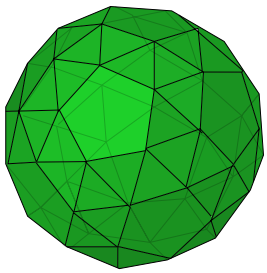
Projects in Mathematics and Applications

NAIVE BAYES

Ngày 27 tháng 8 năm 2018

Nguyễn Nguyễn [*]
Nguyễn Tấn Đạt [‡]
[†]Huỳnh Phạm Phương Mai
[§]Nguyễn An Trường



[*]Trường Phổ thông năng khiếu - ĐHQG TPHCM
[†]Trường THPT Chuyên Lê Hồng Phong - TPHCM
[‡]Trường THPT Chuyên Sư phạm - Hà Nội
[§]Trường THPT Chuyên Nguyễn Tất Thành - Kon Tum

Mục lục

Lời cảm ơn

Nhóm 4 xin chân thành gửi lời cảm ơn đến Ban tổ chức trại hè Toán học PiMA (Projects in Mathematics and Applications) 2018 đã tạo điều kiện thuận lợi nhất cho trại sinh có những nghiên cứu sâu về Toán học ứng dụng nói chung và Machine Learning nói riêng. Không chỉ có những kiến thức Toán, PiMA còn trang bị cho trại sinh những kĩ năng làm việc nhóm, nghiên cứu khoa học và báo cáo dự án trước những giáo sư, cố vấn có đầy kinh nghiệm trong ngành. Đây là những điều chúng tôi ít có cơ hội được tiếp xúc ở môi trường THPT.

Sau ba năm thành công của trại hè, chúng tôi mong muốn PiMA có thể duy trì và phát triển hơn nữa để truyền được đam mê và ham muốn theo đuổi Toán học cho các bạn học sinh cấp 3 trên khắp cả nước.

Ngoài ra, chúng tôi cũng xin gửi lời cảm ơn đặc biệt đến hai mentor chính của nhóm là anh Phạm Nguyễn Mạnh, anh Nguyễn Trường Hải, cùng anh Phạm Hoàng Nhật đã theo sát nhóm trong suốt thời gian nghiên cứu và hoàn thành dự án này.

Cuối cùng, nhóm chúng tôi xin gửi lời cảm ơn đến Trường Đại học Khoa học và Tự nhiên TPHCM đã tạo điều kiện, cung cấp cơ sở vật chất trong suốt 10 ngày diễn ra trại hè này.

Tóm tắt nội dung

Trong bài báo cáo này, nhóm 4 chủ yếu tập trung vào phương pháp phân loại Naive Bayes Classifier (NBC), bắt nguồn từ lí thuyết Bayes - Naive Bayes, với giả sử là các tính trạng của điểm dữ liệu tương đối độc lập. Đặc biệt là áp dụng NBC vào việc phát hiện lừa đảo thẻ tín dụng (fraud detection) và giải quyết các vấn đề phát sinh như dữ liệu không cân bằng (imbalance data), cập nhật dữ liệu (update data),v...v...

Naive Bayes đã được nghiên cứu sâu rộng từ thập niên 50 của thế kỉ XX, là một phương pháp phổ biến trong việc phân loại văn bản (phân loại dữ liệu rác hoặc hợp lệ, thể thao hay chính trị, v...v...) với tính trạng là mật độ xuất hiện của các từ. Trong Machine Learning, hệ thống phân loại naive Bayes là một họ của những phân loại xác suất đơn giản dựa trên định lí Bayes với giả sử là giữa các tính trạng có tính độc lập (ngây thơ) cao. Tiến hành pre-processing hiệu quả, phương pháp naive Bayes sẽ cực kì hữu hiệu khi áp dụng cùng các phương pháp nâng cao hơn như Support Vector Machine (SVM). Ngoài ra, phương pháp này còn được áp dụng vào việc chẩn đoán bệnh tự động.

Trong Thống kê và Ngôn ngữ Khoa học máy tính, mô hình naive Bayes được biết đến dưới nhiều tên gọi, bao gồm Bayes đơn giản và Bayes độc lập. Mọi tên gọi này đều liên quan đến ứng dụng của định lí Bayes trong phân loại quy tắc quyết định, tuy nhiên, Naive Bayes không hẳn là phương pháp Bayesian.

Phép phân loại naive Bayes có thời gian huấn luyện và kiểm tra dữ liệu rất nhanh do giả sử về tính độc lập giữa các thành phần, nếu biết class. Nếu giả sử về tính độc lập được thoả mãn (dựa vào bản chất của dữ liệu), phương pháp này được cho là cho kết quả tốt hơn so với SVM và logistic regression khi có ít dữ liệu huấn luyện. NBC có thể hoạt động với các vector tính trạng mà một phần là liên tục (sử dụng Gaussian Naive Bayes), phần còn lại ở dạng rời rạc (sử dụng Multinomial hoặc Bernoulli). Khi sử dụng Multinomial Naive Bayes, Laplace smoothing thường được sử dụng để tránh trường hợp 1 thành phần trong dữ liệu kiểm tra chưa xuất hiện ở dữ liệu huấn luyện.

Một ưu điểm khác của naive Bayes là phương pháp này chỉ yêu cầu một số lượng nhỏ dữ liệu huấn luyện để dự đoán tham số cần thiết cho việc phân loại.

1 Giới thiệu mô hình Naive Bayes classifier

Naive Bayes là một mô hình Machine Learning (ML) dùng để phân loại dữ liệu bằng cách sử dụng **Maximum a Posteriori Estimation**. Navie Bayes hình thành dựa trên định lí Bayes (Bayes's Theorem), lý thuyết đơn giản, trực quan và là nền tảng cho các mô hình ML phức tạp hơn sau này.

1.1 Maximum a Posteriori Estimation

Trước khi tìm hiểu kỹ về **Naive Bayes**, ta sẽ nhắc lại khái niệm về **Ước lượng cực đại hậu nghiệm** (Maximum a Posteriori Estimation - MAP) vì đây là nền tảng quan trọng của **Naive Bayes classifier**

Định nghĩa 1.1. Cho một họ phân phối $D(\theta)$ với θ trong đó θ là một biến ngẫu nhiên có phân phối tiền nghiệm $\theta \sim P$. Cho một biến ngẫu nhiên $X \sim D(\theta)$ và n giá trị $x_1, x_2, ..., x_n$ thu được từ X. Khi đó **ước lượng cực đại hậu nghiệm** của θ là giá trị θ_0 sao cho xác suất.

$$P(\theta = \theta_0 | (X_1 = x_1)(X_2 = x_2)...(X_n = x_n))$$

là lớn nhất trong đó $X_i \sim D(\theta_0)$ là giá trị của X trong lần thử thứ i.
Biểu thức ở trên là xác suất để kết quả n lần thử ra được $x_1, x_2, ..., x_n$ và được gọi là **độ hợp lý hậu nghiệm của** $D(\theta_0)$.

Do đó ta cần ước lượng cực đại hậu nghiệm cho θ là giá trị:

$$\theta^* = \arg\max_{\theta \sim P} P(\theta | x_1, x_2, ..., x_n)$$

Từ đó sử dụng công thức Bayes, ta có:

$$\begin{aligned}
\theta^* &= \arg\max_{\theta \sim P} P(\theta | x_1, x_2, ..., x_n) \\
&= \arg\max_{\theta \sim P} \frac{P(x_1, x_2, ..., x_n | \theta).P(\theta)}{P(x_1, x_2, ..., x_n)} \\
&= \arg\max_{\theta \sim P} P(x_1, x_2, ..., x_n | \theta).P(\theta)
\end{aligned}$$

Giả thiết rằng các điểm dữ liệu độc lập nhau, kết hợp việc sử dụng hàm logarit, ta có:

$$\begin{aligned}
\theta^* &= \arg\max_{\theta \sim P} P(\theta | x_1, x_2, ..., x_n) \\
&= \arg\max_{\theta \sim P} P(x_1, x_2, ..., x_n | \theta).P(\theta) \\
&= \arg\max_{\theta \sim P} P(x_1|\theta).P(x_2|\theta)...P(x_n|\theta)P(\theta) \\
&= \arg\max_{\theta \sim P} \left[\sum_{i=1}^{n} \log(P(x_i|\theta)) + \log(P(\theta)) \right]
\end{aligned}$$

Để có thể ước lượng được giá trị θ^* để giá trị trên đạt giá trị lớn nhất, phụ thuộc vào việc xác định phân phối tiền nghiệm $P(\theta)$ và tính khả dĩ (likelihood) của dữ liệu $P(x_i|\theta)$. Đối với phân phối tiền nghiệm, việc xác định trước phân phối tiền nghiệm cho θ là rất quan trọng để **MAP** có thể cho kết quả chính xác. Trong thực tế, việc chọn các phân phối này thường không thể tìm được một cách chính xác. Tuy nhiên, trong thực nghiệm, một phân phối tiền nghiệm thường gặp là **phân phối beta** hoặc **phân phối chuẩn**:

Phân phối beta xác định như sau:

$$(P(\theta)) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\theta^{a-1}(1-\theta)^{b-1}$$

Trong đó a,b đóng vai trò là các siêu tham số được cố định từ trước. Khi chưa có kiến thức về phân phối tiền nghiệm, người ta thường chọn $a = b = 1$.

Đối với **phân phối chuẩn**, mỗi phân phối đặc trựng bởi hai tham số là giá trị trung bình μ và phương sai σ^2. Ta có biểu diễn của một phân phối chuẩn như sau:

$$f(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

1.2 Mô hình phân loại Naive Bayes

Trong thực tế, khi chúng ta làm việc với những bài toán xác suất thống kê và các bài toán phân loại, một trong những phương pháp phổ biến là sử dụng xác suất. Ý tưởng cơ bản là với một dữ liệu của các tính trạng cho trước, ta sẽ xếp dữ liệu ấy vào một lớp phân loại nào cụ thể mà ở đấy, nó có khả năng xảy ra cao nhất so với những lớp khác. Phương pháp phổ biến để ước lượng được yếu tố xác suất này là sử dụng mô hình phân loại **Naive Bayes**.
Ta phát biểu bài toán phân loại của chúng ta như sau:

Định nghĩa 1.2. (Bài toán phân loại)
Cho một **hàm mục tiêu** chưa biết $c : \mathrm{R}^d \to C$, trong đó R^d là không gian đầu vào và $C = c_1, ..., c_m$ là m lớp phân loại (**class**), và n điểm dữ liệu huấn luyện (training data):

$$D = \{\langle \mathbf{t}_1, c(\mathbf{t}_1)\rangle ; \langle \mathbf{t}_2, c(\mathbf{t}_2)\rangle ; ...; \langle \mathbf{t}_n, c(\mathbf{t}_n)\rangle\}$$

Với $\mathbf{t}_i \in \mathrm{R}^d$, $c(\mathbf{t}_i) \in C$. Với một dữ liệu $\mathbf{x}$ chưa được huấn luyện, hãy ước lượng một giá trị phân loại $c(\mathbf{t}) \in C$ phù hợp nhất.

Ý tưởng cơ bản là tìm một class sao cho **khả năng** x **thuộc class đó là cao nhất** dựa trên sự phân loại của các điểm $\mathbf{t}_1, \mathbf{t}_2, .., \mathbf{t}_n$. Đến đây rõ ràng **MAP** là phương pháp phù hợp để thực hiện ý tưởng. Thuật toán **Naive Bayes** của chúng ta sử dụng MAP làm công cụ chính.
Giả sử chúng ta đã được cho các lớp phân loại c và n dữ liệu huấn luyện. Khi đó một điểm dữ liệu mới $\mathbf{x}$ sẽ được phân loại theo công thức:

$$\begin{aligned} c^*(\mathbf{x}) &= \arg\max_{c\in C} P(c(\mathbf{x}) = c|x_1, x_2, ..., x_d) \\ &= \arg\max_{c\in C} P(x_1, x_2, ..., x_d|c(\mathbf{x}) = c).P(c) \end{aligned}$$

Lý do mà phương pháp này được gọi là Naive Bayes, vì bản chất của nó là sử dụng ước lượng hậu nghiệm MAP - ước lượng dựa chủ yếu trên định lý Bayes về xác suất. Nhưng để ước lượng giá trị cuối một cách dễ dàng nhất, một cách ngây thơ (naive), ta sẽ giả sử điều kiện là các biến $x_1, ..., x_d$ là **độc lập** với nhau, tức giả sử rằng các **thuộc tính** của dữ liệu là độc lập với nhau. Khi đó ta có thể biến đổi như sau:

$$\begin{aligned} c^*(\mathbf{x}) &= \arg\max_{c\in C} P(x_1, x_2, ..., x_d|c(\mathbf{x}) = c).P(c) \\ &= \operatorname{argmax}_{c\in C} \prod_{i=1}^{d} P(x_i|c).P(c) \\ &= \operatorname{argmax}_{c\in C} \sum_{i=1}^{d} \log(P(x_i|c)) + \log(P(c)) \end{aligned}$$

Như vậy điều quan trọng để ước lượng được $c(\mathbf{x})$ tối ưu là cần phải ước lượng được các phân phối tiền nghiệm và tính khả dĩ của thuộc tính với lớp tương ứng. Hai thông số này thay đổi tùy vào dữ liệu nhập và quá trình cập nhật dữ liệu, chẳng hạn như phân phối tiền nghiệm $P(c)$ có thể thay đổi mỗi khi dữ liệu mới cập nhật. Việc cập nhật dữ liệu, tính toán các phân phối phù hợp, chọn các điểm dữ liệu phù hợp cho mô hình Naive Bayes là điều cần thiết để có những ước lượng chính xác nhất cho mô hình.
Một cách đơn giản để ước lượng $P(c)$ và $P(t_i|c)$ khá trực quan là thống kê từ chính những điểm dữ liệu của D. Ví dụ, để ước lượng $P(c)$ (phân phối tiền nghiệm của class c), ta có thể dùng tỉ lệ ước lượng giữa số phần tử của c với số phần tử tổng cộng, tức là:

$$P(c) = \frac{|\{1 \leq k \leq n : c(\mathbf{t}_k) = c\}|}{n}$$

Với xác suất khả dĩ, ta có thể tính bằng tỉ lệ các dữ liệu có thuộc tính thứ i gần bằng x_i trong các điểm thuộc class c:

$$P(c) = \frac{|\{1 \leq k \leq n : (c(\mathbf{t}_k) = c) \wedge (\mathbf{t}_k(i) \approx x_i)\}|}{|\{1 \leq k \leq n : c(\mathbf{t}_k) = c\}|}$$

Phương pháp ước lượng này có nhiều nhược điểm cần khắc phục. Ta có thể sử dụng nhiều cách ước lượng hiệu quả hơn như phân phối beta để gán cho phân phối tiền nghiệm ban đầu.
Một lưu ý rằng trong thực tế, các thuộc tính của dữ liệu hầu như không độc lập với nhau. Để tiếp tục ước lượng như trên thì ta phải xử lý các thuộc tính bằng nhiều phương pháp để tăng tính độc lập và cho kết quả chính xác cao hơn.

1.3 Ứng dụng của mô hình Naive Bayes Classifier

1. Thường được sử dụng trong các bài toán phân loại văn bản (Text Classification)
2. Do tính độc lập của các sự kiện nên thời gian huấn luyện và kiểm tra dữ liệu nhanh
3. Khi có ít dữ liệu huấn luyện thì cho kết quả tốt hơn phương pháp Support Vector Machine (SVM) và logistic regression
4. Có thể áp dụng đồng thời với các vector tính trạng (features vector) có một phần là liên tục và một phần là rời rạc

2 Phân loại Bayes trên các Class liên tục

Với mô hình phân loại Naive Bayes đã được giới thiệu, ta đang giả sử rằng ta có một số hữu hạn các Class (ví dụ m Class $c_1, c_2, ..., c_m$) hoặc vô hạn các Class đếm được (Class rời rạc). Câu hỏi đặt ra là nếu ta xét các Class phân loại có tính chất liên tục?

Ví dụ, giả sử ta được cho n dữ liệu với d các tính trạng như chiều cao, giới tính, cân nặng,... làm thế nào để ước lượng được tuổi của người đấy? Ở đây, độ tuổi là một đại lượng mang tính liên tục.

Khi ta làm việc với mô hình phân loại Bayes, hai yếu tố chính để ước lượng được xác suất hậu nghiệm (posterior) chính là xác định được xác suất tiền nghiệm (prior) và xác suất có điều kiện hay tính khả dĩ (likelihood). Một phân phối xác suất thường gặp trên phân phối liên tục (đặc biệt khi khoảng liên tục càng lớn) là phân phối chuẩn (phân phối Gauss). Như vậy, ta sẽ tìm cách ước lượng phân phối hậu nghiệm ứng với dữ liệu và class liên tục cho trước.

Bây giờ, giả sử rằng, ta đang cần phải ước lượng một phân phối hậu nghiệm $p(\theta|x_1, ..., x_d)$ với $x_1, ..., x_d$ là các dữ liệu tính trạng cho trước và θ đặc trưng cho một Class liên tục. Giống như mô hình Bayes, để ước lượng tham số θ để phân loại điểm dữ liệu $(x_1, ...x_d)$ vào class tốt nhất, ta có:

$$\theta^* \sim p(\theta) \prod_{i=1}^{d} p(x_d|\theta)$$

Do θ là một biến liên tục, ta có thể giả sử rằng $p(\theta)$ xác định bởi một phân phối chuẩn với giá trị trung bình và phương sai là hai giá trị biết trước. Tức là

$$\theta \sim N(\mu_0, \sigma_0^2)$$

với μ_0, σ_0^2 tương ứng là giá trị trung bình và phương sai của θ. Khi đó, với mỗi i, đặt:

$$x_i|\theta \sim N(\theta, \sigma^2)$$

Như vậy ta có:

$$\begin{aligned}
\theta^* &\sim p(\theta) \prod_{i=1}^{d} p(x_d|\theta) \\
&\sim \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left(-\frac{(\theta-\mu_0)^2}{2\sigma_0^2}\right) . \left(\prod_{i=1}^{d} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x_i-\theta)^2}{2\sigma^2}\right)\right) \\
&\sim \left(-\frac{(\theta-\mu_0)^2}{2\sigma_0^2} - \sum_{i=1}^{d} \frac{(x_i-\theta)^2}{2\sigma^2}\right)
\end{aligned}$$

Tuy nhiên, giả sử rằng thay vì xét một Class mang một giá trị liên tục cụ thể, ta có thể xét một Class là một đoạn liện tục. Khi đó, có thể số Class là hữu hạn nhưng mỗi Class là liên tục, và thay vì tính giá trị cụ thể để $p(\theta|x_1, ..., x_d)$ tối ưu, ta phải tính phân phối hậu nghiệm cho dữ liệu ứng với cả đoạn liên tục đó. Khi đó để tính được phân phối hậu nghiệm của dữ liệu cho trước đối với Class này, ta cần phải tính với từng phần tử liên tục và đặc trưng cho nó bởi một hàm mật độ xác suất $D(\theta)$ như sau:

$$p(\theta|x_1, ..., x_d) = \frac{p(\theta) \prod_{i=1}^{d} p(x_d|\theta)}{p(x_1, ..., x_d)} = c.f(\theta; \mu_0, \sigma_0). \prod_{i=1}^{d} .f(x_i; \theta, \sigma) = D(\theta)$$

Với $f(x; a, b)$ là kí hiệu của hàm phân phối Gauss ứng với biến x và hai tham số giá trị trung bình a và độ lệch chuẩn b, $c = \frac{1}{p(x_1,...,x_d)}$. Để ý trong biểu thức vế phải, do dữ liệu $(x_1, ..., x_d)$ và các tham số $\mu, \sigma, \mu_0, \sigma_0$ đều cố định, nên biểu thức có thể được đặc trưng bởi một hàm mật độ xác suất theo biến θ là $D(\theta)$. Như vậy giả sử Class C ta đang xét đặc trưng bởi tính chất $a \leq \theta \leq b$ thì khi đó:

$$p(a \leq \theta \leq b|x_1, ..., x_d) = \int_a^b D(\theta)d\theta$$

Như vậy ta có thể ước lượng được phân phối hậu nghiệm cho các dữ liệu ứng với class liên tục. Trong thực tế, các class phân loại có thể vừa liên tục hoặc rời rạc (tức là có class rời rạc, có class liên tục). Với mỗi loại class có một cách tính phân phối hậu nghiệm với một dữ liệu tương ứng. Sau đó ước lượng hậu nghiệm cực đại theo công thức:

$$c^*(\mathbf{x}) = \underset{c \in C}{\operatorname{argmax}}\ p(c(\mathbf{x}) = c|x_1, ..., x_d)$$

Với C là tập các class phân loại.

3 Cập nhật dữ liệu (updating data) bằng suy luận Bayes

Ta thấy rằng, việc cập nhật dữ liệu, tức các quan sát, bằng chứng (evidence), ít nhiều ảnh hưởng đến xác suất tiền nghiệm ban đầu. Ví dụ, giả sử chủ tọa đang xét một người có tội hay không với xác suất tiền nghiệm 80% có tội và 20% vô tội. Sau khi ta cập nhật một số quan sát, bằng chứng chứng minh người này vô tội, thì xác suất tiền nghiệm ban đầu phải thay đổi theo (xác suất có tội giảm và vô tội tăng). Ở đây ta có thể hiểu sự thay đổi xác suất tiền nghiệm giống như sự thay đổi niềm tin, tức mức độ tin tưởng vào các giả thuyết ban đầu.

Định lý Bayes điều chỉnh các xác suất khi được cho bằng chứng mới theo cách sau đây:

$$P(H_0|E) = \frac{P(E|H_0).P(H_0)}{P(E)}$$

Trong đó:

- E bằng chứng mới (updated data)
- H_0 giả thuyết (null hypothesis) được suy luận trước khi có E
- $P(H_0)$ xác suất tiền nghiệm của H_0
- $P(E|H_0)$ xác suất có điều kiện việc cập nhật bằng chứng E biết H_0 đúng
- $P(E)$ xác suất biên của E
- $P(H_0|E)$ xác suất hậu nghiệm của H_0 nếu biết E.

Hệ số $\frac{P(E|H_0)}{P(E)}$ thể hiện ảnh hưởng của bằng chứng với độ tin tưởng giả thuyết. Nếu cập nhật bằng chứng cho giả thuyết đúng, hệ số này lớn, khi nhân với $P(H_0)$, ta được $P(H_0|E)$ lớn. Nhờ đó, trong suy luận Bayes, định lý Bayes đo được mức độ mà bằng chứng mới sẽ làm thay đổi sự tin tưởng vào một giả thuyết.

Ví dụ: Kết quả dương tính sai trong xét nghiệm y học.

Giả sử 1 xét nghiệm cho 1 căn bệnh ra kết quả xác suất:

- Dương tính đúng: $0,99$
- Âm tính đúng: $0,95$

Giả sử chỉ có $0,1\%$ dân số mắc căn bệnh này (dĩ nhiên chúng ta chỉ đang dừng ở mức ước lượng ban đầu chứ chưa thực sự biết rõ về xác suất này), chọn ngẫu nhiên 1 người thì người đó có **xác suất tiền nghiệm** mắc bệnh là $0,001$.

Giả sử:

- A: tình huống người bệnh mắc bệnh
- B: bằng chứng (updated data)

Câu hỏi đặt ra là: Chẳng hạn như có một người đi khám và nhận được kết quả là dương tính, vậy xác suất để người đó thật sự bị bệnh là bao nhiêu?

Câu trả lời cho bài toán nằm ở định lý Bayes. Thật vậy, xác suất người đó thực sự nhiễm bệnh khi biết kết quả dương tính là:

$$\begin{aligned} P(A|B) &= \frac{P(B|A).P(A)}{P(B|A).P(A) + P(B|notA).P(notA)} \\ &= \frac{0,99.0,001}{0,99.0,001 + 0,05.0,999} \approx 0,019 \end{aligned}$$

Tuy có thể thấy rằng xác suất đúng là rất nhỏ nhưng nếu so với xác suất nhiễm bệnh ban đầu $(0,1\%)$ thì kết quả này cao hơn gần 19 lần. Ý nghĩa của kết quả này là, dựa vào kết quả (dữ liệu) mà ta quan sát được (kết quả dương tính), niềm tin của ta vào khả năng nhiễm bệnh đã thay đổi, chẳng hạn như ta nghĩ rằng xác suất nhiễm bệnh đã cao hơn so với ban đầu (và thực sự là cao hơn 19 lần!, dù đó là với con số vẫn rất nhỏ) vì do khả năng đoán trúng dương tính vẫn rất cao. Do đó kết quả này không đơn thuần là vô ích mà còn thay đổi độ tin cậy của ta.

Nếu ta giải thích dưới ngôn ngữ phân loại Bayes, coi hai class ở đây là nhiễm bệnh hoặc không nhiễm bệnh và với kết quả dương hoặc âm tính là tính trạng của dữ liệu, thì khi có một dữ liệu mới được cập nhật (ví dụ như dương tính), thì niềm tin hay sự tin tưởng về khả năng nhiễm bệnh sẽ bị thay đổi, và xác suất tiền nghiệm của chúng ta cũng sẽ bị thay đổi.

Cũng xét với ví dụ trên, ta quan sát xem niềm tin của chúng ta đã thay đổi thế nào dựa vào sự quan sát dữ liệu đầu vào. Trước khi có chẩn đoán bệnh, xác suất nhiễm bệnh là 0.001. Nhưng sau khi quan sát kết quả khám nghiệm, ta điều chỉnh xác suất tiên nghiệm từ $P(\text{bệnh}) = 0,001$ lên $P(\text{bệnh}|\text{dương tính}) = 0.019$. Như vậy, qua mỗi lần cập nhật dữ liệu, ta sẽ cập nhật **xác suất tiền nghiệm** bằng chính **xác suất hậu nghiệm** ngay sau khi ta cập nhật dữ liệu gần nhất.

Xét bài toán phân loại Bayes: với n điểm dữ liệu $x_1, ..., x_n$ của d thuộc tính đã được phân loại vào m iả sử class $c_1, ..., c_m$. Khi đó, ta quan sát được một dữ liệu x. Giả sử khi đấy, ta có các xác suất tiền nghiệm của các class là $p(c_1), ..., p(c_m)$. Khi đó với $1 \leq k \leq m$, ta có phân phối hậu nghiệm là $p(c_k|x)$, tức là đây là xác suất biểu thị niềm tin của ta về class c_k sau khi quan sát x. Do đó sau khi quan sát x, ta có thể điều chỉnh xác suất tiền nghiệm như sau: với $1 \leq k \leq m$

$$p(c_k) := p(c_k|x)$$

Ở đây, biểu thức có nghĩa là ta gán xác suất hậu nghiệm $p(c_k|x)$ cho xác suất tiền nghiệm $p(c_k)$.

Lưu ý Suy luận Bayes được dùng để tính xác suất quyết định trong tình huống không chắc chắn. Bên cạnh xác suất, ta nên tính một hàm mất mát (loss function) nhằm phản ánh hậu quả của việc phạm sai lầm.

Ta có thể xét một ví dụ của việc cập nhật dữ liệu này trên trường hợp biến liên tục như sau: Dựa như trên, ta đã biết rằng

$$p(\theta|x_1, ..., x_d) = c.f(\theta; \mu_0, \sigma_0).\prod_{i=1}^{d}.f(x_i; \theta, \sigma) = D(\theta)$$

Với θ là một biến liên tục ứng với một class liên tục nào đó, $(x_1, ...x_d)$ là một dữ liệu đang được huấn luyện.

Bằng một số biến đổi, ta có thể chỉ ra được rằng:

$$P(\theta|x_1, ..., x_d) \sim f(\theta; \mu_1, \sigma_1)$$

Với

$$\sigma_1^2 = \left(\frac{1}{\sigma_0^2} + \frac{d}{\sigma^2} \right)^{-1}$$

và

$$\mu_1 = \sigma_1^2 \left(\frac{\mu_0}{\sigma_0^2} + \frac{\sum_{i=1}^{d} x_i}{\sigma^2} \right)$$

Điều này có ý nghĩa là, sau khi cập nhật xác suất tiền nghiệm của mỗi biến liên tục θ bằng xác suất hậu nghiệm $P(\theta|x_1, ..., x_d)$ dựa trên dữ liệu mới $(x_1, ..., x_d)$, thì xác suất tiền nghiệm vẫn là một phân phối chuẩn, với giá trị trung bình và phương sai được cập nhật như trên. Điều này là hợp lý và cho phép ta tiếp tục thực hiện việc cập nhật dữ liệu. Nếu giả sử class của ta là một đoạn liên tục $[a, b]$ thì ta cập nhật từng xác suất tiền nghiệm theo data mới, sau đó cập nhật xác suất tiền nghiệm cho class như sau:

$$P'([a, b]) = \int_a^b P'(\theta) d\theta$$

Với $P'(\theta)$ là xác suất tiền nghiệm sau khi cập nhật của biến θ.

4 Xử lí những dữ liệu không cân bằng (imbalanced data)

Tưởng tượng 1 giáo viên được yêu cầu phải dự đoán kết quả tổng kết của học sinh trong lớp cho năm học mới: lên lớp hoặc lưu ban. Để thực hiện việc này, giáo viên phải tập hợp dữ liệu từ năm học trước (điểm số, điểm danh, kết quả tổng kết).
Giả sử giáo viên rất giỏi nên gần như không có học sinh cũ nào bị lưu ban. Giả sử 99% học sinh của người đó được lên lớp.
Cách nhanh và trực tiếp nhất là dự đoán rằng 100% học sinh của giáo viên đó sẽ lên lớp. Độ chính xác trong trường hợp này là 99% khi so sánh với những năm trước. Tuy nhiên, mô hình này không còn đúng khi có bằng chứng mới từ học sinh mới sai với dự đoán của người này.
Giáo viên có thể dự đoán rằng cả lớp sẽ được lên lớp dựa vào dữ liệu từ những năm trước và vẫn nhận được độ chính xác cao. Tuy nhiên, dự đoán như thế không có giá trị và thể hiện chưa chính xác bản chất dữ liệu.

Trong áp dụng naive Bayes vào việc tính xác suất và phân loai dữ liệu, mất cân bằng tập dữ liệu có thể cho những kết luận có độ chính xác cao nhưng không phản ánh đúng hoàn toàn bản chất của tập dữ liệu. Do đó, vấn đề ở đây là giải quyết dữ liệu mất cân bằng như ví dụ trên. Trong bài báo cáo này sẽ đề cập đến 2 phương pháp chính được sử dụng nhiều nhất.

- **Random-over sampling:** là phương pháp non-heuristic. Mục đích chính là cân bằng lại dữ liệu bằng cách tự tạo ra các dữ liệu mới thuộc lớp dữ liệu thiểu số qua sao chép từ những dữ liệu sẵn có.

- **Random-under sampling:** là phương pháp non-heuristic. Mục đích chính là cân bằng dữ liệu bằng cách ngẫu nhiên loại bớt những dữ liệu từ lớp dữ liệu đa số.

Nhiều nhận định cho rằng, Random-over sampling tăng khả năng xảy ra overfit vì sao chép y hệt các dữ liệu thuộc lớp dữ liệu thiểu số. Bằng cách này, ta có thể phân loại đúng nhưng lại bao hàm dữ liệu sao chép. Đối với Random-under sampling, bất lợi lớn nhất là nguy cơ mất đi những dữ liệu quan trọng. Do đó, 2 phương pháp nêu trên được dùng kết hợp với một số phương pháp khác để tối ưu hoá việc giải quyết vấn đề này.

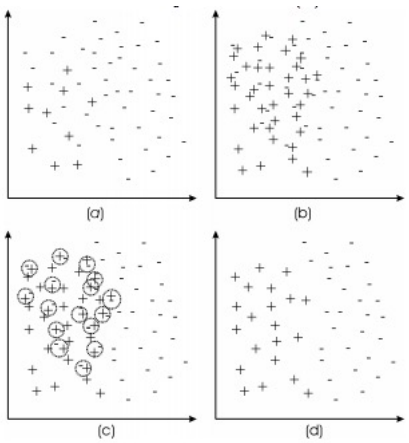
- **Tomek link** Giả sử như 2 dữ liệu E_i và E_j thuộc 2 lớp khác nhau và $d(E_i, E_j)$ là khoảng cách giữa E_i và E_j. Cặp $A(E_i, E_j)$ được gọi là Tomek link nếu không tồn tại E_l sao cho $d(E_i, E_l) < d(E_i, E_j)$ hay $d(E_j, E_l) < d(E_i, E_j)$. Nếu có 2 dữ liệu tạo thành Tomek link thì 1 trong 2 dữ liệu này sẽ nhiễu hoặc cả 2 đều nằm ở đường biên. Tomek links có thể sử dụng như 1 phương pháp:

  - under-sampling: chỉ loại bỏ dữ liệu thuộc lớp dữ liệu đa số.
  - làm sạch dữ liệu: loại bỏ dữ liệu từ cả lớp dữ liệu đa số và thiểu số.

- **One-sided selection (OSS)** 1 phương pháp under-sampling suy ra từ ứng dụng của Tomek links trong Convolutional Neural Network (CNN). Tomek links được dùng như 1 phương pháp under-sampling và loại bỏ những dữ liệu nhiễu và dữ liệu nằm trên đường biên từ lớp dữ liệu đa số. Dữ liệu nằm trên đường biên gây nguy hiểm vì chỉ cần vài dữ liệu nhiễu cũng có thể khiến dữ liệu bị phân loại nhầm ở đường biên quyết định. Mục đích của CNN là bỏ những dữ liệu từ lớp đa số mà có khoảng cách xa đến đường biên quyết định. Những dữ liệu còn lại (dữ liệu an toàn từ lớp đa số và toàn bộ dữ liệu lớp thiểu số) được dùng để tiếp tục quá trình khảo sát và tính toán.

- **Smote Synthetic Minority Over-sampling Technique (Smote)** Đây là 1 phương pháp over-sampling. Ý tưởng chính của phương pháp này là tạo ra dữ liệu mới xen vào những

dữ liệu có sẵn nằm cạnh nhau ở lớp thiểu số. Do đó, tránh được vấn đề dữ liệu bị overfit và làm cho đường biên quyết định của lớp dữ liệu thiểu số mở rộng ra tiến gần hơn đến không gian lớp dữ liệu đa số.Tuy nhiên, một vài dữ liệu lớp đa số có thể lấn sang không gian của lớp dữ liệu thiểu số. Trường hợp ngược lại vẫn có thể diễn ra vì việc xen các dữ liệu vào lớp thiểu số có thể mở rộng cụm các lớp thiểu số, đưa các dữ liệu thiểu số nhân tạo vào sâu trong không gian của lớp đa số. Quyết định phân loại dữ liệu trong tình huống như trên có thể dẫn tới vấn đề overfitting. Do đó, thay vì chỉ loại bỏ những dữ liệu từ lớp đa số tạo nên Tomek links, ta sử dụng kết hợp giữa SMOTE và Tomek links để dữ liệu từ cả 2 lớp thiểu số và đa số được loại bỏ ngẫu nhiên. Ứng dụng của phương pháp kết hợp này được miêu tả trong hình sau.



Đầu tiên, dữ liệu gốc (a) được over-sampled bằng phương pháp SMOTE (b). Sau đó, Tomek links được xác định (c) và được loại bỏ, tạo ra tập dữ liệu cân bằng với các cụm lớp dữ liệu được phân định rõ ràng (d). Phương pháp SMOTE + Tomek links được sử dụng đầu tiên để cải thiện sự phân loại dữ liệu cho vấn đề chú thích, gọi tên proteins trong Tin sinh học.

- **Neighborhood Cleaning Rule (NCL)** Sử dụng Wilson's Edited Nearest Neighbor Rule (ENN) để loại bỏ bớt dữ liệu thuộc lớp dữ liệu đa số. ENN loại bỏ những dữ liệu mà lớp của nó được dán nhãn khác với ít nhất 2 trong 3 dữ liệu liền kề nó. NCL được sử dụng cùng với ENN để tăng số lượng dữ liệu được làm sạch. Đối với vấn đề dữ liệu không cân bằng giữa 2 lớp, thuật toán có thể được diễn tả như sau: với mỗi dữ liệu E_i trong tập dữ liệu huấn luyện, 3 dữ liệu gần nhất của nó được xem xét. Nếu E_i thuộc lớp dữ liệu đa số và 3 dữ liệu liền kề với nó được phân loại khác thì dữ liệu E_i bị loại bỏ. Nếu E_i thuộc lớp dữ liệu thiểu số và 3 dữ liệu liền kề nhất của nó được phân loại khác thì dữ liệu thuộc lớp đa số gần nhất sẽ được loại bỏ.

Trong các phương pháp nêu trên, phương pháp over-sampling nói chung, SMOTE + Tomek và SMOTE + ENN nói riêng, cho kết quả rất tốt khi áp dụng cho những tập dữ liệu có lớp tích cực là lớp thiểu số. Hơn nữa, Random-over sampling, thường được xem là phương pháp không ổn định, có thể đưa ra những kết quả có tính cạnh tranh cao với những kết quả có được bằng cách sử dụng các phương pháp phức tạp hơn. SMOTE + Tomek hay SMOTE + ENN được đề xuất áp dụng với những tập dữ liệu có số lượng dữ liệu tích cực nhỏ. Với những tập dữ liệu có số lượng dữ liệu tích cực lớn, phương pháp Random over-sampling được thực hiện dễ dàng hơn trên máy tính so với những phương pháp khác nhưng vẫn cho ra những kết quả tin cậy.

5 Lựa chọn tính trạng độc lập

5.1 Đặt vấn đề

Machine learning hoạt động theo một nguyên tắc cơ bản: Nếu dữ liệu cho vào bị nhiễu, dữ liệu đầu ra sau khi máy học xong cũng sẽ bị nhiễu.
Khi dữ liệu đầu vào là rất lớn. Chúng ta không nhất thiết phải dùng toàn bộ dữ liệu để cho máy học. Chúng ta chỉ cần cung cấp những dữ liệu thật sự quan trọng, để:

1. Quá trình xử lý của máy nhanh hơn.
2. Làm giảm độ phức tạp của mô hình.
3. Tăng tính đúng đắn của mô hình.
4. Giảm thiểu overfitting.

Khi đó, việc tìm ra phương pháp chọn được đúng dữ liệu cần thiết để đưa vào làm việc là rất cần thiết nhằm tăng tính hiệu quả của việc tính toán. Một trong những phương pháp khá đơn giản có thể được sử dụng là **Phương pháp Pearson's Correlation**

5.2 Phương pháp Pearson's Correlation

5.2.1 Hệ số tương quan Pearson

Trước khi đi vào nghiên cứu về phương pháp Pearson's Correlation, ta cần tìm hiểu sơ bộ về một trong những hệ số tương quan phổ biến nhất trong lí thuyết xác suất thống kê - hệ số tương quan Pearson (Pearson correlation coefficient), hay còn gọi là Pearson's r, PMMC, hay mối tương quan giữa hai biến số.

Đây là một chỉ số thống kê đo lường mối liên hệ tuyến tính giữa các biến độc lập và biến phụ thuộc, ví dụ như giữa MỨC ĐỘ HÀI LÒNG (y) và TIỀN LƯƠNG (x). Hệ số tương quan có giá trị từ -1 đến 1. Hệ số tương quan có giá trị:

- 0 (hay gần 0) có nghĩa là hai biến số không có liên hệ gì với nhau
- -1 hay 1 có nghĩa là hai biến số có một mối liên hệ tuyệt đối
- âm nghĩa là khi x tăng cao thì y giảm (và ngược lại, khi x giảm thì y tăng)
- dương nghĩa là khi x tăng cao thì y cũng tăng, và khi x tăng cao thì y cũng tăng theo

5.2.2 Ý tưởng

Trong lí thuyết xác suất và thống kê, hệ số tương quan cho biết mức độ tương quan tuyến tính giữa hai biến ngẫu nhiên. Trong đó phổ biến nhất là hệ số tương quan Pearson.

5.2.3 Phương pháp

Cho hai biến ngẫu nhiên X, Y, trước khi áp dụng Naive Bayes để phân loại ta cần kiểm tra về tính độc lập của hai biến ngẫu nhiên trên (bài toán khi có nhiều biến hơn cũng tương tự vì chỉ cần so sánh tính độc lập của từng cặp biến ngẫu nhiên).

Hệ số tương quan $\rho_{X,Y}$ giữa hai biến ngẫu nhiên X, Y với kỳ vọng tương ứng μ_X, μ_Y và độ lệch chuẩn lần lượt là σ_X, σ_Y được tính như sau:

$$\rho_{X,Y} = \frac{\operatorname{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E\left((X-\mu_X)(Y-\mu_Y)\right)}{\sigma_X \sigma_Y}$$

Ta có:

$$\mu_X = E(x)$$
$$\sigma_X{}^2 = E\left[(X - E(X))^2\right]$$

Tương tự với Y.

Mặt khác, ta có:

$$E\left((X-\mu_X)(Y-\mu_Y)\right) = E(XY) - E(X)E(Y)$$

Biến đổi biểu thức ta thu được:

$$\rho_{X,Y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E(X^2) - E^2(X)}.\sqrt{E(Y^2) - E^2(Y)}}$$

Giá trị của hệ số tương quan nằm trong khoảng [-1;1].Do đó ta sẽ có một số lưu ý sau cho kết quả hệ số tương quan trên:

- Nếu $\rho_{X,Y} = 0$ thì hai biến ngẫu nhiên X,Y là độc lập, ta dùng dữ liệu của cả hai biến cho quá trình thực hiện
- Khi giá trị tuyệt đối càng gần về 1 thì mức độ liên quan giữa hai biến càng lớn, chúng ta có thể chọn một trong hai biến làm dữ liệu đầu vào.
- Với các giá trị khác biểu thị các mức độ liên hệ tương ứng của hai biến
- Khi có nhiều dữ liệu đầu vào, ta thực hiện so sánh từng cặp để chọn những dữ liệu phù hợp nhất.

5.3 Giới thiệu phương pháp Principle Component Analysis (PCA)

Một trong những phương pháp phổ biến và hiệu quả nhất để có thể xử lý và tăng tính độc lập giữa các tính trạng đó là phương pháp **Principle Component Analysis (PCA)**. Sử dụng PCA, ta có thể thay đổi và đưa các tính trạng của chúng ta thành các tính trạng khác mang tính kết hợp và độc lập hoàn toàn với nhau. Để có thể xử lý và đưa các tính trạng này về độc lập, ta cần phải tính toán trên ma trận hiệp phương sai (covariance matrix) và tìm một vector tối ưu thỏa mãn một tính chất nào đó. Về cơ bản, ý tưởng của phương pháp PCA khá gần và tổng quát hơn phương pháp Pearson's Correlation.

6 Áp dụng mô hình

Ứng dụng Naive Bayes vào 1 vấn đề thực tế: lừa đảo thẻ tín dụng.

Chú thích:

- Giả sử các features chúng ta xét đều độc lập với nhau và 2 class:
  - Label 1: thẻ lừa đảo
  - Label 2: thẻ không lừa đảo
- Chọn data và train data với số lượng thẻ tín dụng lừa đảo tương đối vì thực tế số lượng thẻ tín dụng lừa đảo là rất bé nên thường gặp imbalanced data, sau khi test xong thì xác suất thẻ tín dụng lừa đảo bằng 0 nhưng thực tế lại không phải vậy.

Code:

- Khai báo thư viện

```
from __future__ import print_function
from sklearn.naive_bayes import GaussianNB
import numpy as np
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
```

- Đọc dữ liệu

```
Location = r'D:\python\creditcard_it.csv'
df = pd.read_csv(Location)
```

- Chọn cột mà sẽ dùng dùng cho dữ liệu và các labels

```
columns = df.columns[0:30]
label = df.columns[30:]
```

- Tách dữ liễu ra thành 2 phần: 80% dùng để train và 20% để test

```
r = df.shape[0]
train_r = int(r *.8)
train = df.iloc[0:train_r]
test = df.iloc[train_r:r]
```

- Vì các features nhận là liên tục nên dùng Gaussian distribution (phân phối chuẩn) để giải quyết bài toán

```
gnb = GaussianNB()
gnb.fit(train[columns] , train[label])
```

- Kết quả dự đoán bộ test

```
test_predict = gnb.predict(test[columns])
print('Predictions using the testing set:\n',test_predict) #in ra kết quả dự đoán c
```

- Mỗi khi dự đoán kết quả sau khi train xong thì mọi kết quả dường như không đúng hoàn toàn được và 1 trong những phương pháp kiểm tra mức độ đúng của data bằng accuracy score, average precision score và recall score

```
from sklearn.metrics import accuracy_score
print(accuracy_score(test[label], test_predict))
from sklearn.metrics import average_precision_score
print(average_precision_score(test[label], test_predict))
from sklearn.metrics import recall_score
print(recall_score(test[label], test_predict))
```

- Kết quả khá ổn (accuracy score = 0.976405322847064, average precision score = 0.0587818696884, recall score = 0.84693877551) nhưng có thể improve càng chính xác càng tốt vì một vài features có điểm chung do data ban đầu nên sẽ thử bỏ vài features (ở đây chúng ta sẽ bỏ cột 1 của data đi và train lại)

```
columns = columns[1:]
gnb = GaussianNB()
train = df.iloc[0:train_r]
test = df.iloc[train_r:r]
gnb.fit(train[columns], train[label])
test_predict = gnb.predict(test[columns])
print(accuracy_score(test[label], test_predict))
print(average_precision_score(test[label], test_predict))
print(recall_score(test[label], test_predict))
```

- Bỏ thử cột tính trạng 1 và thấy được score đều tăng.Tuy nhiên cách bỏ cột đầu chưa phải là tối ưu nhất nhưng score chúng ta đạt được cũng khá hiệu quả.Từ đó ta rút ra được kết luận phải chọn những dự kiện phù hợp trong điểm dữ liệu để chúng đủ phân biệt với các dữ liệu khác để có kết quả tối ưu nhất.

Kết quả ngoài đời: người ta nhận ra 89% những vụ lừa đảo thẻ tín dụng .

7 Kết luận đánh giá

Mô hình phân loại naive Bayes (NBC) tỏ ra khá hiệu quả trong việc phân loại các dữ liệu phức tạp, với nhiều kiểu dữ liệu khác nhau, vừa có thể rời rạc hoặc liên tục. Hơn nữa, kết quả chính xác tỏ ra cũng rất cao trong cả những trường hợp các tính trạng không hoàn toàn hoặc không quá phân biệt. Điều này đồng nghĩa với việc rằng đôi khi không cần phải xử lý các tính trạng trước khi chạy và huấn luyện dữ liệu mà vẫn cho những kết quả có độ chính xác lớn, tránh trường hợp phải xử lý các tính trạng bằng các phương pháp phức tạp và khó xử lý (nếu có thể tăng tính độc lập thì độ chính xác càng tốt).

Hơn nữa, mô hình Naive Bayes có khả năng dự đoán được sự thay đổi của một hệ phân phối tiền nghiệm sau khi cho một hệ dữ liệu đã được huấn luyện. Điều này có ý nghĩa và ứng dụng rất lớn vào thực tiễn, cũng như thể hiện được mối tương quan chặt chẽ giữa các dữ liệu khác nhau. Dựa vào điều này, ta có thể điều chỉnh, sắp xếp dữ liệu phù hợp để mong muốn ra một kết quả khả thi và chính xác hơn.

Tuy nhiên, phương pháp này đôi khi vẫn tỏ ra có một vài hạn chế nhất định. Chẳng hạn, vì Naive Bayes rất dễ bị ảnh hưởng bởi sự chênh lệch dữ liệu, nên ta rất khó có thể xử lý dữ liệu nếu xảy ra trường hợp imbalanced data, tức là các dữ liệu có sự chênh lệch rất lớn, nên cần phải xử lý imbalanced data trước khi thật sự huấn luyện và xử lý dữ liệu. Với số lượng dữ liệu, số tính trạng lớn hay nhỏ hơn mà phương pháp này không thực sự tối ưu bằng những phương pháp khác. Và dĩ nhiên do bản chất của Naive Bayes, giữa hai tính trạng có sự độc lập hoàn toàn hoặc rất lớn, nên khi xử lý dữ liệu mức độ sai số cũng như những xác suất của kết quả không chính xác cũng cần phải điều chỉnh cho thích hợp. Ta có thể điều chỉnh kết quả của dữ liệu khi chỉnh sửa imbalanced data tùy theo mức độ mong muốn của ta đối với kết quả của dữ liệu.

Tổng quan lại, mô hình phân loại naive Bayes là một mô hình rất tốt trong các mô hình phân loại xác suất, và có nhiều ứng dụng tốt cho Machine Learning. Tuy nhiên, một số tính chất của thuật toán vẫn cần phải giải quyết, khắc phục và ta có thể mở rộng thành các mô hình khác tổng quát hơn để đạt được những kết quả tốt hơn.

8 Hướng phát triển trong tương lai

Mô hình phân loại naive Bayes (Naive Bayes Classifier - NBC) tuy rất hiệu quả trong việc phân loại văn bản với các dữ liệu phức tạp, với nhiều kiểu dữ liệu khác nhau, vừa rời rạc vừa liên tục, thì chắc chắn không thể thiếu những bất lợi làm giảm độ chính xác, cũng như mức độ tin cậy vào kết quả cuối cùng. Do đó, công việc lớn nhất của những nhà Toán học ứng dụng khi sử dụng naive Bayes trong phân loại dữ liệu là giải quyết những nhược điểm mà phương pháp này mắc phải.

Vấn đề đầu tiên là cần phải tối ưu hóa việc cập nhật xác suất tiền nghiệm (prior probability) mỗi khi có những bằng chứng hay quan sát mới được thêm vào tập dữ liệu. Chính những dữ liệu và bằng chứng mới được thêm vào này có thể thay đổi hoàn toàn kết quả phân loại cũ, biến những sự việc hay sự kiện có độ chính xác cao thành thấp và ngược lại. Nói chung, việc giảm thiểu đến tối đa hàm mất mát (loss function) sau mỗi lần cập nhật dữ liệu sẽ giúp tăng độ chính xác cũng như độ tin cậy về giả thuyết ban đầu.

Vấn đề thứ hai cũng rất quan trọng đó là xử lí dữ liệu mất cân bằng (imbalanced data). Trong các bài toán phân loại dữ liệu có sử dụng naive Bayes thì khó tránh khỏi việc tồn tại những lớp dữ liệu có số lượng áp đảo so với những lớp dữ liệu còn lại. Điều này dẫn đến những kết luận tuy có độ chính xác cao nhưng lại không bộc lộ rõ bản chất của dữ liệu, làm giảm giá trị của tập dữ liệu và không hoàn thành được mục đích chính của phương pháp NBC là phân loại dữ liệu. Vấn đề này sẽ làm giảm hiệu quả trong các công việc như phân loại thư rác (spam), phát hiện lừa đảo (fraud detection), hay chẩn đoán bệnh ban đầu (automatic medical diagnosis).

Vấn đề thứ ba là lựa chọn tính trạng độc lập. Khi làm việc với tập dữ liệu lớn, việc lấy toàn bộ tất cả các số liệu gần như là không thể vì sẽ rất tốn thời gian, công sức, cũng như tài nguyên máy. Do đó, chúng ta cần phải cải thiện hơn nữa những phương pháp lựa chọn tính trạng độc lập nhằm giảm thiểu thời gian, công sức trong quá trình làm việc, cũng như rút ngắn và tăng độ chính xác trong quá trình làm việc.

Ba vấn đề nêu trên có thể được xem như những vấn đề quan trọng nhất khi áp dụng naive Bayes vào tính toán trên bất kì tập dữ liệu nào. Khi những vấn đề trên được cải thiện, phương pháp NBC có thể phát huy được những ưu điểm sẵn có của mình để đem lại những kết quả tối ưu và đáng tin cậy nhất, góp phần vào sự phát triển của nhiều ngành khoa học khác nói chung và Machine Learning nói riêng.

Ngoài ra, việc nghiên cứu ra những phương pháp phân loại khác có tính tối ưu cao hơn naive Bayes Classifier cũng là vấn đề rất đáng được lưu tâm. Việc nghiên cứu ra những phương pháp mới có thể sẽ dựa trên nền tảng của phương pháp NBC nhưng giản lược bớt các nhược điểm của nó và bổ sung thêm những tính ưu việt để đạt được hiệu quả làm việc lớn nhất, đem lại những kết quả có độ chính xác và độ tin cậy cao nhất.

Tài liệu

[1] *Bayes Inference for the Normal Distribution*.

[2] Maria Carolina Monard Gustavo E. A. P. A. Batista, Ronaldo C. Prati. *A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data*. Instituto de Ciências Matemáticas e de Computação, 2004.

[3] Kaggle. *Fraud Detection with Naive Bayes Classifier*. Credit Card Fraud Detection, 2018.

[4] Rafael Pierre. *Detecting Financial Fraud Using Machine Learning: Winning the War Against Imbalanced Data*. 2018.

[5] Trần Hoàng Bảo Linh Vũ Lê Thế Anh, Phạm Nguyễn Mạnh. *Xác suất và phân phối*. Projects in Mathematics and Applications, 2018.

[6] Wikipedia. *Bayes inference*.

[7] Wikipedia. *Correlation and dependence*.

[8] Wikipedia. *Naive Bayes classifier*.

[9] Wikipedia. *Pearson correlation coefficient*.

[10] Wikipedia. *Suy luận Bayes*.

[11] Nhóm MBA Đại học Bách Khoa TPHCM. *Hệ số tương quan Pearson, cách thao tác phân tích tương quan trong SPSS*.