ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
UIT-HCM

BÁO CÁO ĐỒ ÁN MÔN LẬP TRÌNH PYTHON CHO MÁY HỌC

Naive Bayes Classifer

Giảng viên hướng dẫn: TS. Nguyễn Vinh Tiệp

STT	Họ và tên	MSSV
1	Hoàng Xuân Vũ	19522531
2	Nguyễn Văn Thành	19522243
3	Nguyễn Trọng Thoại	19522298

Lời cám ơn

Sau quá trình học tập và rèn luyện tai trường Đại học Công Nghệ Thông Tin, chúng em đã được trang bị các kiến thức cơ bản, các kỹ năng thực tế để có thể oàn thành đồ án môn học của mình.

Chúng em xin gửi lời cảm ơn chân thành đến thầy Nguyễn Vinh Tiệp đã tận tâm hướng dẫn, truyền đạt những kiến thức cũng như kinh nghiệm cho chúng em trong suốt thời gian học tập môn lập trình Python cho máy học.

Trong quá trình làm đồ án môn học, chúng em không tránh được những sai sót. Mong nhận được sự góp ý cũng như kinh nghiệm quý báu của các các thầy để được hoàn thiện hơn và rút kinh nghiệm cho những môn học sau. Chúng em xin chân thành cảm ơn!

TP. Hồ Chí Minh, tháng 11 năm 2021.

Mục lục

1 Giới thiệu **4**

2 Naive Bayes Classifier **4**
2.1 Định lý Bayes 4
2.2 Naive Bayes Classifier 4

3 Các dạng phân phối **6**
3.1 Gaussian Naive Bayes 6
3.2 Multinomial Naive Bayes 6
3.3 Bernouli Naive Bayes 7

4 Ví dụ **7**
4.1 Dự đoán khách hàng mua máy tính 7
4.2 Dự đoán khách hàng mua máy tính với sklearn 8

5 Siêu tham số của các phân phối trong thư viện scikit-learn **8**
5.1 Gaussian Naive Bayes 8
5.2 Multinomial Naive Bayes 8
5.3 Bernoulli Naive Bayes 9

6 Thực nghiệm **9**
6.1 Bài toán 9
6.2 Dữ liệu 9
6.3 Phương pháp 9
6.3.1 Tiền xử lý dữ liệu 9
6.3.2 Trích xuất đặc trưng 10
6.3.3 Tinh chỉnh siêu tham số của mô hình 10
6.4 Kết quả chạy thí nghiệm các mô hình 11

7 Kết luận **12**
7.1 Ưu điểm của mô hình 12
7.2 Nhược điểm của mô hình 12

1 Giới thiệu

Naive Bayes Classifiers (viết tắt là NBC) là một thuật toán máy học có giám sát (supervised learning) thường được dùng cho bài toán phân lớp (Classification). NBC thuộc họ các thuật toán phân lớp dựa trên xác suất của dữ liệu. Như cái tên gợi ý, NBC hoạt động bằng cách sử dụng định lý Bayes với giả thuyết các đặc trưng của dữ liệu hoàn toàn độc lập. Trên thực tế thì điều này rất hiếm khi xảy ra nên giả thuyết này được xem là *ngây ngô* (naive assumption)

Mặc dù đơn giản và bị xem là ngây ngô nhưng thực nghiệm cho thấy mô hình NBC thực thi với tốc độ nhanh và cho độ chính xác cao trên nhiều bài toán như phân loại văn bản, phân loại bệnh ...

2 Naive Bayes Classifier

2.1 Định lý Bayes

Định nghĩa: Định lý Bayes (Bayes' Theorem) là một định lý toán học để tính xác suất xảy ra của một sự kiện ngẫu nhiên A khi biết sự kiện liên quan B đã xảy ra.
Gọi A, B là hai biến cố:

Với $\mathrm{P}(\mathrm{B}) > 0$:

$$P(A|B) = P(AB)$$

Suy ra:

$$P(AB) = P(A|B)P(B) = P(B|A)P(A)$$

Công thức Bayes:

$$P(B|A) = \frac{P(AB)}{P(A)} = \frac{P(A|B)P(B)}{P(A)} = \frac{P(A|B)P(B)}{P(AB)+P(A\overline{B})}$$
$$= \frac{P(A|B)P(B)}{P(AB)+P(A\overline{B})} = \frac{P(A|B)P(B)}{P(A|B)P(B)+P(A|\overline{B})P(\overline{B})}$$

Trong đó:

A và B là hai biến cố xảy ra độc lập với nhau.

Công thức Bayes tổng quát:
Với $P(A) > 0$ và $\{B_1, B_2, B_3, ..., B_n\}$ là một hệ đầy đủ các biến cố
Tổng xác suất của hệ bằng 1:

$$\sum_{k=1}^{n} P(B_k) = 1 \tag{2.1.1}$$

Từng đôi 1 xung khắc:

$$P(B_i \cap B_j) = 0 \tag{2.1.2}$$

Khi đó ta có:

$$P(B_k|A) = \frac{P(A|B_k)P(B_k)}{P(A)} = \frac{P(A|B_k)P(B_k}{\sum_{i=1}^{n} P(A|B_i)P(B_i)} \tag{2.1.3}$$

2.2 Naive Bayes Classifier

Xét bài toán classification với C classes 1,2,3,4,... C. Giả sử có một điểm dữ liệu này rơi vào class c. Nói cách khác, hãy tính: $p(y = c|\mathbf{x})$ hoặc viết ngắn gọn thành $p(c|\mathbf{x})$ Tức tính xác suất để đầu ra là class c biết đầu vào là vector $\mathbf{x}$ 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 để xác định class của điểm dữ liệu bằng cách chọn class có xác suất điểm dữ liệu đó rơi vào cao nhất:

$$c = \underset{c\in\{1,2,...C\}}{\arg\max}\ p\big(c|\mathbf{x}\big) \tag{2.2.1}$$

Biểu thức trong dấu argmax ở 2.2.1 nhìn chung khó có cách tính trực tiếp. Thay vào đó, quy tắc Bayes thường được sử dụng:

$$c = \arg\max_c \frac{p\big(\mathbf{x}|c\big)}{p\big(\mathbf{x}\big)} = \arg\max_c p\big(\mathbf{x}|c\big)p\big(c\big) \tag{2.2.2}$$

Dấu bằng thứ hai xảy ra theo quy tắc Bayes, dấu bằng thứ ba xảy ra vì $\mathrm{p}\big(\mathrm{x}\big)$ ở mẫu số không phụ thuộc vào c. Tiếp tục quan sát, $\mathrm{p}\big(\mathrm{c}\big)$ có thể được hiểu là xác suất để một điểm bất kỳ rơi vào class c. Nếu training set lớn, nó có thể được xác định bằng maximum likelihood estimation (MLE) - là tỷ lệ giữa điểm thuộc class c và số điểm trong training set. Nếu training set nhỏ, giá trị này có thể được ước lượng bằng maximum a posteriori (MAP). Cách thứ nhất thường được sử dụng nhiều hơn.

Thành phần còn lại $\mathrm{p}\big(\mathrm{x}|\mathrm{c}\big)$, tức phân phối dữ liệu của các điểm trong class c, thường rất khó tính toán vì x là một biến ngẫu nhiên nhiều chiều. Để có thể ước lượng được phân phối đó, training set phải rất lớn. Để giúp cho việc tính toán được đơn giản, người ta thường giả sử rằng các thành phần của biến ngẫu nhiên x là đọc lập với nhau khi đã biết c:

$$p\big(x|c\big) = p\big(x_1, x_2, ..., x_d|c\big) = \prod_{i=1}^{d} p\big(x_i|c\big) \tag{2.2.3}$$

Giả thiết các chiều của dữ liệu độc lập với nhau là quá chặt và trên thực tế,ít khi tìm được dữ liệu mà các thành phần hoàn toàn độc lập với nhau. Tuy nhiên, giả thiết *ngây thơ* $\big(naive\big)$ này đôi khi 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 label của dữ liệu dựa trên giả thuyết này có tên là *naive Bayes classifier* $\big(NBC\big)$

Ở bước huấn luyện, các phân phối $p\big(c\big)$ và $p\big(x_i|c\big), i = 1, ..., d$ sẽ được xác định dựa vào dữ liệu huấn luyện. Việc xác định các giá trị này có thể dựa vào MLE hoặc MAP.

Ở bước kiểm thử,label của một điểm dữ liệu mới $\mathbf{x}$ được xác định bởi:

$$c = \underset{c\in\{1,....,C\}}{\arg\max} \prod_{i=1}^{d} p\big(x_i|c\big) \tag{2.2.4}$$

Khi d lớn và các xác suất nhỏ, biểu thức ở vế phải của 2.2.3 là một số rất nhỏ, khi tính toán có thể gặp sai số. Để giải quyết việc này, 2.2.3 thường được viết lại dưới dạng tương đương bằng cách lấy log của vế phải:

$$c = \underset{c\in\{1,....,C\}}{\arg\max} \left(\log\big(p\big(c\big)\big) + \sum_{i=1}^{d} \log\big(p\big(x_i|c\big)\big) \right) \tag{2.2.5}$$

Sự đơn giản của NBC mang lại hiệu quả đặc biệt trong các bài toán phân loại văn bản, ví dụ bài toán lọc tin nhắn hoặc email rác. Trong phần sau của chương này, chúng em sẽ xây dựng một bộ lọc email rác tiếng Anh đơn giản. Cả việc huấn luyện và kiểm thử của NBC là cực kỳ nhanh khi so với các phương pháp phân loại phức tạp khác. Việc giả sử các thành phần trong dữ liệu là đọc lập với nhau khiến cho việc tính toán mỗi phân phối

$p(x_i|c)$ không mất nhiều thời gian.

Việc tính toán $p(x_i|c)$ phụ thuộc vào loại dữ liệu. Có ba loại phân bố xác suất thường được sử dụng phổ biến là *Gausian naive Bayes, multinomial naive Bayes,* và *Bernoulli Naive*. Chúng ta cùng xem xét vào từng loại.

3 Các dạng phân phối

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

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

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

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

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.

3.2 Multinomial Naive Bayes

Mô hình Multinomial Naive Bayes được sử dụng khi chúng ta có dữ liệu rời rạc. Vì thế hay triển khai trong bài toán phân loại văn bản (document classification) sử dụng Bags of words để tạo vector đặc trưng. 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:

$$\lambda_{ci} = p\left(x_i \mid c\right) = \frac{N_{ci}}{N_c} \tag{3.2.1}$$

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.
- N_c là tổng số từ (kể cả lặp) xuất hiện trong class c. Nói cách khác, nó bằng tổng độ dài của toàn bộ các văn bản thuộc vào class c. Có thể suy ra rằng $N_c = \sum_{i=1}^{d} N_{ci}$, từ đó $\sum_{i=1}^{d} \lambda_{ci} = 1$.

Cách tính này có một hạn chế là nếu có một từ mới chưa bao giờ xuất hiện trong class c thì biểu thức (3.2.1) sẽ bằng 0, điều này dẫn đến vế phải của (2.2.5) bằng 0 bất kể các giá trị còn lại có lớn thế nào. Việc này sẽ làm cho kết quả không chính xác.

Để giải quyết việc này, một kỹ thuật được gọi là Laplace smoothing được áp dụng:

$$\hat{\lambda}_{ci} = \frac{N_{ci} + \alpha}{N_c + d\alpha} \tag{3.2.2}$$

Với α là một số dương, thường bằng 1, để tránh trường hợp tử số bằng 0. Mẫu số được cộng với α 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: $\hat{\lambda}_c = \left\{\hat{\lambda}_{c1,\ldots,}\hat{\lambda}_{cd}\right\}$

3.3 Bernouli Naive Bayes

Mô hình này được áp dụng cho các loại dữ liệu mà mỗi thành phần là một giá trị nhị phân - bằng 0 hoặc 1(**Binary**).Ví dụ, cũng với loại văn bản, ta chỉ cần quan tâm tới từ đó có xuất hiện hay không.
Khi đó, $p(x_i|c)$ được tính bằng:

$$p(x_i|c) = p(i|c)x_i + \big(1 - p(i|c)\big)\big(1 - x_i\big) \tag{3.3.1}$$

trong đó: $P(i|c)$ là xác suất bức thư đó là spam.

4 Ví dụ

4.1 Dự đoán khách hàng mua máy tính

Một cửa hàng trong quá trình bán máy tính, người ta đã thu được một bảng dữ liệu khách hàng và kết quả khách hàng đó có mua sản phẩm hay không như bảng dưới đây:

ID	Age	Income	Student	Credit rating	Result
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle	medium	no	excellent	yes
13	middle	high	yes	fair	yes
14	senior	medium	no	excellent	no

Giả sử ta có một khách hàng mới An có các thuộc tính(age = youth, income = medium, student = yes, credit_rating = fair). Bây giờ cần xác định xem An có thuộc lớp C_{yes} (mua máy tính) hay không, ta tính toán như sau:

$$\begin{aligned}
P(C_{yes}) &= 9/14 = 0.643 \\
P(C_{no}) &= 9/14 = 0.357 \\
P(age = youth|C_{yes}) &= 2/9 = 0.222 \\
P(age = youth|C_{no}) &= 3/5 = 0.6 \\
P(income = medium|C_{yes}) &= 4/9 = 0.444 \\
P(income = medium|C_{no}) &= 2/5 = 0.4 \\
P(student = yes|C_{yes}) &= 6/9 = 0.667 \\
P(student = yes|C_{no}) &= 1/5 = 0.2 \\
P(credit_rating = fair|C_{yes}) &= 6/9 = 0.667 \\
P(credit_rating = fair|C_{no}) &= 2/5 = 0.2
\end{aligned}$$

Cuối cùng:

$P(An|C_{yes}) = 0.222 * 0.444 * 0.667 * 0.667 = 0.044$
$P(An|C_{no}) = 0.60.4 * 0.2 * 0.4 = 0.019$
$P(An|C_{yes})*P(C_{yes}) = 0.044 * 0.643 = 0.0283$
$P(An|C_{no})*P(C_{no}) = 0.019 * 0.357 = 0.007$

Từ kết quả này ta thấy $P(An\ |C_{yes})P(C_{yes})$ có giá trị lớn nhất, do đó thuật toán Bayes sẽ kết luận là khách hàng An sẽ mua máy tính.

4.2 Dự đoán khách hàng mua máy tính với sklearn

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

```
import pandas as pd
from sklearn.naive_bayes import MultinomialNB

#Đọc bảng dữ liệu thông tin và kết quả của khách hàng
data = pd.read_csv('/content/Data.csv')

#Tách kết quả trong training
res = data['result']
#Tách các đặc trừng training ra khỏi cột kết quả
data_train = data.drop(columns='result',axis=0)

#Dữ liệu test
data_train.loc[len(data_train.index)] = ['youth', 'medium', 'yes','fair']
data_train = pd.get_dummies(data_train,drop_first=True)
an = data_train.iloc[[14]]
data_train = data_train.drop(labels=14, axis=0)

#gọi MultinomialNB
clf = MultinomialNB()
clf.fit(data_train, res)
print('Xác suất An chọn không mua máy và mua máy:', clf.predict_proba(an))
print('An có mua máy tính hay không: ',clf.predict(an))
```

Kết quả:

```
Xác suất An chọn không mua máy và mua máy: [[0.18066604 0.81933396]]
An có mua máy tính hay không:  ['yes']
```

Với kết quả thu được đầu ra dự đoàn các phần hoàn toàn trùng khớp.

5 Siêu tham số của các phân phối trong thư viện scikit-learn

5.1 Gaussian Naive Bayes

- priors : Mảng chứa xác suất tiên nghiệm (prior probabilities) của các lớp. Nếu được khai báo trước thì các xác suất này sẽ được giữ cố định mà không thay đổi dựa trên dữ liệu thực tế
- var_smoothing : float, mặc định=1e-9. Giá trị phương sai (variance) do người dùng khai báo để cộng vào phương sai của các thuộc tính để tăng tính ổn định

5.2 Multinomial Naive Bayes

- alpha (float, mặc định = 1.0). Giá trị để làm mịn Laplace (Laplace smoothing).
- fit_prior (bool, mặc định=True). Dùng để quyết định có sử dụng các xác suất tiên nghiệm được tính toán ra hay không. Nếu không, phân phối xác suất đồng nhất (Uniform prior probability) sẽ được dùng thay thế

- class_prior: Mảng chứa xác suất tiên nghiệm (prior probabilities) của các lớp. Nếu được khai báo trước thì các xác suất này sẽ được giữ cố định mà không thay đổi dựa trên dữ liệu thực tế

5.3 Bernoulli Naive Bayes

- alpha (float, mặc định = 1.0). Giá trị để làm mịn Laplace (Laplace smoothing).
- binarizer (float hoặc None, mặc định=0.0). Giá trị ngưỡng để thực hiện chuyển đổi các đặc trưng sang dạng nhị phân. Nếu giá trị của đặc trưng lớn hơn ngưỡng thì sẽ được gán giá trị 1, ngược lại là 0. Nếu không khai báo giá trị binarizer thì thuật toán sẽ mặc định input đã ở dạng nhị phân sẵn
- fit_prior (bool, mặc định=True). Dùng để quyết định có sử dụng các xác suất tiên nghiệm được tính toán ra hay không. Nếu không, phân phối xác suất đồng nhất (Uniform prior probability) sẽ được dùng thay thế
- class_prior Mảng chứa xác suất tiên nghiệm (prior probabilities) của các lớp. Nếu được khai báo trước thì các xác suất này sẽ được giữ cố định mà không thay đổi dựa trên dữ liệu thực tế

Các siêu tham số trên có thể do người dùng quyết định hoặc dò tìm bằng các thuật toán tự động. Có nhiều phương pháp dò tìm bộ siêu tham số tốt nhất cho các thuật toán máy học như Random Search, Grid Search, Thuật toán tiến hóa... Trong bài toán này, nhóm chúng em sử dụng thuật toán Grid Search để tìm bộ siêu tham số cho các dạng phân phối, chi tiết thực hiện sẽ được trình bày kỹ ở phần thực nghiệm

6 Thực nghiệm

6.1 Bài toán

Phân loại bài báo tiếng Việt vào để xác định bài báo đó thuộc thể loại nào trong 8 thể loại: Chính trị, xã hội, pháp luật, kinh doanh , đời sống, sức khỏe, thế giới, thể thao, văn hóa
- Đầu vào: Một đoạn văn bản chưa nội dung của 1 trong 8 thể loại trên
- Đầu ra: Nhãn của 1 trong 8 thể loại trên

6.2 Dữ liệu

Bộ dữ liệu gồm có 16000 bài báo. Số lượng mỗi bào báo trong các thể loại được mô tả trong hình 1

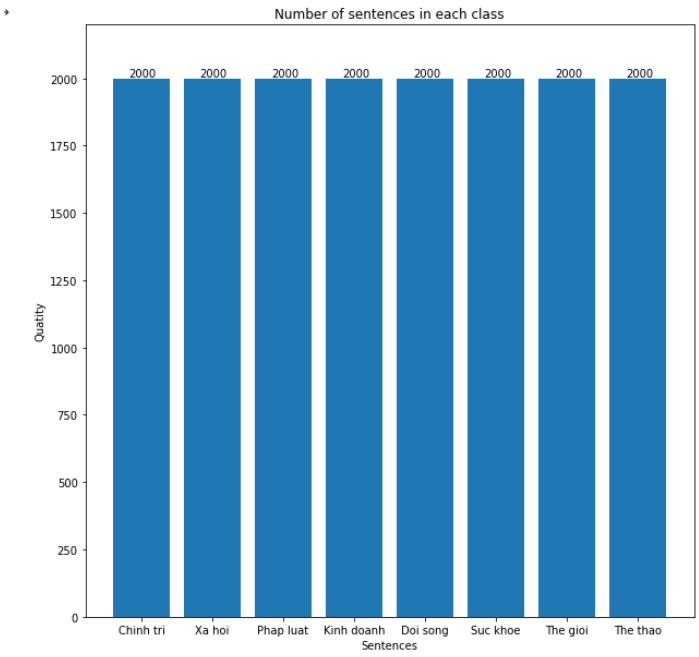
Bài báo dài nhất	8013 từ vựng
Bài báo ngắn nhất	5 từ vựng
Độ dài bài báo trung bình	353 từ vựng

Bảng 1: Thống kê độ dài các bài báo

6.3 Phương pháp

6.3.1 Tiền xử lý dữ liệu

Sau khi tiến hành thống kê dữ liệu nhóm đã tiến hành tiền xử lý dữ liệu bằng cách loại bỏ các ký tự kéo dài, tách từ sử dụng bộ tách từ VnCoreNLP để mô hình hiểu đúng nghĩa của từ, giảm kích thước của bộ từ vựng và biểu diễn các đặc trưng có độ chính xác cao



Hình 1: Số lượng bài báo trong mỗi thể loại

hơn.
Sau đó tiến hành chia tập train tập test với tỷ lệ 8:2 để thử nghiệm các mô hình.

6.3.2 Trích xuất đặc trưng

Nhóm đã sử dụng 2 phương pháp trích xuất đặc trưng để thực nghiệm và so sánh giữa các mô hình đối với 2 phương pháp này đó là Bag-of-Words và TF-IDF(Term Frequency - Inverse Document Frequency):
- Bag-of-Words: Khi sử dụng phương pháp này, chúng ta sẽ thu được một ma trận mà trong đó, mỗi hàng sẽ đại diện cho một văn bản, mỗi cột đại diện cho một từ có trong từ điển, và mỗi ô (cell) sẽ chứa tần suất xuất hiện của từ trong văn bản tương ứng.
- TF-IDF: đây là một phương pháp cực kì phổ biến trong xử lý văn bản. Trọng số này được sử dụng để đánh giá tầm quan trọng của một từ trong một văn bản. Giá trị cao thể hiện độ quan trọng cao và nó phụ thuộc vào số lần từ xuất hiện trong văn bản nhưng bù lại bởi tần suất của từ đó trong tập dữ liệu. Thường được áp dụng trong các bài toán như tóm tắt văn bản, phân loại văn bản.
Sau khi trích xuất đặc trưng do số chiều của vector đặc trưng lớn cần phải có cấu hình phần cứng cao nên nhóm đã sử dụng kỹ thuật Singular Value Decomposition (SVD) để giảm số chiều của vector đặc trưng.Tuy nhiên khi sử dụng phương pháp này thì các giá trị trong vector đặc trưng sẽ có giá trị âm nên không thể thử nghiệm với Naive Bayes sử dụng hàm phần phối Multinomial

6.3.3 Tinh chỉnh siêu tham số của mô hình

Việc thay đổi giá trị các siêu tham số sẽ ảnh hưởng đến độ chính xác của mô hình do đó nhóm đã tiến hành tinh chỉnh tham số bằng thuật toán Grid search.

Grid Search là một thuật toán tìm kiếm vét cạn hoạt động bằng cách tạo ra các bộ tham số trong phạm vi tìm kiếm do người dùng khai báo. Các bộ tham số này sẽ được thử nghiệm lần lượt và sau đó trả về kết quả tốt nhất và bộ tham số tương ứng, ví dụ giá trị của 2 parameter lần lượt từ 0-9. Grid Search sẽ lần lượt ghép từng giá trị của siêu tham số 1 với siêu tham số 2 để tính toán độ chính xác của mô hình.
Để sử dụng Grid search tinh chỉnh các tham số trong Naive Bayes thì phải tạo bộ từ điển trong đó chìa khóa là tên của tham số. Giá trị của từ điển là các giá trị khác nhau của

tham số. Ví dụ để tinh chỉnh siêu tham số "var smoothing" trong mô hình Naive Bayes với hàm phân phối Gauuse thì triển khai như sau:

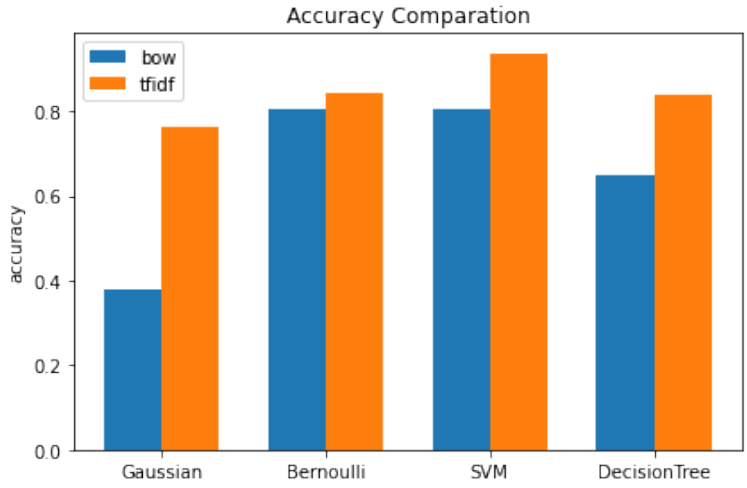
```
nb_classifier = GaussianNB()
params_NB = {"var_smoothing": np.logspace(0,-9, num=100)}
gs_NB = GridSearchCV(estimator=nb_classifier,
                     param_grid=params_NB,verbose=1, scoring='accuracy')
```

Trong đó:

"estimator": Là mô hình cần tinh chỉnh ở đây là mô hình GausianNB
"param_grid": Là các siêu tham số của mô hình cần tinh chỉnh tham số
"verbose":Độ dài thông báo hiển thị thông tin trong quá trình tinh chỉnh
"scoring": Phương thức đánh giá để tìm bộ tham số
Ngoài ra còn có 1 số tham số khác trong grid search có thể tham khảo thêm tại Sklearn

Tương tự nhóm cũng đã tinh chỉnh siêu tham số cho mô hình Naive Bayes với hàm phân phối Bernouli Naive bayes. Ngoài ra nhóm cũng tiến hành thử nghiệm thêm các mô hình Support Vector Machine và Decision Tree để so sánh với mô hình Naive Bayes.

6.4 Kết quả chạy thí nghiệm các mô hình



Hình 2: Biểu đồ so sánh kết quả các mô hình

Model	Accuracy
Gaussian + BoW	0.378
Gaussian + TF-IDF	0.761
Bernoulli + BoW	0.804
Bernoulli + TF-IDF	0.843
SVM + BoW	0.804
SVM + TF-IDF	0.938
Decision Tree + BoW	0.649
Decision Tree + TF-IDF	0.838

Bảng 2: Bảng thống kê kết quả thí nghiệm

- **Thời gian chạy**: Trong quá trình thí nghiệm các mô hình thì nhóm nhận thấy rằng mô hình Naive Bayes đào tạo nhanh hơn so với mô hình SVM và Decision Tree. Thời gian để sử dụng thuật toán gridsearch để tìm kiếm siêu tham số ít hơn rất nhiều so

với các thuật toán như SVM, Decision Tree.

- **Độ chính xác**: Kết quả tương đối tốt so với các mô hình thử nghiệm. Trong đó mô hình Bernouli khi sử dụng đặc trưng của phương pháp trích xuất đặc trưng TF-IDF cho kết quả độ chính xác đặt 0.843, tốt thứ 2 chỉ sau mô hình SVM khi sử dụng đặc của của TF-IDF là 0.938.

- Trong 2 phương pháp trích xuất đặc trưng từ dữ liệu văn bản thì phương pháp trích xuất đặc trưng TF-IDF đều cho kết quả tốt hơn phương pháp Bag of Words

7 Kết luận

7.1 Ưu điểm của mô hình

- Giả định độc lập: hoạt động tốt cho nhiều bài toán/miền sữ liệu và ứng dụng.
- Đơn giản nhưng đủ tốt để giải quyết nhiều bài toán như phân lớp văn bản, lọc spam...
- Cho phép kết hợp tri thức tiền nghiệm (prior knowledge) và dữ liệu quan sát được (obserwed data).
- Tốt khi có sự chênh lệch số lượng giữa các lớp phân loại.
- Huấn luyện mô hình (ước lượng tham số) dễ và nhanh.

7.2 Nhược điểm của mô hình

- Giả định độc lập của dữ liệu dù trên thực tế cho kết quả tốt nhưng vẫn chưa đủ cho những dữ liệu có độ phức tạp hơn
- Hầu hết các trường hợp thực tế trong đó có các thuộc tính trong các đối tượng thường phụ thuộc lẫn nhau.
- Vấn đề zero (đã nêu cách giải quyết ở phía trên)
- Mô hình không được huẩn luyện bằng phượng pháp tối ưu mạnh và chặt chẽ.
- Tham số mủa mô hình là các ước lượng xác suất điều kiện đơn lẻ.
- Không tính đến sự tương tác giữa các ước lượng này.

Link google colab thực nghiệm

Tài liệu

[1] Vũ Hữu Tiệp - Naive Bayes Classifier

[2] Mô hình Bag-of-words trong xử lý văn bản

[3] Phân loại văn bản với phương pháp trọng số TF-IDF

[4] Singular Value Decomposition

[5] GridSearchCV