ có mức độ quan trong cao trong một chủ đề, có mức ảnh hưởng lớn tới phân loại văn bản (spam hay không spam)
- Cách thức: đánh trọng số cho các từ xuất hiện ở một vài văn bản cụ thể lớn hơn nhờ TF/IDF, sau đó loại các từ không quan trọng

$$\mathrm{idf}(t, D) = \log \frac{|D|}{|\{d \in D; t \in d\}| + 1} = \log \frac{|D|}{\mathrm{df}(d, t) + 1}$$

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \times \mathrm{idf}(t, D)$$

- Với $|D|$ là số lượng các văn bản trong *bộ văn bản*.
  $\mathrm{df}(d, t) = |\{d \in D; t \in d\}|$ là tần suất các văn bản $d \in D$ mà từ t xuất hiện.
  $\mathrm{tf}(t, d)$ là tần suất xuất hiện của từ t trong văn bản d.

- idf(t, D): là chỉ số nghịch đảo tần suất văn bản (inverse document frequency) chỉ số này bằng logarith của nghịch đảo số lượng văn bản chia cho số lượng văn bản chứa một từ cụ thể t. Nếu chỉ số này lớn ➔ từ đó chỉ xuất hiện trong một số ít các văn bản

Một số kĩ thuật khác trong Trích chọn đặc trưng

❖ Phương pháp TF-IDF

$$\text{idf}(t, D) = \log \frac{|D|}{|\{d \in D; t \in d\}| + 1} = \log \frac{|D|}{\text{df}(d, t) + 1}$$
$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

$|D|$ là số lượng các văn bản trong *bộ văn bản*.

$\text{df}(d, t) = |\{d \in D; t \in d\}|$ là tần suất các văn bản $d \in D$ mà từ t xuất hiện.

$\text{tf}(t, d)$ là tần suất xuất hiện của từ t trong văn bản d.

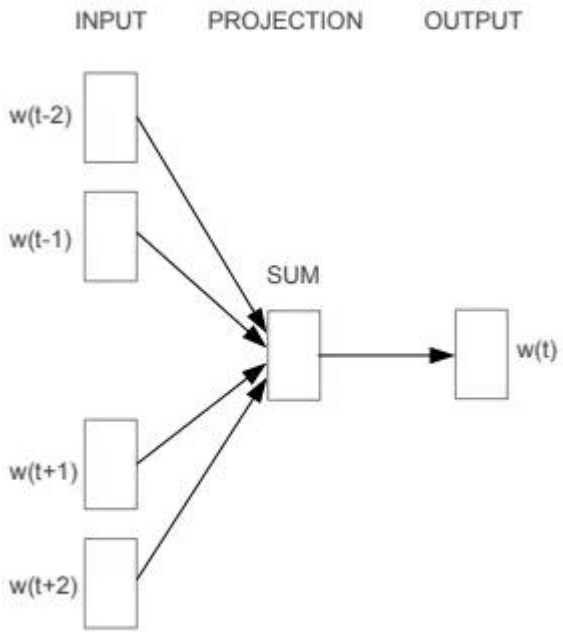
- tfidf (t,d,D) tỷ lệ thuận với tần suất của từ xuất hiện trong văn bản và nghịch đảo tần suất văn bản.
- Khi một từ càng quan trọng thì nó sẽ có tần suất xuất hiện trong một văn bản cụ thể, chẳng hạn văn bản d lớn, tức là tf(t,d) lớn và số lượng văn bản mà nó xuất hiện trong toàn bộ bộ văn bản nhỏ, suy ra idf(t,D) lớn.
- Trong scikit-learn, sử dụng hàm TfidfVectorizer

Một số kĩ thuật khác trong Trích chọn đặc trưng

- Phương pháp nhúng từ (Word embeddings)
  - Trích xuất đặc trưng văn bản với học sâu
  - Các phương pháp truyền thống xử lý văn bản không có cấu trúc, gây mất các thông tin bổ sung như ngữ nghĩa, cấu trúc, trình tự và ngữ cảnh xung quanh các từ gần đó mỗi tài liệu văn bản.
    - ➔ Véc tơ không giàu thông tin, dễ bị overfitting
    - VD: không phân biệt được “I have no dogs” và “No, I have dogs”
  - Phương pháp nhúng từ: "nhúng" các vectơ từ trong không gian vectơ liên tục dựa trên sự tương đồng về mặt ngữ nghĩa, ngữ cảnh
  - Có 2 phương pháp: CBOW và Skip-gram
- Mô hình Word2vec
  - CBOW - Continuous Bag of Words: dự đoán từ trung tâm (center word hoặc target word) dựa trên ngữ cảnh được tạo ra từ các từ xung quanh nó (surrounding words).

Một số kĩ thuật khác trong Trích chọn đặc trưng

- CBOW
  - VD: "the quick brown fox jumps over the lazy dog“
  - Các cặp: (context_window, target_word)
  - Khi context_window = 2, sẽ có:
    ([quick, fox], brown), ([the, brown], quick), ([the, dog], lazy)
- CBOW học không giám sát, dữ liệu không cần gán nhãn.
- Sau khi huấn luyện xong sẽ thu được véc tơ embedding của các từ

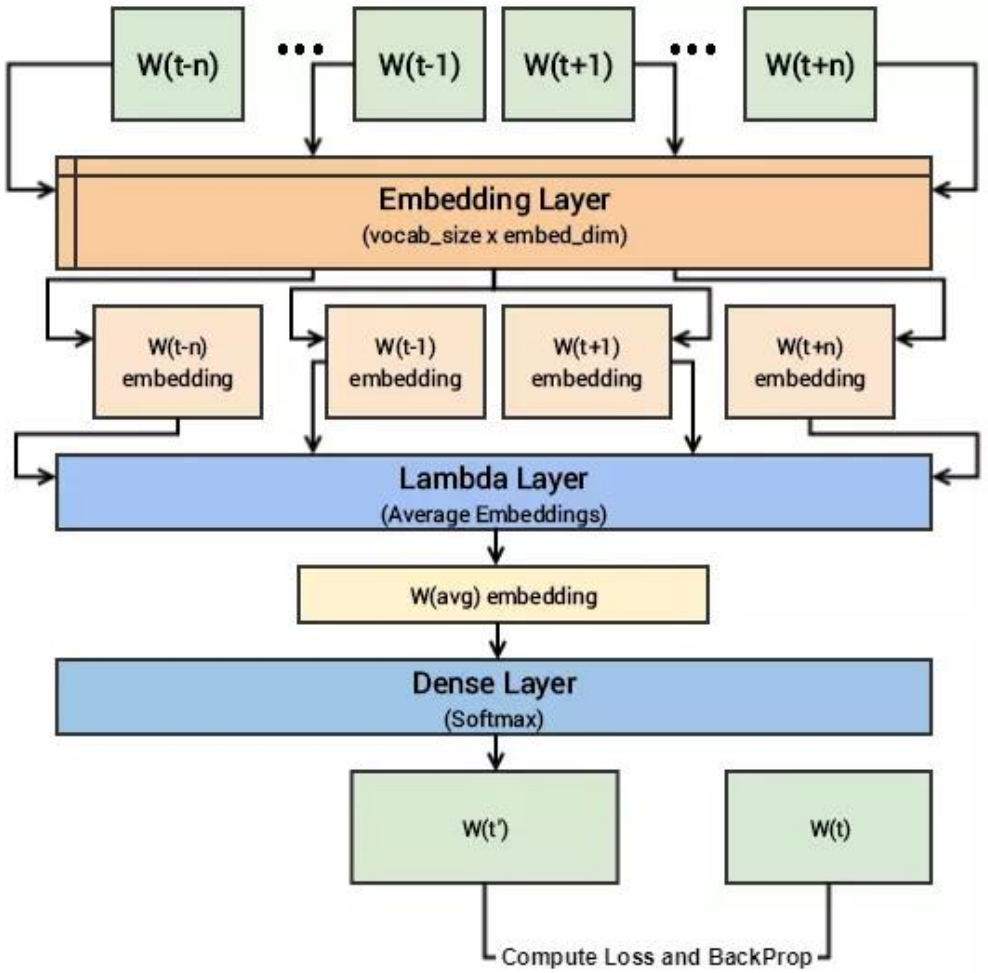


CBOW

Có thể dự đoán target_word dựa trên context_window

Một số kĩ thuật khác trong Trích chọn đặc trưng

- CBOW: Kiến trúc mô hình



Một số kĩ thuật khác trong Trích chọn đặc trưng

- CBOW: Ví dụ kết quả 5 từ với vector embedding tương ứng (1 x 100)

(12424, 100)

	0	1	2	3	4	5	6	7	8	9	...	90	91	92	93
shall	-1.183386	-2.866214	1.046431	0.943265	-1.021784	-0.047069	2.108584	-0.458692	-1.698881	0.905800	...	0.655786	0.703828	0.821803	-0.093732
unto	-1.725262	-1.765972	1.411971	0.917713	0.793832	0.310631	1.541964	-0.082523	-1.346811	0.095824	...	1.682762	-0.872293	1.908597	0.977152
lord	1.694633	-0.650949	-0.095796	0.950002	0.813837	1.538206	1.125482	-1.655581	-1.352673	0.409504	...	1.553925	-0.819261	1.086127	-1.545129
thou	-1.590623	-0.801968	1.659041	1.314925	-0.455822	1.733872	-0.233771	-0.638922	0.104744	0.490223	...	0.652781	-0.362778	-0.190355	0.040719
thy	0.386488	-0.834605	0.585985	0.801969	-0.165132	0.999917	1.224088	-0.317555	-0.671106	-1.073181	...	1.267184	-0.564660	0.089618	-0.979835

5 rows × 100 columns

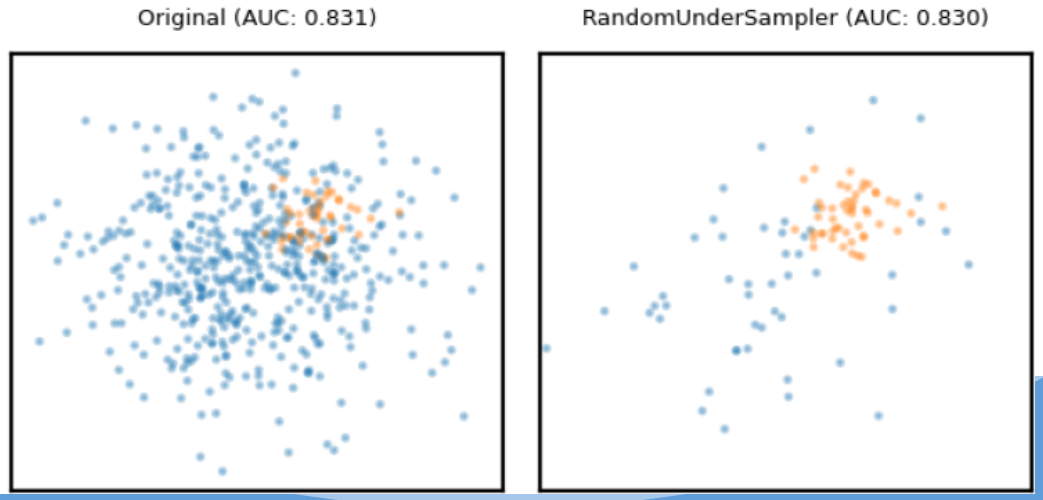
- Các từ tương đồng thật sự theo ngữ cảnh:
  - VD: (god, heaven), (gospel, church)

Xử lý dữ liệu mất cân bằng

- Tập dữ liệu phân loại đa lớp, nhưng số lượng mẫu dữ liệu của các lớp (categories) không bằng nhau
- Hoặc: lớp cân bằng, nhưng chi phí sai sót (cost of errors) lại khác nhau.
  - Ví dụ, một dự đoán "âm tính giả" nghiêm trọng hơn "dương tính giả“
- Cách xử lý đã biết:
  - Sử dụng tham số class_weight='balanced' trong nhiều thư viện học máy để thiết lập trọng số tự động cân bằng.
  - Thay đổi ngưỡng phân loại đầu ra của mô hình để ưu tiên giảm thiểu một loại lỗi nhất định. Ví dụ, bạn có thể điều chỉnh ngưỡng để giảm thiểu số FN nếu FN nghiêm trọng hơn FP.

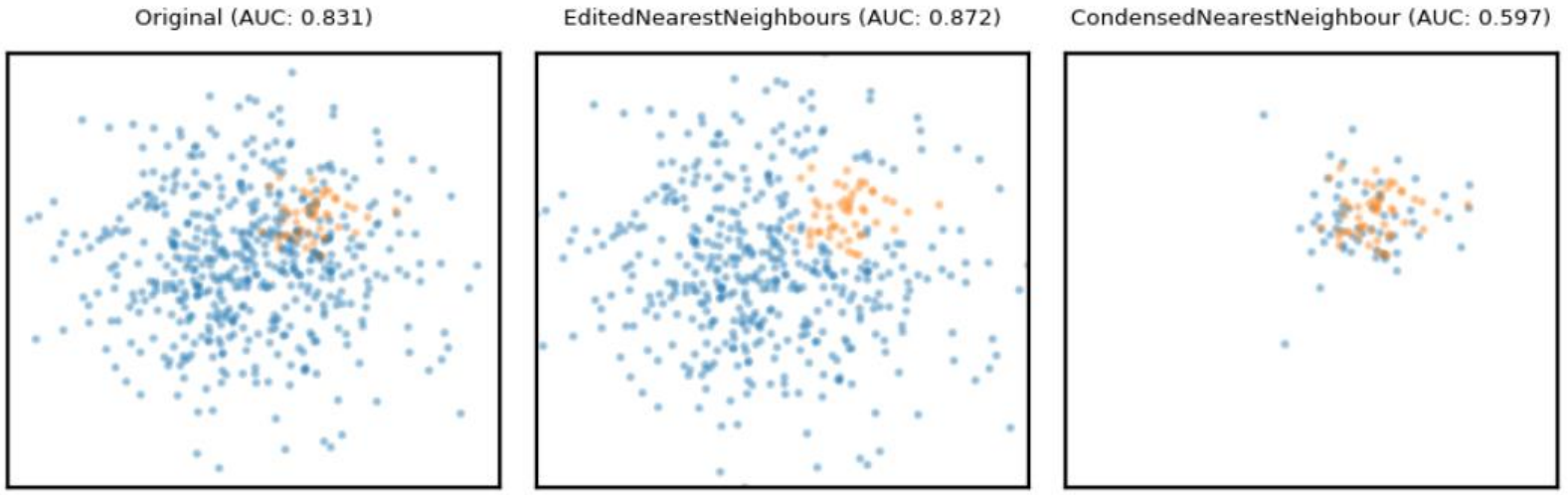
Lấy mẫu ngẫu nhiên giảm thiểu lớp đa số (Random Undersampling)

- Giữ nguyên tất cả các mẫu dữ liệu thuộc lớp thiểu số
- Chọn ngẫu nhiên một tập con mẫu từ lớp đa số cho đến khi có kích thước bằng với lớp thiểu số
  - Có hoặc không thay thế
  - Lấy mẫu cho đến khi đạt tỷ lệ mất cân bằng nhất định
  - Nếu có nhiều hơn hai lớp, lặp lại các bước trên với từng lớp còn lại so sánh với lớp thiểu số đã chọn.
- Thích hợp cho tập dữ liệu lớn: Mô hình nhỏ và nhanh hơn, hiệu suất phân loại tương tự



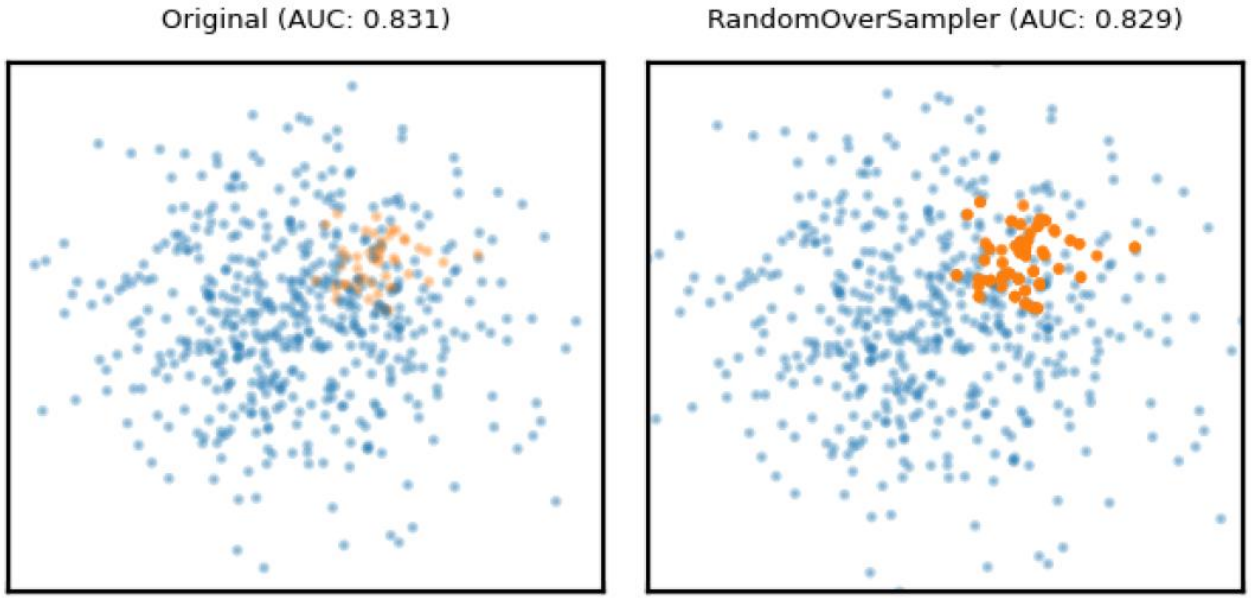
Lấy mẫu giảm thiểu lớp đa số dựa trên mô hình

- Nearest Neighbors được chỉnh sửa:
  - Loại bỏ các mẫu đa số bị phân loại sai bởi kNN hoặc các mẫu đa số có hàng xóm từ lớp khác
  - Loại bỏ các mẫu đa số gây "nhiễu loạn" cho các mẫu thiểu số
- Condensed Nearest Neighbors
  - Loại bỏ các mẫu đa số không bị kNN phân loại sai
  - Tập trung vào các mẫu khó phân loại



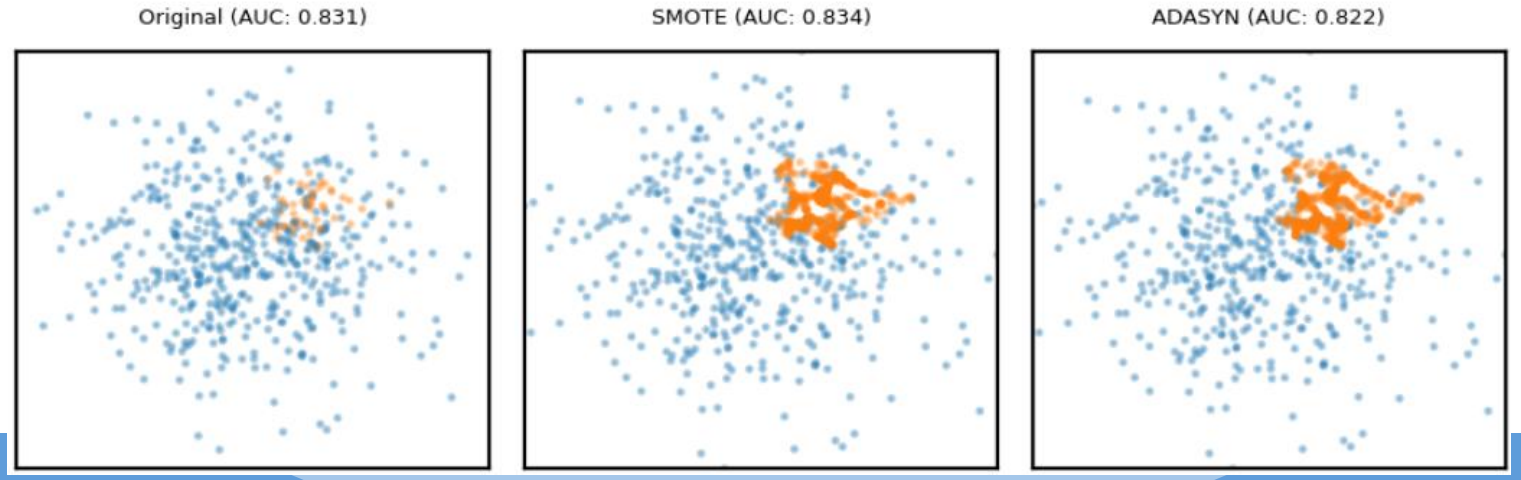
Lấy mẫu nhân rộng ngẫu nhiên (Random Oversampling)

- Sao chép các mẫu từ lớp đa số
- Lấy mẫu ngẫu nhiên từ lớp thiểu số, có thay thế, cho tới khi cân bằng mẫu
- Làm mô hình huấn luyện tốn kém hơn, không phải luôn hoạt động tốt
- Tương tự với việc cho các lớp thiểu số một trọng số cao hơn



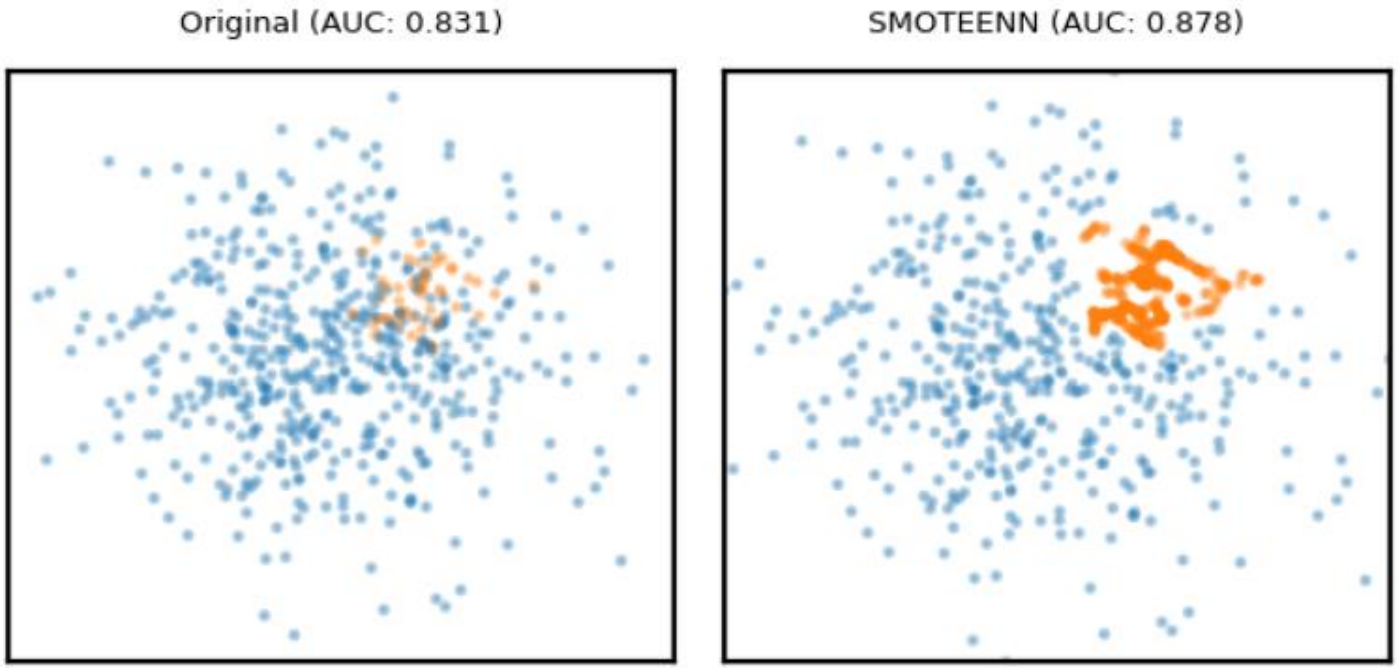
Synthetic Minority Oversampling Technique (SMOTE)

- Lặp lại chọn ngẫu nhiên một mẫu từ lớp thiểu số và mẫu thiểu số hàng xóm
  - Tạo một mẫu dữ liệu mới nằm trên đường thẳng nối giữa mẫu ban đầu và mẫu kNN đã chọn. Vị trí của mẫu mới được xác định ngẫu nhiên trên đoạn thẳng này.
- SMOTE có thể làm sai lệch phân bố dữ liệu gốc. Cần tránh tạo các mẫu nhân tạo trên tập dữ liệu kiểm thử (test set) vì điều này sẽ làm rò rỉ thông tin từ tập huấn luyện sang tập kiểm thử.
- ADASYN (Adaptive Synthetic):
  - Tương tự, nhưng bắt đầu từ các mẫu thiểu số "khó" (kNN phân loại nhầm)



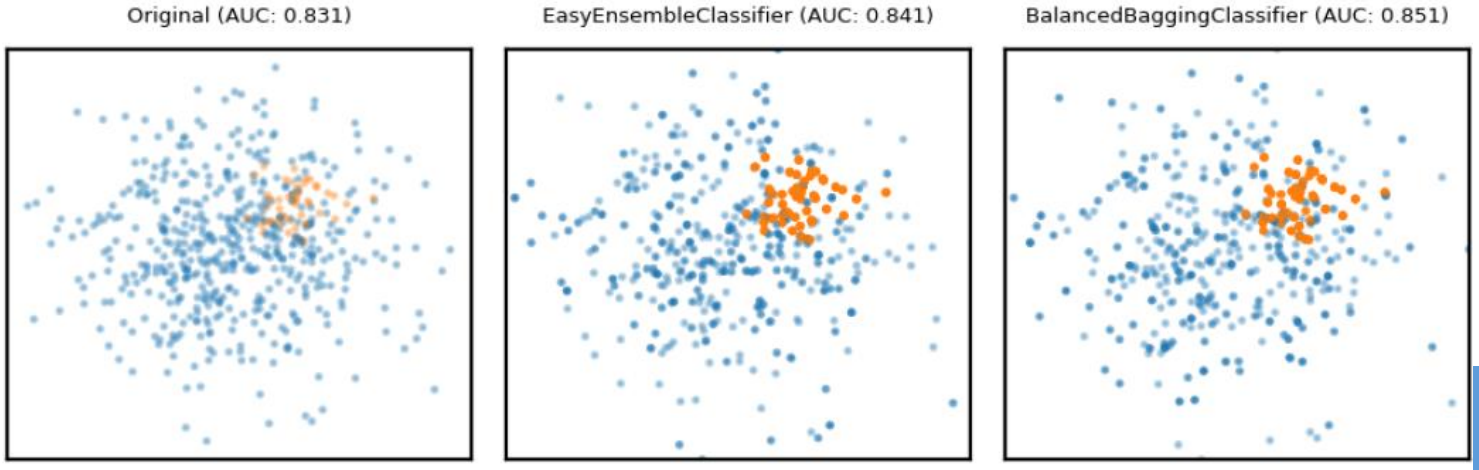
Kết hợp các kỹ thuật

- Kết hợp over- và under-sampling
- Ví dụ: oversampling với SMOTE, undersampling với Edited Nearest Neighbors (ENN)
  - SMOTE có thể tạo các mẫu nhiễu, gần với các mẫu thuộc lớp đa số
  - ENN có thể xóa hết các mẫu đa số để làm sạch không gian



Lấy mẫu kết hợp

- Kỹ thuật kết hợp lấy mẫu lại với học kết hợp (ensemble) để giải quyết vấn đề mất cân bằng lớp. Không phải là bộ tiền xử lý đơn thuần.
  - Thay vì áp dụng lấy mẫu lại trực tiếp lên toàn bộ dữ liệu, chúng ta lấy mẫu lại nhiều lần để tạo ra các tập dữ liệu con (subset) cân bằng, sau đó huấn luyện các mô hình riêng lẻ trên mỗi tập con này. Cuối cùng, kết hợp dự đoán của các mô hình con (ensemble) để đưa ra dự đoán cuối cùng.
- BalancedBagging: Tạo ra các tập dữ liệu con (bootstraps) từ dữ liệu gốc, sau đó áp dụng lấy mẫu ngẫu nhiên giảm thiểu (undersampling) trên mỗi bootstrap để đạt được sự cân bằng giữa các lớp, rồi huấn luyện mô hình.
- EasyEnsemble: Thực hiện nhiều lần lấy mẫu ngẫu nhiên giảm thiểu trực tiếp trên dữ liệu gốc, sau đó huấn luyện một mô hình trên mỗi tập dữ liệu con được lấy mẫu lại.
  - Thường sử dụng thuật toán AdaBoost làm mô hình học con, nhưng bạn có thể thay thế bằng các thuật toán khác



So sánh

- Hiệu quả của mỗi phương pháp phụ thuộc rất nhiều vào bản chất của dữ liệu (dung lượng dữ liệu, mức độ mất cân bằng,...)
  - Đối với tập dữ liệu lớn (very large dataset): Lấy mẫu ngẫu nhiên giảm thiểu (Random Undersampling) có thể là lựa chọn phù hợp.
- Ngoài phương pháp lấy mẫu lại: vẫn cần lựa chọn thuật toán học máy phù hợp với bài toán cụ thể.
- Đừng quên rằng kỹ thuật lấy mẫu lại có thể kết hợp với các phương pháp khác như gán trọng số cho các lớp và điều chỉnh ngưỡng dự đoán để cải thiện hiệu quả hơn
  - Một số kết hợp mang lại hiệu quả tốt. Ví dụ: SMOTE (tạo mẫu nhân tạo) + gán trọng số lớp + điều chỉnh ngưỡng dự đoán.

Thực hành

- Theo mô hình fit-sample: Giống với fit-transform thường thấy trong xử lý văn bản, nhưng fit-sample không chỉ biến đổi dữ liệu đầu vào (X) mà còn ảnh hưởng đến cả dữ liệu nhãn (y).
- Undersampling: RandomUnderSampler, EditedNearestNeighbours,...
- (Synthetic) Oversampling: RandomOverSampler, SMOTE, ADASYN,...
- Kết hợp: SMOTEENN,...

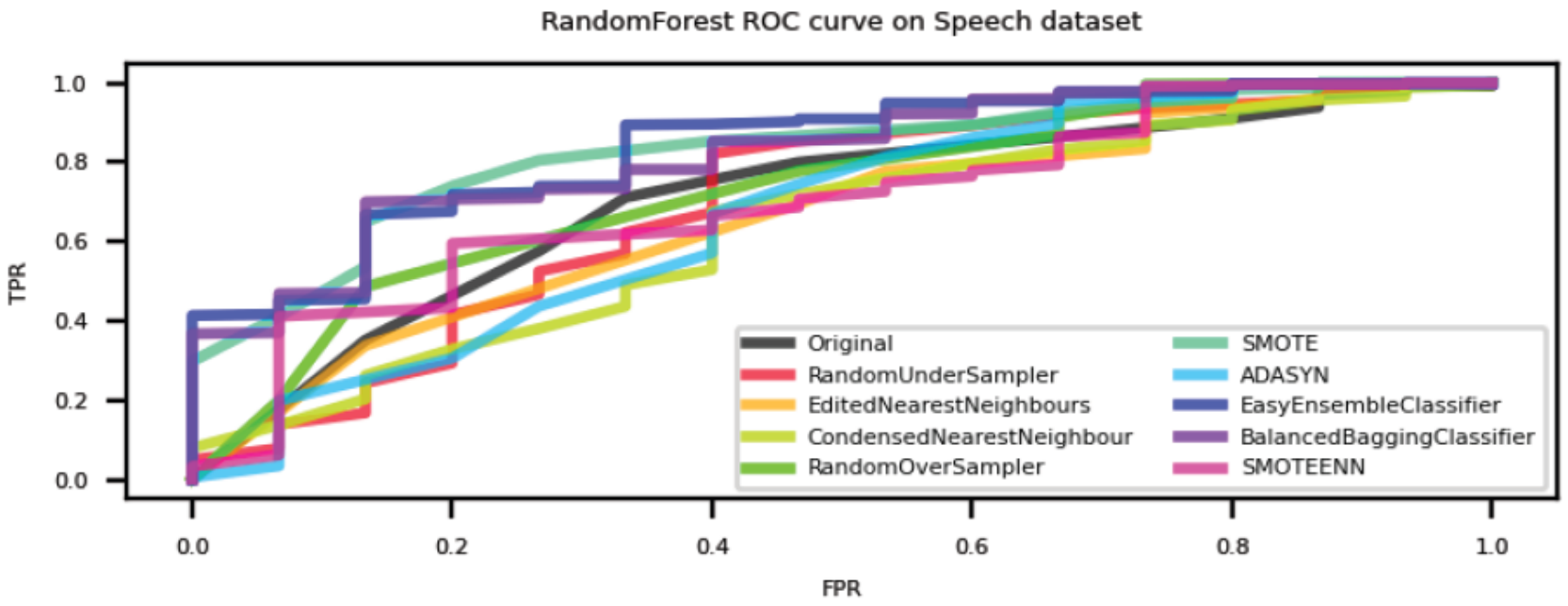
```
X_resampled, y_resampled = SMOTE(k_neighbors=5).fit_sample(X, y)
```

- Có thể sử dụng theo pipeline
- Kỹ thuật học kết hợp có thể thực hiện theo wrapper

```
clf = EasyEnsembleClassifier(base_estimator=SVC()).fit(X_train,
y_train)
```

So sánh trên dữ liệu thực

- Tác động của kĩ thuật lấy mẫu khó dự đoán
- Phương pháp tốt nhất có thể phụ thuộc vào dữ liệu và đánh đổi FP/FN
- SMOTE và các kĩ thuật học kết hợp thường hoạt động tốt



Tiền xử lý dữ liệu

- Dữ liệu trong một lĩnh vực trước khi vào hệ thống học máy phải được thu thập và biểu diễn thành dạng cấu trúc với một số đặc tính: đầy đủ, ít nhiễu, nhất quán, có cấu trúc xác định.
- Dữ liệu thu thập cho quá trình học là tập nhỏ, tuy vậy cần phản ánh dầy đủ các mặt vấn đề cần giải quyết.
- Dữ liệu thô sau khi thu thập và tiền xử lý phải giữ được sự đầy đủ các đặc trưng ngữ nghĩa – các đặc trưng ảnh hưởng đến khả năng giải quyết vấn đề.
- Tiền xử lý:
  - Chuẩn hóa là quan trọng đối với nhiều phương pháp dựa trên khoảng cách (ví dụ: kNN, SVM, Neural Nets)
  - Mã hóa danh mục là cần thiết cho các phương pháp số học (hoặc các triển khai)
  - Trích chọn đặc trưng có thể tăng tốc độ mô hình và giảm thiểu hiện tượng overfitting
  - Kỹ thuật xây dựng đặc trưng thường hữu ích cho các mô hình tuyến tính
  - Tốt hơn là bổ sung dữ liệu bị thiếu hơn là loại bỏ dữ liệu
  - Tập dữ liệu mất cân bằng cần xử lý đặc biệt để xây dựng các mô hình hữu ích

Nội dung

- Đánh giá mô hình học máy
  - Ma trận nhầm lẫn
  - Các độ đo đánh giá

Câu hỏi cho KT Tiền xử lý dữ liêu

Đánh giá mô hình học máy

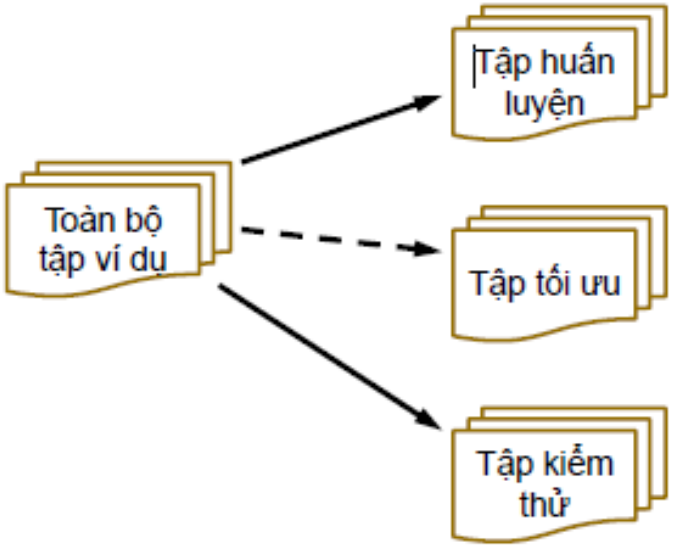
- Là việc đánh giá hiệu năng
  - Thường được thực hiện dựa trên thực nghiệm, hơn là dựa trên phân tích
  - Các đánh giá phân tích nhằm chứng minh một hệ thống là đúng đắn (correct) và hoàn chỉnh (complete) (vd: các bộ chứng minh định lý trong Logics)
- Tập trung vào việc đánh giá hiệu năng của hệ thống
  - Thực hiện một cách tự động, sử dụng một tập các mẫu (tập thử nghiệm)
  - 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?

Phương pháp đánh giá

- Chia tập dữ liệu thành 3 tập con:
  - Toàn bộ tập ví dụ/mẫu
  - Tập huấn luyện: huấn luyện mô hình, chiếm phần lớn tập dữ liệu.
  - Tập tối ưu: để điều chỉnh các siêu tham số của mô hình và tối ưu hóa mô hình.
  - Tập kiểm thử: để đánh giá hiệu suất cuối cùng của mô hình.



Phương pháp đánh giá

- 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 học 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 học máy được sử dụng, mà còn phụ thuộc vào:
  - Phân bố lớp
  - Chi phí của việc phân lớp sai
  - Kích thước của tập huấn luyện
  - Kích thước của tập kiểm thử

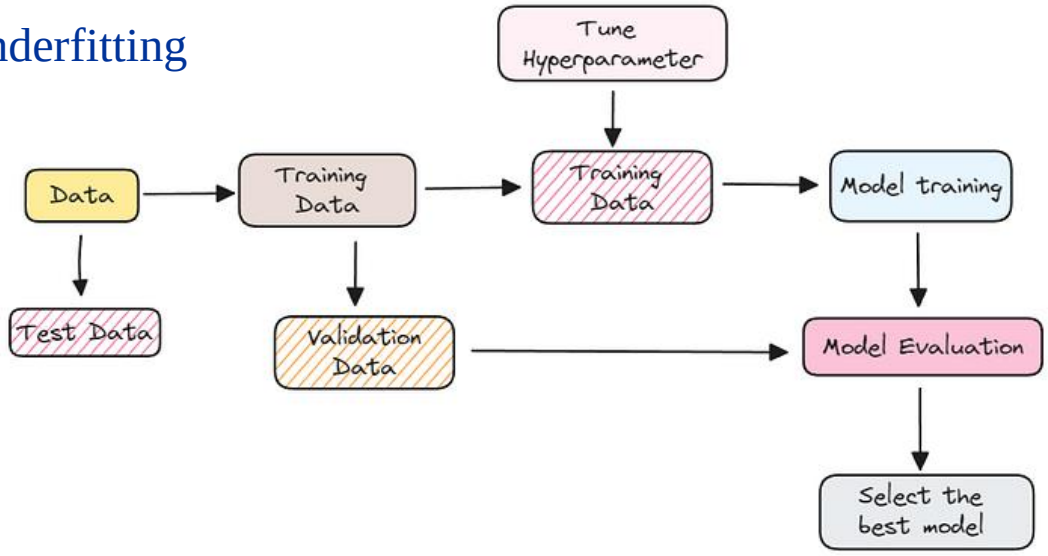
Phương pháp đánh giá

- Các phương pháp phân chia và đánh giá tập dữ liệu
  - Hold-out
  - Stratified sampling
  - Repeated hold-out
  - Cross-validation
    - k-fold
    - Leave-one-out
  - Bootstrap sampling

Phương pháp đánh giá

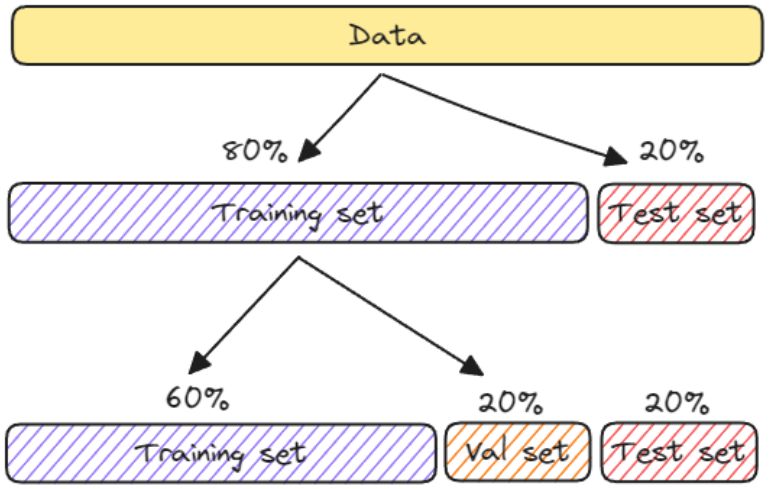
❖ Phân chia dữ liệu

- Phân chia dữ liệu thành tập huấn luyện và tập kiểm thử (train test split) là một quá trình kiểm chứng mô hình cho phép mô phỏng hiệu suất của mô hình trên dữ liệu mới.
- Tại sao?
  - Để tránh overfitting và underfitting



Phương pháp đánh giá

- Phương pháp Hold-out (splitting):
  - chia tập dữ liệu ban đầu thành hai hoặc ba phần không giao nhau:
    - Tập huấn luyện (train): Để huấn luyện mô hình.
    - Tập kiểm thử (test): Để đánh giá hiệu suất của mô hình sau khi huấn luyện.
    - Tập tối ưu (Valid) (nếu có): Để điều chỉnh siêu tham số và tối ưu hóa mô hình.
  - Phù hợp khi có nhiều dữ liệu



Phương pháp đánh giá

- Phương pháp Stratified sampling (lấy mẫu phân tầng):
  - Đối với các tập dữ liệu có kích thước nhỏ hoặc không cân xứng (unbalanced datasets), các dữ liệu trong tập huấn luyện và kiểm thử có thể không phải là đại diện
  - Ví dụ: Có (rất) ít, hoặc không có, các dữ liệu 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 mẫu (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 mẫu 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 học máy dự đoán/hồi quy (vì giá trị đầu ra của hệ thống là một giá trị số, không phải là một nhãn lớp)

Phương pháp đánh giá

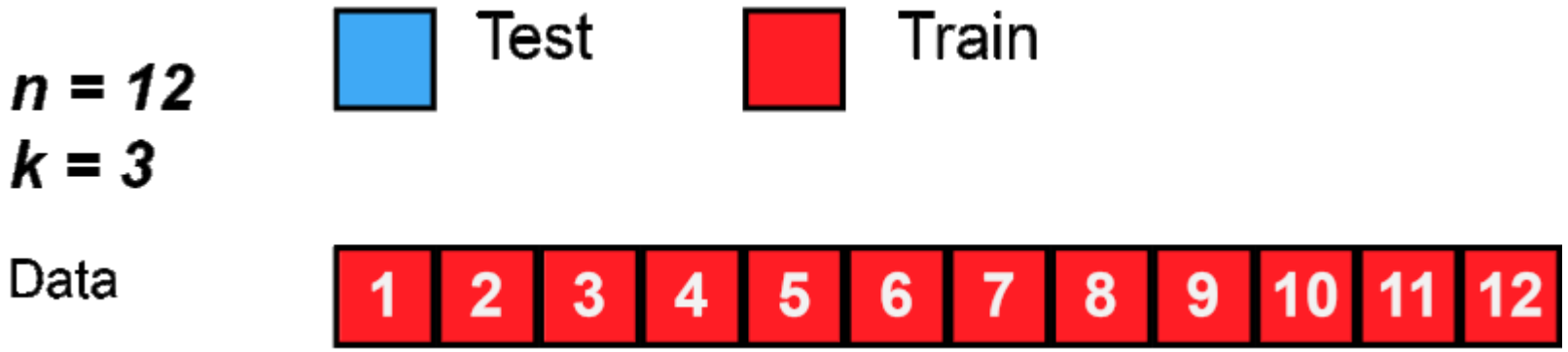
- Phương pháp Repeated hold-out:
  - Áp dụng phương pháp đánh giá Hold-out nhiều lần, để sinh ra (sử dụng) các tập huấn luyện và kiểm thử khác nhau
    - Trong mỗi bước lặp, một tỷ lệ nhất định của tập D được lựa chọn ngẫu nhiên để tạo nên tập huấn luyện (có thể sử dụng kết hợp với phương pháp lấy mẫu phân tầng – stratified sampling)
    - Các giá trị lỗi (hoặc các giá trị đối với các tiêu chí đánh giá khác) ghi nhận 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 mẫu trùng lặp (được sử dụng lại nhiều lần) trong các tập kiểm thử này

Phương pháp đánh giá

- Phương pháp Cross-validation:
  - Để tránh việc trùng lặp giữa các tập kiểm thử (một số mẫu cùng xuất hiện trong các tập kiểm thử khác nhau)
  - k-fold cross-validation, k thường là 5 hoặc 10
    - Tập dữ liệu toàn bộ 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
  - 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 dữ liệu vừa và nhỏ

Phương pháp đánh giá

- Phương pháp Cross-validation:
  - k-fold cross-validation, k thường là 5 hoặc 10
    - Tập dữ liệu toàn bộ 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



https://commons.wikimedia.org/wiki/File:KfoldCV.gif

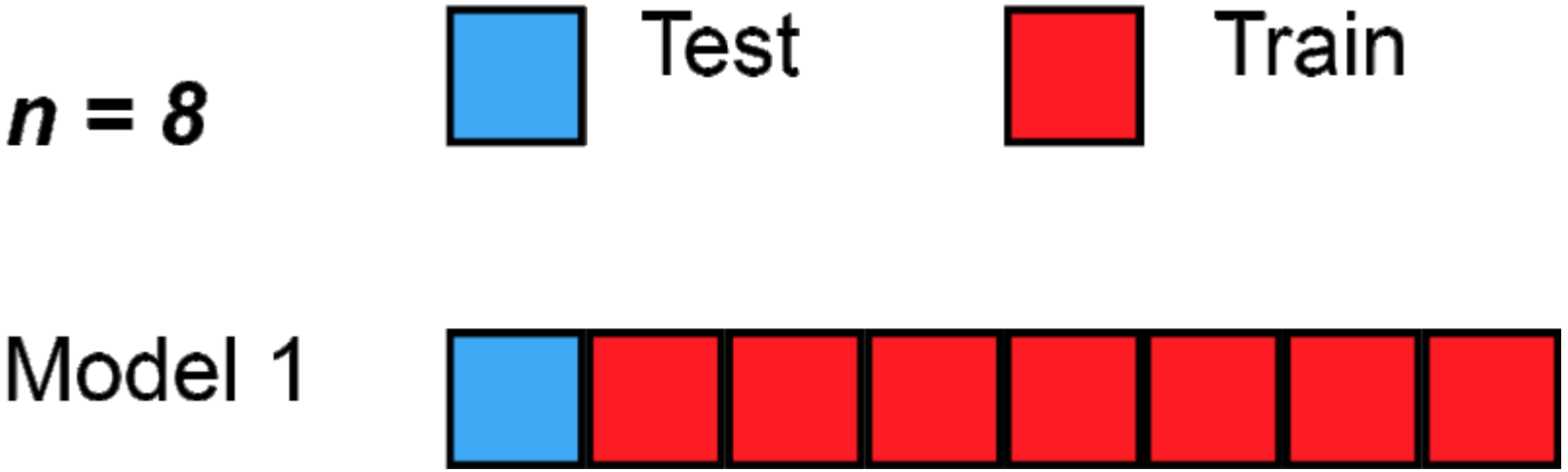
Phương pháp đánh giá

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

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

Phương pháp đánh giá

❖ Phương pháp Leave-one-out cross-validation:



Phương pháp đánh giá

- Phương pháp Bootstrap sampling:
  - Phương pháp Cross-validation sử dụng việc lấy mẫu không lặp lại
    → Đối với mỗi dụ mẫu, 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 để tạo nên tập huấn luyện
    - Giả sử tập toàn bộ D bao gồm n mẫu
    - Lấy mẫu có lặp lại n lần đối với tập D, để tạo nên tập huấn luyện D_train gồm n mẫu
      - Từ tập D, lấy ra ngẫu nhiên một mẫu x (nhưng không loại bỏ x khỏi tập D)
      - Đưa mẫu x vào trong tập huấn luyện: D_train
      - Lặp lại 2 bước trên n lần
    - Sử dụng tập D_train để huấn luyện hệ thống
    - Sử dụng tất cả các mẫu thuộc D nhưng không thuộc D train để tạo nên tập thử nghiệm: D_test

Phương pháp đánh giá

- Phương pháp Bootstrap sampling (tiếp):
  - Trong mỗi bước lặp, một mẫu có xác suất = (1 – 1/n) để không được lựa chọn đưa vào tập huấn luyện
  - Vì vậy, xác suất để một mẫu (sau quá trình lấy mẫu lặp lại – bootstrap sampling) được đưa vào tập kiểm thử là: 0.368
- Có nghĩa là:
  - Tập huấn luyện có kích thước =n) bao gồm xấp xỉ 63.2% các mẫu trong D (Lưu ý: Một mẫu 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 mẫu trong D (Lưu ý: Một mẫu 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 mẫu 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 học máy, quá trình huấn luyện hệ thống bao gồm 2 giai đoạn
  - Giai đoạn 1: Huấn luyện hệ thống (= Học hàm mục tiêu)
  - Giai đoạn 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 mẫu 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 trong giải thuật học máy 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

- Tính chính xác (Accuracy)

  →Mức độ dự đoán (phân lớp) chính xác của hệ thống (đã được huấn luyện) đối với các mẫu 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 mẫu nhiễu (lỗi) hoặc thiếu giá trị

Các tiêu chí đánh giá (tiếp)

- Khả năng mở rộng (Scalability)

  →Hiệu năng của hệ thống (vd: tốc độ học/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

Accuracy/Tính chính xác

- Đối với bài toán phân loại

  → Giá trị (kết quả) q) đầu ra của hệ thống là một giá trị định danh

$$Accuracy = \frac{1}{|D\_test|} \sum_{x \in D\_test} Identical(o(x), c(x)); \qquad Identical(a,b) = \begin{cases} 1, \text{if} & (a=b) \\ 0, \text{if otherwise} \end{cases}$$

  x: Một mẫu trong tập kiểm thử D_test

  o(x): Giá trị đầu ra (phân lớp) bởi hệ thống đối với mẫu x

  c(x): Phân lớp thực sự (đúng) đối với mẫu x

- Đối với bài toán hồi quy

  →Giá trị (kết quả) đầu ra của hệ thống là một giá trị số

$$Error = \frac{1}{|D\_test|} \sum_{x \in D\_test} Error(x); \qquad Error(x) = |d(x) - o(x)|$$

  o(x): Giá trị đầu ra (dự đoán) bởi hệ thống đối với mẫu x

  d(x): Giá trị đầu ra thực sự (đúng) đối với mẫu x

  Accuracy là một hàm đảo (inverse function) đối với Error

Ma trận nhầm lẫn (Confusion matrix)

- Còn được gọi là Contingency Table
  - Chỉ được sử dụng đối với bài toán phân loại
  - Không thể áp dụng cho bài toán hồi quy (dự đoán)

- ***TP***$_i$: Số lượng các ví dụ thuộc lớp c_i được phân loại chính xác vào lớp c_i
- ***FP***$_i$: Số lượng các ví dụ không thuộc lớp c_i bị phân loại nhầm vào lớp c_i
- ***TN***$_i$: Số lượng các ví dụ không thuộc lớp c_i được phân loại (chính xác)
- ***FN***$_i$: Số lượng các ví dụ thuộc lớp c_i- bị phân loại nhầm (vào các lớp khác c_i)

Lớp c_i		Được phân lớp **bởi hệ thống**	
		Thuộc	Ko thuộc
Phân lớp **thực sự (đúng)**	Thuộc	TP_i	FN_i
	Ko thuộc	FP_i	TN_i

Precision and Recall

- Accuracy:
  - Nhiều trường hợp thước đo Accuracy không phản ánh đúng hiệu quả của mô hình, nhất là mất cân bằng về lớp
  - Giả sử mô hình (rất kém) dự đoán tất cả 1100 email là không phải spam (với 1000 email không phải spam), thì Accuracy vẫn đạt tới 1000/1100 = 90.9%
- Độ đo Precision and Recall khắc phục được vấn đề này
- Rất hay được sử dụng để đánh giá các hệ thống phân loại văn bản

Precision and Recall (tiếp)

- Precision đối với lớp c_i:

  → Tổng số các mẫu thuộc lớp c_i được phân loại chính xác chia cho tổng số các mẫu được phân loại vào lớp c_i

  ➔ Precision sẽ cho chúng ta biết thực sự có bao nhiêu dự đoán Positive là thật sự True

$$\Pr ecision(c_i) = \frac{TP_i}{TP_i + FP_i}$$

- Recall đối với lớp c_i

  → Tổng số các mẫu thuộc lớp c_i được phân loại chính xác chia cho tổng số các mẫu thuộc lớp c_i

  ➔ Recall đo lường tỷ lệ dự báo chính xác các trường hợp positive trên toàn bộ các mẫu thuộc nhóm positive.

  ➔ Recall cao đồng nghĩa với việc True Positive Rate cao, tức là tỷ lệ bỏ sót các điểm thực sự là positive là thấp

$$\Re call(c_i) = \frac{TP_i}{TP_i + FN_i}$$

Precision and Recall (tiếp)

- Precision sẽ cho chúng ta biết thực sự có bao nhiêu dự đoán Positive là thật sự True
- Recall đo lường tỷ lệ dự báo chính xác các trường hợp positive trên toàn bộ các mẫu thuộc nhóm positive.

Precision and Recall (tiếp)

- Làm thế nào để tính toán được giá trị Precision và Recall (một cách tổng thể) cho toàn bộ các lớp $C=\{c_i\}$?
- Trung bình vi mô (Micro-averaging):

$$\Pr ecision = \frac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} (TP_i + FP_i)} \qquad \Re call = \frac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} (TP_i + FN_i)}$$

- Trung bình vĩ mô (Macro-averaging)

$$\Pr ecision = \frac{\sum_{i=1}^{|C|} \Pr ecision(c_i)}{|C|} \qquad \Re call = \frac{\sum_{i=1}^{|C|} \Re call(c_i)}{|C|}$$

Precision and Recall (tiếp)

- Rất hay được sử dụng để đánh giá các hệ thống phân loại văn bản
- Precision đối với lớp c_i:

  → Tổng số các mẫu thuộc lớp c_i được phân loại chính xác chia cho tổng số các mẫu được phân loại vào lớp c_i
- Recall đối với lớp c_i

  → Tổng số các mẫu thuộc lớp c_i được phân loại chính xác chia cho tổng số các mẫu thuộc lớp c_i

F1-score

- Tiêu chí đánh giá F1 là sự kết hợp của 2 tiêu chí đánh giá Precision và Recall

$$F_1 = \frac{2.\mathrm{Pr}\,ecision.\mathrm{Re}\,call}{\mathrm{Pr}\,ecision + \mathrm{Re}\,call} = \frac{2}{\frac{1}{\mathrm{Pr}\,ecision} + \frac{1}{\mathrm{Re}\,call}}$$

- F1 là một trung bình điều hòa (harmonic mean) của các tiêu chí Precision và Recall
  - F1 có xu hướng lấy giá trị gần với giá trị nào nhỏ hơn giữa 2 giá trị Precision và Recall
  - F1 có giá trị lớn nếu cả 2 giá trị Precision và Recall đều lớn

Sensitivity – Specificity

- Sensitivity và Specificity là 2 độ đo được sử dụng trong các bài toán phân loại liên quan đến y tế, sinh học và an toàn thông tin. Chúng được định nghĩa như sau:

$$Sensitivity = Recall = \frac{True\ Positive}{\text{True Positive+False Negative}}$$

$$Specificity = \text{True Negative Rate} = \frac{True\ Negative}{\text{True Negative+False Positive}}$$

- Câu hỏi: dự đoán chính xác như thế nào?
  - Tình huống 1: Một người hoàn toàn khỏe mạnh nhưng kết quả xét nghiệm lại chỉ ra "người này có bệnh". Tình huống này được gọi là "Dương tính giả".
  - Tình huống 2: Một người mang bệnh nhưng kết quả xét nghiệm lại cho thấy "người này khỏe mạnh". Tình huống này được gọi là "Âm tính giả".

Sensitivity – Specificity

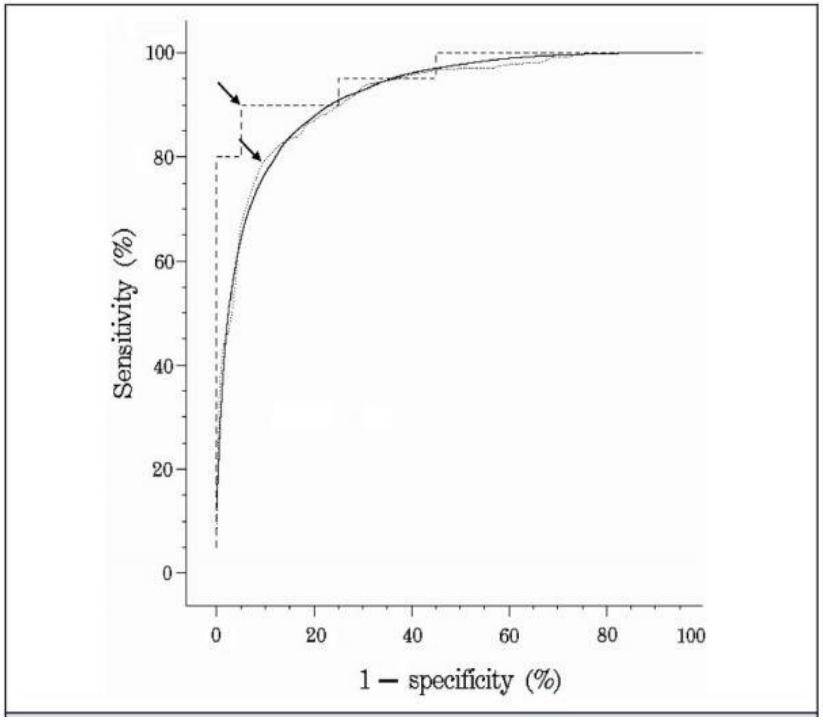
❖ Các tình huống có thể xảy ra:

	Sự thật		
		Người có bệnh	*Người khỏe*
Xét nghiệm	*Dương tính*	A (chẩn đoán chính xác - True positive/TP)	B (dương tính giả - false positive/FP)
	Âm tính	C (âm tính giả - false negative/FN)	D (chẩn đoán chính xác - True negative/TN

❖ Độ đo:

- Độ nhạy phản ánh khả năng một người có bệnh được chẩn đoán chính xác, tức là độ nhạy = A / (A+C) = TP/(TP+TN).
  - Độ nhạy cao, kết quả có độ chính xác cao ➔ "âm tính giả" càng khó xảy ra ➔ Thà bắt lầm còn hơn bỏ sót
  - Độ nhạy = 100% ➔ nếu kết quả âm tính thì chắc chắn yên tâm là không có bệnh
- Độ đặc hiệu phản ánh khả năng một người khỏe mạnh được chẩn đoán chính xác, tức là độ đặc hiệu = D / (B+D) = .
  - Độ đặc hiệu cao ➔ "dương tính giả" càng khó xảy ra
  - độ đặc hiệu 100% mà trả kết quả dương tính thì có thể nói là "không còn nghi ngờ gì nữa"

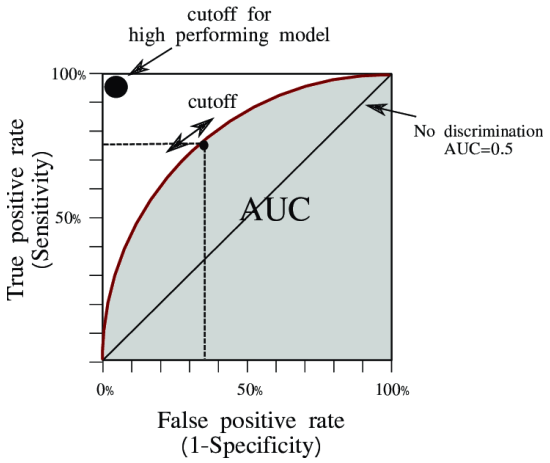
Sensitivity – Specificity



- Mô hình hoàn hảo:
  - độ nhạy và độ đặc hiệu đều đạt 100%, tức là tình trạng "âm tính giả" và "dương tính giả" đều không xảy ra
- Thực tế:
  - khi ta cố làm tăng độ nhạy của một phương pháp thì độ đặc hiệu sẽ giảm và ngược lại , vì vậy cần có sự lựa chọn.
  - Phối hợp 2 mô hình:
    - Độ nhạy cao: ứng dụng cho kiểm tra nhanh
    - Độ đặc hiệu cao: ứng dụng trong mang tính "khẳng định lại" ➔ đưa ra quyết định cuối cùng

AUC – ROC curve

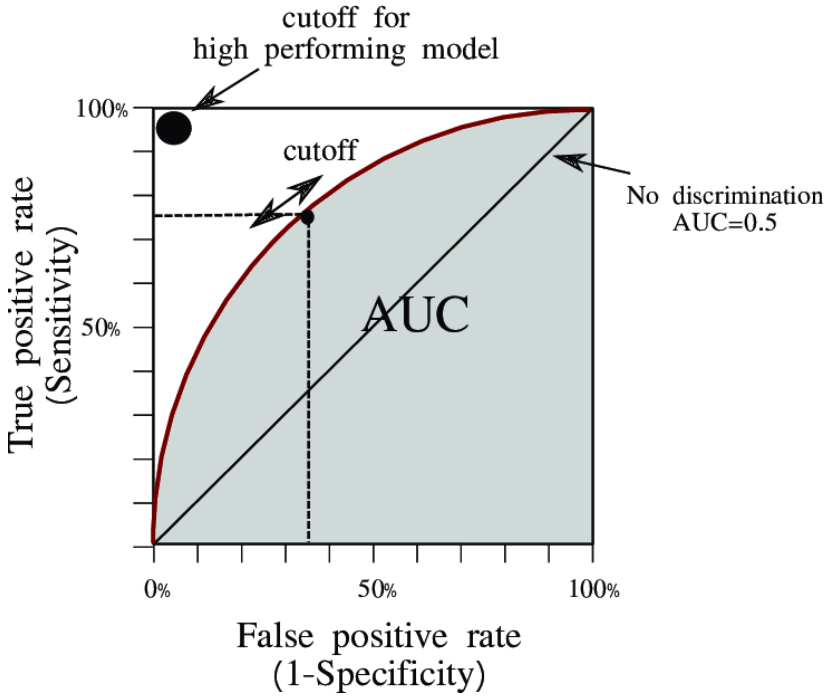
- ROC là đường cong biểu diễn khả năng phân loại của một mô hình phân loại tại các ngưỡng threshold. Đường cong này dựa trên hai chỉ số :
  - TPR (true positive rate): chính là recall/sensitivity.
  - FPR (false positive rate): Tỷ lệ dự báo sai các trường hợp thực tế là negative thành positive trên tổng số các trường hợp thực tế là negative.
    - FPR = 1 - specificity



Statistical Analysis of the Evolutive Effects of Language Development in the Resolution of Mathematical Problems in Primary School Education

AUC – ROC curve

- AUC:
  - là chỉ số được tính toán dựa trên đường cong ROC (receiving operating curve) nhằm đánh giá khả năng phân loại của mô hình tốt như thế nào?
  - AUC (area under curve) có giá trị nằm trong khoảng [0, 1]. AUC càng lớn thì đường cong ROC có xu hướng tiệm cận đường thẳng.
  - Khi đường cong ROC nằm sát với đường chéo đi qua hai điểm (0, 0) và (1, 1), mô hình sẽ tương đương với một phân loại ngẫu nhiên.



Statistical Analysis of the Evolutive Effects of Language Development in the Resolution of Mathematical Problems in Primary School Education

Lựa chọn mô hình học được

- 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 dữ liệu 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

Câu hỏi cho Đánh giá hiệu năng mô hình ML

Kết luận

- Giới thiệu các bước Tiền xử lý dữ liệu
  - Làm sạch dữ liệu
  - Trích chọn đặc trưng
  - Giảm chiều dữ liệu
  - Xử lý dữ liệu không cân bằng
- Cách đánh giá mô hình học máy dựa trên:
  - Ma trận nhầm lẫn
  - Các độ đo đánh giá

Bài tập

1. Thực hành các kĩ thuật tiền xử lý, theo Hướng dẫn thực hành.
2. Thực hành đánh giá hiệu năng dựa trên các cách chia dữ liệu, các độ đo khác nhau, theo Hướng dẫn thực hành.