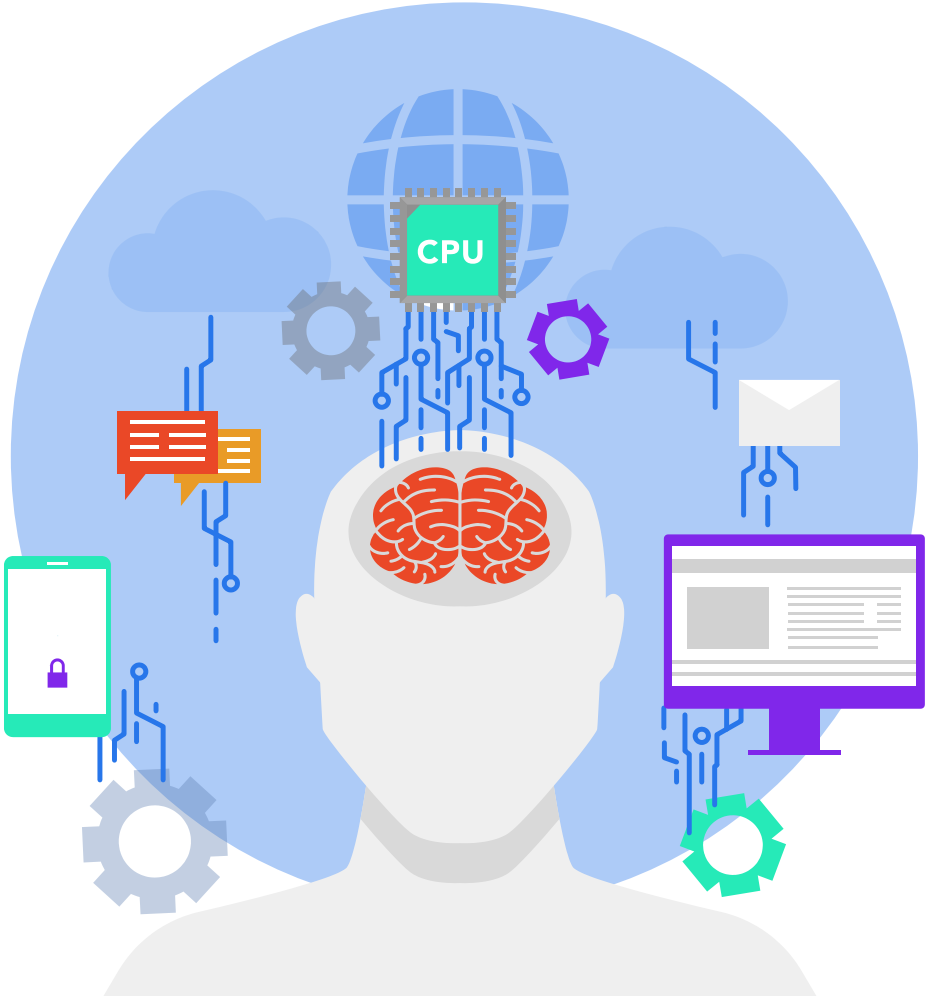
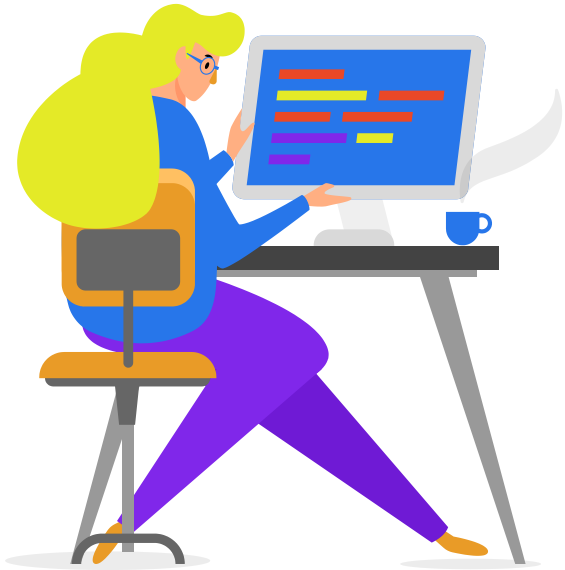


Chương 3: Hệ phân lớp Naive Bayes

3.1 Định lý Bayes



01

Xác suất có điều kiện

Hiểu khái niệm xác suất có điều kiện trước khi vào định lý Bayes

02

Định lý Bayes

Công thức định lý Bayes

03

Diễn giải bằng ngôn ngữ phân lớp

Sử dụng ngôn ngữ phân lớp để thể hiện định lý Bayes

04

Vai trò trong học máy

Mục đích sử dụng của định lý trong học máy

3.1.1 Giới thiệu xác suất có điều kiện

Trước khi đi vào định lý Bayes, cần hiểu khái niệm xác suất có điều kiện:

- Gọi A và B là hai biến cố
- Xác suất xảy ra của A khi biết B đã xảy ra gọi là xác suất có điều kiện, ký hiệu là:

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \ (với\ P(B) > 0)$$

Tương tự

3.1.2 Định lý Bayes

Dựa vào định nghĩa xác suất có điều kiện, suy ra:

$$P(A|B) = \frac{P(B|A).P(A)}{P(B)}$$

Định lý Bayes là công cụ quan trọng trong thống kê học máy để cập nhật niềm tin (xác suất) về một giả thuyết sau khi có thông tin quan sát mới

3.1.3 Diễn giải bằng ngôn ngữ phân lớp

Trong bài toán phân lớp:

- $A \equiv Y = c$: giả thuyết dữ liệu thuộc lớp c
- $B \equiv x$: dữ liệu quan sát đầu vào

Áp dụng định lý Bayes:

$$P(Y = c|x) = \frac{P(x|Y = c).P(Y = c)}{P(x)}$$

Trong đó:

- $P(Y = c)$: xác suất tiên nghiệm (prior) của lớp ccc
- $P(x|Y = c)$: xác suất có điều kiện (likelihood) – mô hình hóa sự phân bố dữ liệu trong lớp
- $P(x)$: xác suất biên (evidence) – không phụ thuộc vào lớp

3.1.4 Vai trò trong học máy

- Định lý Bayes là nền tảng cho các thuật toán học xác suất như Naive Bayes, Bayesian Networks, Bayesian Inference,...
- Trong Naive Bayes, ta sử dụng công thức này để tính xác suất hậu nghiệm $P(Y|x)$ và chọn lớp có giá trị cao nhất.

- Ví dụ:

Giả sử:

•Xác suất mưa trong ngày là 30% $\rightarrow P(Mưa) = 0.3$

•Khi mưa thì 80% người mang ô $\rightarrow P(Ô|Mưa) = 0.8$

•Khi không mưa thì 20% người vẫn mang ô $\rightarrow P(Ô|Không\ mưa) = 0/2$

Câu hỏi: **Nếu bạn thấy ai đó mang ô, thì xác suất hôm nay mưa là bao nhiêu?**

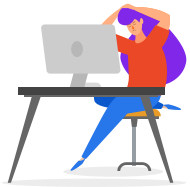
Áp dụng định lý Bayes:

$$P(Mưa|Ô) = \frac{P(Ô|Mưa).P(Mưa)}{P(Ô)}$$

Trong đó:

$$P(Ô) = P(Ô|Mưa).P(Mưa) + P(Ô|Không\ mưa).P(Không\ mưa) = 0.8 \cdot 0.3 + 0.2 \cdot 0.7 = 0.38$$

3.2 Hệ phân lớp Naive Bayes Classifier (NBC)



- Xét bài toán classification với C classes 1,2,…,C. Giả sử có một điểm dữ liệu x∈Rd. Hãy tính xác suất để điểm dữ liệu này rơi vào class c. Nói cách khác, hãy tính:

$$p(y = c|x) \quad (1)$$

Hoặc viết gọn lại thành $p(c|x)$

- Tức tính xác suất để đầu ra là class c biết rằng đầu vào là vector x.

- Biểu thức này, nếu tính được, sẽ giúp chúng ta xác định được xác suất để điểm dữ liệu rơi vào mỗi class. Từ đó có thể giúp xác định class của điểm dữ liệu đó bằng cách chọn ra class có xác suất cao nhất:

$$c = arg \max_{c \in \{1, \dots, C\}} p(c|x) \quad (2)$$

- Biểu thức (2) thường khó được tính trực tiếp.

3.2 Hệ phân lớp Naive Bayes Classifier (NBC)

- Thay vào đó, quy tắc Bayes thường được sử dụng:

$$c = \arg\max p(c|x) \quad (3) = \arg\max \frac{p(x|c)p(c)}{p(x)} \quad (4) = \arg\max p(x|c)p(c) \quad (5)$$

- Từ (3) → (4): áp dụng **quy tắc Bayes**.
- Từ (4) → (5): vì **p(x)p(x)p(x)** không phụ thuộc vào lớp ccc.
- Trong (5):
- p(c)p(c)p(c): xác suất một điểm thuộc class ccc, ước lượng bằng:
  - **MLE**: tỉ lệ số điểm thuộc class đó trong tập huấn luyện.
  - **MAP**: ước lượng có thêm prior (ít dùng hơn).
- p(x|c)p(x|c)p(x|c): phân phối của dữ liệu trong class ccc, thường **khó tính** do xxx nhiều chiều.
- Để đơn giản hóa, thường giả sử các **thành phần của xxx độc lập nhau** nếu biết ccc → gọi là **giả định Naive Bayes**.

3.2 Hệ phân lớp Naive Bayes Classifier (NBC)

- Tức là:

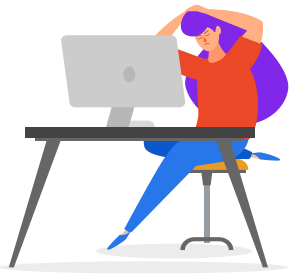
$$p(x|c) = p(x1, x2, \dots, xd|c) = \prod_{i=1}^{d} p(x_i|c) \quad (6)$$

- Giả thiết các chiều của dữ liệu độc lập với nhau, nếu biết c, là quá chặt và ít khi tìm được dữ liệu mà các thành phần hoàn toàn độc lập với nhau. Tuy nhiên, giả thiết này lại mang lại những kết quả tốt bất ngờ. Giả thiết về sự độc lập của các chiều dữ liệu này được gọi là *Naive Bayes*. Cách xác định class của dữ liệu dựa trên giả thiết này có tên là *Naive Bayes Classifier (NBC)*.
- NBC, nhờ vào tính đơn giản, có tốc độ training và test rất nhanh. Việc này giúp nó mang lại hiệu quả cao trong các bài toán large-scale.

3.2 Hệ phân lớp Naive Bayes Classifier (NBC)

Bước training:

- Xác định p(c)p(c)p(c) và p(xi|c),i=1,...,dp(x_i | c), i = 1,...,dp(xi|c),i=1,...,d từ dữ liệu huấn luyện.
- Dùng **MLE** hoặc **MAP** để ước lượng.



Bước test:

- Với điểm mới x, phân lớp theo:

$$c = \arg\max_{c \in \{1,\dots,C\}} p(c) \prod_{i=1}^{d} p(x_i|c) \quad (7)$$

- Thường dùng **log** để tính → không ảnh hưởng kết quả vì log là hàm đồng biến.

Mặc dù giả định độc lập trong Naive Bayes rất mạnh, thuật toán vẫn hiệu quả thực tế, đặc biệt:

- **Phân loại văn bản**: lọc spam, phân loại email, phân loại tin nhắn,...

Ba biến thể phổ biến của Naive Bayes:

- **Gaussian NB** – cho dữ liệu liên tục.
- **Multinomial NB** – cho dữ liệu rời rạc, đếm (bag-of-words).
- **Bernoulli NB** – cho dữ liệu nhị phân (có/không).

3.3 Các phân phối xác suất

3 loại phân phối xác suất gồm:

3.3.1 Gaussian Naive Bayes

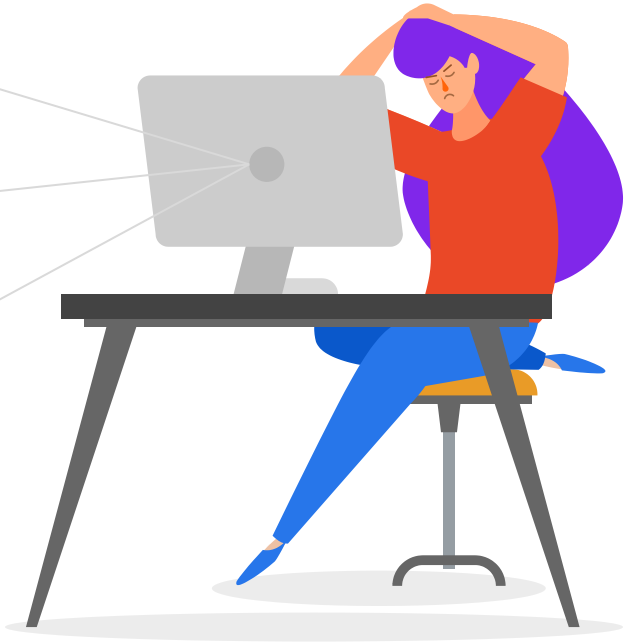
01

3.3.2 Multinomial Naive Bayes

02

3.3.3 Bernoulli Naive Bayes

03



3.3.1 Gaussian Naive Bayes

Mô hình này được sử dụng chủ yếu trong loại dữ liệu mà các thành phần là các biến liên tục.

Với mỗi chiều dữ liệu i và một class c, x_i tuân theo một phân phối chuẩn có kỳ vọng μ_{ci} và phương sai σ_{ci}^2:

$$p(x_i|c) = p\left(x_i|\mu_{ci}, \sigma_{ci}^2\right) = \frac{1}{\sqrt{2\pi\sigma_{ci}^2}} \exp\left(-\frac{(x_i - \mu_{ci})^2}{2\sigma_{ci}^2}\right) \quad (8)$$

Trong đó, bộ tham số θ={μci,σ2ci} $\theta = \{\mu_{ci}, \sigma_{ci}^2\}$ được xác định bằng Maximum Likelihood:

$$\left(\mu_{ci}, \sigma_{ci}^2\right) = \arg\max_{\mu_{ci}, \sigma_{ci}^2} \prod_{n=1}^{N} p\left(x_i^{(n)} \middle| \mu_{ci}, \sigma_{ci}^2\right) \quad (9)$$

Đây là cách tính của thư viện sklearn. Chúng ta cũng có thể đánh giá các tham số bằng MAP nếu biết trước priors của μ_{ci} và σ_{ci}^2

3.3.2 Multinomial Naive Bayes

- Mô hình này chủ yếu được sử dụng trong phân loại văn bản mà feature vectors được tính bằng Bags of Words. Lúc này, mỗi văn bản được biểu diễn bởi một vector có độ dài ***d*** chính là số từ trong từ điển. Giá trị của thành phần thứ ***i*** trong mỗi vector chính là số lần từ thứ ***i*** xuất hiện trong văn bản đó.
- Khi đó, $p(x_i|c)$ tỉ lệ với tần suất từ thứ ***i*** (hay feature thứ ***i*** cho trường hợp tổng quát) xuất hiện trong các văn bản của class c. Giá trị này có thể được tính bằng cách:

$$\lambda_{ci} = p(x_i|c) = \frac{N_{ci}}{N_c} \qquad (10)$$

Trong đó:

•N_{ci} là tổng số lần từ thứ i xuất hiện trong các văn bản của class c, nó được tính là tổng của tất cả các thành phần thứ i của các feature vectors ứng với class c.

3.3.2 Multinomial Naive Bayes

- Cách tính này có một hạn chế là nếu có một từ mới chưa bao giờ xuất hiện trong class c thì biểu thức (10) sẽ bằng 0, điều này dẫn đến vế phải của (7) bằng 0 bất kể các giá trị còn lại có lớn thế nào. Việc này sẽ dẫn đến kết quả không chính xác (xem thêm ví dụ ở mục sau).
- Để giải quyết việc này, một kỹ thuật được gọi là *Laplace smoothing* được áp dụng:

$$\hat{\lambda}_{ci} = \frac{N_{ci} + \alpha}{N_c + d\alpha} \qquad (11)$$

- Với α là một số dương, thường bằng 1, để tránh trường hợp tử số bằng 0. Mẫu số được cộng với dα để đảm bảo tổng xác suất $\sum_{i=1}^{d} \hat{\lambda}_{ci} = 1$.
- Như vậy, mỗi class c sẽ được mô tả bởi bộ các số dương có tổng bằng 1: ^λc={^λc1,…,^λcd} $\hat{\lambda}_c = \{\hat{\lambda}_{c1}, \ldots, \hat{\lambda}_{cd}\}$.

3.3.3 Bernoulli Naive Bayes

Mô hình này được áp dụng cho các loại dữ liệu mà mỗi thành phần là một giá trị binary - bẳng 0 hoặc 1. Ví dụ: cũng với loại văn bản nhưng thay vì đếm tổng số lần xuất hiện của 1 từ trong văn bản, ta chỉ cần quan tâm từ đó có xuất hiện hay không.

Khi đó, $p(x_i|c)$ được tính bằng:

$$p(x_i|c) = p(i|c)^{x_i}(1 - p(i|c)^{1-x_i}$$

Với $p(i|c)$ có thể được hiểu là xác suất từ thứ i xuất hiện trong các văn bản của class c.

3.4 Ví dụ

3.4.1 Bắc hay Nam

Giả sử trong tập training có các văn bản d1, d2, d3, d4 như trong bảng dưới đây. Mỗi văn bản này thuộc vào 1 trong 2 classes: B (*Bắc*) hoặc N (*Nam*). Hãy xác định class của văn bản d5

	Document	Content	Class
Training	d1	hanoi pho chaolong hanoi	B
	d2	hanoi buncha pho omai	B
	d3	pho banhgio omai	B
	d4	saigon hutiu banhbo pho	N
Test	d5	hanoi hanoi buncha hutiu	?

Ta có thể dự đoán rằng d5 thuộc class *Bắc*.

3.4 Ví dụ

3.4.1 Bắc hay Nam

- Nhận thấy rằng ở đây có 2 class B và N, ta cần đi tính p(B) và p(N). Dựa trên tần số xuất hiện của mỗi class trong tập training. Ta sẽ có:

3.4 Ví dụ

3.4.1 Bắc hay Nam

Hình dưới đây minh hoạ quá trình Training và Test cho bài toán này khi sử dụng Multinomial Naive Bayes, trong đó có sử dụng Laplace smoothing với α=1.

TRAINING

class = B

	hanoi	pho	chaolong	buncha	omai	banhgio	saigon	hutiu	banhbo	
d1: $\mathbf{x}_1$	2	1	1	0	0	0	0	0	0	
d2: $\mathbf{x}_2$	1	1	0	1	1	0	0	0	0	
d3: $\mathbf{x}_3$	0	1	0	0	1	1	0	0	0	$d = \|V\| = 9$
Total	3	3	1	1	2	1	0	0	0	$\Rightarrow N_{\mathrm{N}} = 11$
$\Rightarrow \hat{\lambda}_{\mathrm{B}}$	4/20	4/20	2/20	2/20	3/20	2/20	1/20	1/20	1/20	$(20 = N_{\mathrm{B}} + \|V\|)$

class = N

d4: $\mathbf{x}_4$	0	1	0	0	0	0	1	1	1	$\Rightarrow N_{\mathrm{N}} = 4$
$\Rightarrow \hat{\lambda}_{\mathrm{N}}$	1/13	2/13	1/13	1/13	1/13	1/13	2/13	2/13	2/13	$(13 = N_{\mathrm{N}} + \|V\|)$

TEST

d5: $\mathbf{x}_5 = [2, 0, 0, 1, 0, 0, 0, 1, 0]$

$$p(\mathrm{B}|\mathrm{d5}) \propto p(\mathrm{B}) \prod_{i=1}^{d} p(x_i|\mathrm{B})$$

$$= \frac{3}{4} \left(\frac{4}{20}\right)^2 \frac{2}{20} \frac{1}{20} \approx 1.5 \times 10^{-4}$$

$$p(\mathrm{N}|\mathrm{d5}) \propto p(\mathrm{N}) \prod_{i=1}^{d} p(x_i|\mathrm{N})$$

$$= \frac{1}{4} \left(\frac{1}{13}\right)^2 \frac{1}{13} \frac{2}{13} \approx 1.75 \times 10^{-5}$$

$\Rightarrow p(\mathbf{x}_5|\mathrm{B}) > p(\mathbf{x}_5|\mathrm{N}) \Rightarrow \mathrm{d5} \in \mathrm{class(B)}$

3.4 Ví dụ

3.4.1 Bắc hay Nam

$$p(B|d5) = \frac{}{1{,}5.10^{-4} + 1{,}75.10^{-5}} \approx 0.8955, \qquad p(N|d5) = 1 - p(B|d5) \approx 0.1045$$

Bạn đọc có thể tự tính với ví dụ khác: **d6 = pho hutiu banhbo**. Nếu bạn và tôi tính ra kết quả

3.4 Ví dụ

3.4.2 Bắc hay Nam với sklearn

Để kiểm tra lại các phép tính toán phía trên, chúng ta cùng giải quyết bài toán này với sklearn.

```
from __future__ import print_function
from sklearn.naive_bayes import MultinomialNB
import numpy as np

# train data
d1 = [2, 1, 1, 0, 0, 0, 0, 0, 0]
d2 = [1, 1, 0, 1, 1, 0, 0, 0, 0]
d3 = [0, 1, 0, 0, 1, 1, 0, 0, 0]
d4 = [0, 1, 0, 0, 0, 0, 1, 1, 1]

train_data = np.array([d1, d2, d3, d4])
label = np.array(['B', 'B', 'B', 'N'])

# test data
d5 = np.array([[2, 0, 0, 1, 0, 0, 0, 1, 0]])
d6 = np.array([[0, 1, 0, 0, 0, 0, 0, 1, 1]])

## call MultinomialNB
clf = MultinomialNB()
# training
clf.fit(train_data, label)

# test
print('Predicting class of d5:', str(clf.predict(d5)[0]))
print('Probability of d6 in each class:', clf.predict_proba(d6))
```

Kết quả:

```
Predicting class of d5: B
Probability of d6 in each class: [[
0.16948581  0.83051419]]
```

3.4 Ví dụ

3.4.2 Bắc hay Nam với sklearn

Nếu sử dụng mô hình Bernoulli Naive Bayes, chúng ta cần thay đổi một chút về feature vectors. Lúc này, các giá trị khác không sẽ đều được đưa về 1 vì ta chỉ quan tâm đến việc từ đó có xuất hiện trong văn bản không.

```
from __future__ import print_function
from sklearn.naive_bayes import BernoulliNB
import numpy as np
# train data
d1 = [1, 1, 1, 0, 0, 0, 0, 0, 0]
d2 = [1, 1, 0, 1, 1, 0, 0, 0, 0]
d3 = [0, 1, 0, 0, 1, 1, 0, 0, 0]
d4 = [0, 1, 0, 0, 0, 0, 1, 1, 1]
train_data = np.array([d1, d2, d3, d4])
label = np.array(['B', 'B', 'B', 'N']) # 0 - B, 1 - N
# test data
d5 = np.array([[1, 0, 0, 1, 0, 0, 0, 1, 0]])
d6 = np.array([[0, 1, 0, 0, 0, 0, 0, 1, 1]])
## call MultinomialNB
clf = BernoulliNB()
# training
clf.fit(train_data, label)
# test
print('Predicting class of d5:', str(clf.predict(d5)[0]))
print('Probability of d6 in each class:',
clf.predict_proba(d6))
```

Kết quả:

```
Predicting class of d5: B
Probability of d6 in each class: [[
0.16948581  0.83051419]]
```

3.4 Ví dụ

3.4.3 Bài toán Spam Filtering

Đề 1:

Bạn có tập dữ liệu gồm 10 email đã được gán nhãn (spam hoặc not spam) và số lần xuất hiện của từ khóa:
["free", "win", "money", "urgent", "click"] trong mỗi email.
Yêu cầu:

- a) Tính xác suất tiên nghiệm của mỗi lớp (spam / not spam).
- b) Tính xác suất có điều kiện của mỗi từ với từng lớp (dùng Laplace smoothing).
- c) Với một email mới: "free urgent click", hãy dùng **Multinomial Naive Bayes** để tính xác suất thuộc từng lớp và phân loại nó.

3.4 Ví dụ

3.4.3 Bài toán Spam Filtering

Đề 2:

Tải về tập dữ liệu **SMS Spam Collection Dataset** từ UCI (hoặc Kaggle).

Mỗi dòng là một tin nhắn với nhãn spam hoặc ham.

Yêu cầu:

1. Tiền xử lý văn bản: viết hàm loại bỏ dấu câu, chuyển về chữ thường, bỏ stop words.
2. Vector hóa văn bản (Bag-of-Words hoặc TF-IDF).
3. Áp dụng **Multinomial Naive Bayes** để huấn luyện mô hình.
4. Đánh giá mô hình trên tập test (accuracy, precision, recall, F1-score).
5. Đưa ví dụ một số tin nhắn thực và dự đoán xem chúng có phải spam không.

3.4 Ví dụ

3.4.3 Bài toán Spam Filtering

Đề 3:

So sánh hiệu quả giữa **Gaussian**, **Multinomial** và **Bernoulli Naive Bayes** trên cùng tập dữ liệu spam.

Yêu cầu:

- Dùng cùng một tập dữ liệu tin nhắn spam (có thể cắt nhỏ).
- Triển khai 3 biến thể Naive Bayes và so sánh các mô hình theo các tiêu chí:
    - Accuracy
    - Confusion Matrix
    - Speed (thời gian huấn luyện và test)

3.4 Ví dụ

3.4.3 Bài toán Spam Filtering

Đề 6:

Cho danh sách 10 tin nhắn sau và nhãn tương ứng. Viết một đoạn mã Python ngắn để huấn luyện Naive Bayes (Multinomial) bằng sklearn.naive_bayes.MultinomialNB

```
emails = [
    "Win money now", "Urgent: click to claim your prize", "Meeting at 3pm",
    "Let's have lunch tomorrow", "You won a free ticket", "Don't miss out, win big!",
    "See you at the office", "Click this link for surprise", "Reminder: project deadline",
    "Join us to win rewards"
]
labels = [1,1,0,0,1,1,0,1,0,1]  # 1 = spam, 0 = not spam
```

Yêu cầu:

- Tiền xử lý văn bản
- Vector hóa
- Huấn luyện mô hình
- Dự đoán nhãn cho tin nhắn "Free prize for you"