ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

——————— * ———————

BÀI TẬP LỚN

NHẬP MÔN HỌC MÁY VÀ KHAI PHÁ DỮ LIỆU

PHÂN LOẠI THƯ RÁC
(EMAIL SPAM FILTERING)

Sinh viên thực hiện	**Mã sinh viên**
Trần Đức Hải	20194270
Nguyễn Mạnh Duy	20194262
Nguyễn Phương Nam	20194336
Nguyễn Quốc Phương	20194355
Từ Hoàng Giang	20183518

Giảng viên hướng dẫn : TS. Nguyễn Nhật Quang

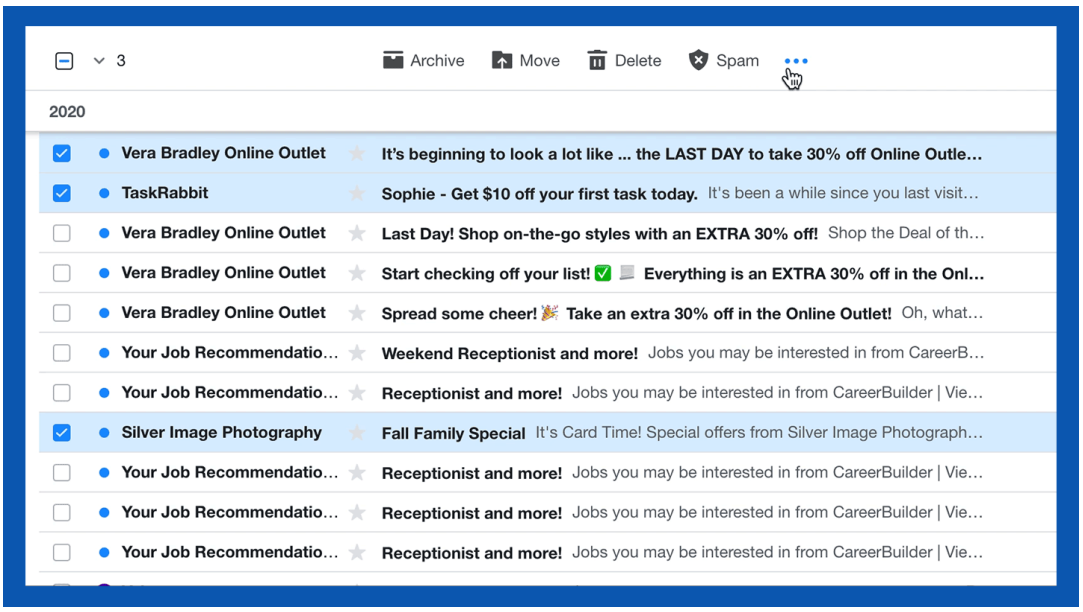
Hà Nội, tháng 1 năm 2023

MỤC LỤC

I. Giới thiệu và mô tả bài toán

1. Đặt vấn đề

Trong thời đại công nghệ phát triển, thư rác vẫn luôn là vấn đề nhức nhối mà nhiều người sử dụng dịch vụ thư điện tử gặp phải. Phát tán thư rác là hành vi gửi thư điện tử mà người nhận không mong muốn, thường với nội dung quảng cáo, được gửi hàng loạt với số lượng lớn tới một tập hợp người nhận không phân biệt. Kẻ phát tán thư rác có thể lấy địa chỉ thư điện tử từ các trang web, phòng chat, tập dữ liệu cá nhân bị rò rỉ, …



Thư rác gây ra nhiều phiền toái và thiệt hại, bao gồm, ngăn người dùng sử dụng tối ưu thời gian, dung lượng lưu trữ và băng thông mạng. Một số lượng lớn thư rác truyền trong mạng máy tính có thể phá hủy không gian nhớ của máy chủ thư điện tử, băng thông đường truyền, tài nguyên tính toán và thời gian sử dụng của thiết bị người dùng. Bên cạnh đó, thư rác làm cho các thư từ khác quan trọng hơn thay vì nhận được thì lại bị trả về cho người gửi với lí do hộp thư người nhận đã quá đầy. Vì vậy, việc nghiên cứu về bộ lọc thư rác tiên tiến và hiệu quả là điều cần thiết, không chỉ từ phía người dùng mà từ cả những nhà cung cấp.

2. Bài toán phân loại thư rác

Ngày nay có rất nhiều công nghệ lọc thư rác. Nó dựa trên các đặc trưng cơ bản của thư điện tử như: tiêu đề thư, địa chỉ người gửi, hay các cụm từ thường xuất hiện trong các thư rác. Hoặc cũng có thể so sách các

thư điện tử mẫu với các thư điện tử nhận được sau đó tìm ra thư rác. Và một công nghệ nữa là sử dụng công nghệ học máy để phân loại thư rác

Đây thực chất là bài toán phân loại văn bản hai lớp trong đó tập cơ sở dữ liệu ban đầu gồm có thư rác (spam) và thư hợp lệ (ham). Dựa trên các đặc điểm đầu vào của 1 email như: tiêu đề, địa chỉ gửi, nội dung,... Khi càng tận dụng được các thông tin này thì khả năng phân loại thư sẽ càng chính xác và kết quả phân loại cũng phụ thuộc rất nhiều vào kích thước của tập huấn luyện.

Ý tưởng của phương pháp là tìm cách xây dựng một mô hình phân loại nhằm phân loại cho một thư mới thuộc loại thư rác thay thư hợp lệ bằng cách huấn luyện các mẫu thư có sẵn. Có rất nhiều phương pháp đã được kiểm chứng có thể đem lại kết quả cao như SVM, k-NN… Tuy nhiên phương pháp Naive Bayes có hiệu quả tuy nhỏ hơn nhưng giải thuật lại đơn giản, được áp dụng khá phổ biến và hoàn toàn phù hợp với đề tài. Vì vậy, nhóm chúng em đã chọn mô hình Naive Bayes để cài đặt giải thuật phân loại thư rác cho bài tập lớn môn học.

II. Thuật toán Naive Bayes

1. Định lý Bayes

Định lý Bayes là một định lý toán học để tính xác suất xảy ra của một sự kiện ngẫu nhiên h khi biết sự kiện liên quan D đã xảy ra. Xác suất này được kí hiệu là P(h|D) và được đọc là “Xác suất của h nếu có D". Đại lượng này được gọi là xác suất có điều kiện hay xác suất hậu nghiệm vì nó được rút ra từ giá trị được cho của D hoặc phụ thuộc vào giá trị đó.

$$P(h \mid D) = \frac{P(D \mid h).P(h)}{P(D)}$$

Theo định lý Bayes, xác suất xảy ra h khi biết D phụ thuộc vào 3 yếu tố:

- P (h): Xác suất trước (tiên nghiệm) của giả thiết (phân loại) h
- P (D) Xác suất trước (tiên nghiệm) của việc quan sát được dữ liệu D
- P (D | h): Xác suất (có điều kiện) của việc quan sát được dữ liệu D, nếu biết giả thiết (phân loại h) là đúng

2. Phân lớp Naive Bayes

Phân loại là một vấn đề cơ bản của lĩnh vực máy học và khai phá dữ liệu. Mục đích của một thuật toán phân loại là xây dựng một phân lớp từ một tập các ví dụ huấn luyện đã được gán nhãn lớp.

Biểu diễn bài toán phân loại: Một tập D_train trong đó mỗi ví dụ học x được biểu diễn là một vector n chiều: (x1, x2, …, xn), một tập xác định các nhãn lớp: C = {c1, c2, c3,...cn}. Với một ví dụ mới z thì z sẽ được phân loại vào lớp nào?

Phân lớp Naive Bayes là một trong những bộ phân loại trong thống kê Bayes dựa trên định lý Bayes. Nó có thể dự đoán được xác suất của các lớp bộ phận, chẳng hạn như nó có thể dự đoán được xác suất mà một ví dụ đã cho thuộc vào một lớp nào đó. Điểm đặc biệt của Naive Bayes là giả định rằng sự ảnh hưởng của một thuộc tính là độc lập với giá trị của các thuộc tính khác trong một phân lớp đã cho. Giả thiết này được gọi là độc lập về điều kiện, điều này đã làm cho việc tính toán trở nên dễ dàng hơn.

Mục tiêu của thuật toán là xác định phân lớp có thể (phù hợp) nhất đối với z.

$$c_{NB} = \arg\max_{c_i \in C} P(c_i).\prod_{j=1}^{n} P(z_j \mid c_i)$$

Giai đoạn học (training phase), sử dụng một tập học**.** Đối với mỗi phân lớp có thể, tính các giá trị xác suất trước: P(c_i). Đối với mỗi giá trị thuộc tính xj, tính giá trị xác suất xảy ra của giá trị thuộc tính đó đối với một phân lớp c_i: P(x_j | c_i)

Giai đoạn phân lớp (classification phase), Đối với mỗi phân lớp c_i thuộc C, tính giá trị của biểu thức:

$$P(c_i).\prod_{j=1}^{n} P(x_j \mid c_i)$$

Xác định phân lớp của z là lớp có thể nhất:

$$c^* = \arg\max_{c_i \in C} P(c_i).\prod_{j=1}^{n} P(x_j \mid c_i)$$

3. Vấn đề đối với thuật toán Naive Bayes

Vấn đề 1: Trong quá trình thực hiện, nếu không có ví dụ nào gắn với phân loại c_i có giá trị thuộc tính x_j…, ta sẽ có $P(x_j \mid c_i) = 0$, và vì thế:

$$P(c_i).\prod_{j=1}^{n} P(x_j \mid c_i) = 0$$

Giải pháp để khắc phục tình trạng trên, sử dụng phương pháp Bayes để ước lượng $P(x_j \mid c_i)$:

$$P(x_j \mid c_i) = \frac{n(c_i, x_j) + mp}{n(c_i) + m}$$

- n (c_i): số lượng các ví dụ học gắn với phân lớp c_i
- n (c_i, x_j): số lượng các ví dụ học gắn với phân lớp c_i có giá trị thuộc tính xj
- p: ước lượng đối với giá trị xác suất $P(x_j \mid c_j)$: Các ước lượng đồng mức: p=1/k, với thuộc tính f_j có k giá trị có thể
- m: một hệ số để bổ sung cho n (c_i) các vị dụ thực sự được quan sát với thêm m mẫu ví dụ với ước lượng p

Vấn đề 2: Giới hạn về độ chính xác trong tính toán của máy tính. Vì $P(x_j \mid c_i) < 1$. nên khi số lượng các giá trị thuộc tính càng lớn thì tính của các xác suất tính dược sẽ gần về 0

$$\lim_{n \to \infty} \left(\prod_{j=1}^{n} P(x_j \mid c_i) \right) = 0$$

Giải pháp cho vấn đề là sử dụng logarit cho các giá trị xác suất:

$$c_{NB} = \arg\max_{c_i \in C} \left(\log \left[P(c_i) . \prod_{j=1}^{n} P(x_j \mid c_i) \right] \right)$$

III. Thực nghiệm và đánh giá

1. Chuẩn bị dữ liệu

Trong bài tập lớn, nhóm chúng em sử dụng các bộ dữ liệu từ hai nguồn chính là https://www.kaggle.com/ và https://archive.ics.uci.edu/.

Bộ dữ liệu gồm 3 trường dữ liệu. Trường 'label_text' gồm để gán nhãn thư hợp lệ (ham) hoặc thư rác (spam), trường 'label_num' gồm nhãn 0 (ham) và 1 (spam) (Đối với các tập dữ liệu khác nhau có thể có 'label_text' hoặc 'label_num'). Trường 'mail_text' gồm nội dung cái email.

Bộ dữ liệu của bài toán được thu thập khá đa dạng. Đối với các tập dữ liệu trên kaggle được thu thập từ khoảng năm 2019 - 2021. Đối với các tập dữ liệu trên UCI được thu thập cũ hơn, trong khoảng những năm 2012. Bài toán sử dụng dữ liệu từ 2 nguồn khác nhau với mục đích so sánh giữa các tập dữ liệu hiện tại và đã cũ sẽ cho ra mô hình phân loại hiệu quả như thế nào.

Danh sách các bộ dữ liệu bao gồm: Bộ 'kaggel1.csv' gồm 5728 email khác nhau, bộ 'kaggle2.csv' gồm 5172 email, bộ 'kaggle3.csv' gồm 10000 email; bộ 'uci.csv" gồm 5572 email (Các email đã cũ và thường rất ngắn).

Mỗi bộ dữ liệu được chia thành 2 bộ dữ liệu training: 'data_train' và dữ liệu test: data_test' với tỉ lệ: 8:2.

2. Tiền xử lý dữ liệu

Các email thường chứa những ký tự không phải chữ như số, dấu câu, đường link, ... hoặc các từ được viết tắt, viết hoa, các từ xuất hiện thường xuyên trong các văn bản. Những kí tự, từ này không có ích trong việc phân loại văn bản, làm nhiễu, làm giảm độ chính xác của mô hình. Do vậy, cần loại bỏ những kí tự, từ này ra khỏi văn bản, đưa các từ còn lại trong văn bản về dạng gốc.

- Các bước tiền xử lý dữ liệu:
- Chuyển tất cả ký tự hoa thành ký tự thường.
- Loại bỏ các ký tự không phải chữ

- Loại bỏ từ dừng (stopword): loại bỏ các từ thường xuyên xuất hiện trong văn bản nhưng không có ích trong phân loại văn bản như trạng từ, liên từ, giới từ.Tập stopword này được thư viện cung cấp có sẵn. Ví dụ: “about”, “at”, “again”, “after”, “all”, “to”, “the”,...
- Đưa từ về dạng gốc (lemmatization).Ví dụ: “best”, “better” -> “good”; “does”, “did” -> “do”

Trong project, nhóm đã sử dụng thư viện nltk (Natural Language Toolkit) để phục vụ cho việc loại bỏ từ dừng và đưa các từ về dạng gốc một cách hiệu quả hơn.

Ví dụ minh họa dữ liệu sau bước tiền xử lý:

	label_text	mail_text
0	ham	Yep, by the pretty sculpture
1	ham	Yes, princess. Are you going to make me moan?
2	ham	Welp apparently he retired
3	ham	Havent.
4	ham	I forgot 2 ask Ã¼ all smth.. There's a card on...
5	ham	Ok i thk i got it. Then u wan me 2 come now or...
6	ham	I want kfc its Tuesday. Only buy 2 meals ONLY ...
7	ham	No dear i was sleeping :-P
8	ham	Ok pa. Nothing problem:-)
9	ham	Ill be there on <#> ok.

	label_text	mail_text
0	ham	yep pretty sculpture
1	ham	yes make moan
2	ham	apparently retired
3	ham	havent
4	ham	forgot 's present lei want write sign
5	ham	got come
6	ham	want buy meal gravy
7	ham	sleeping
8	ham	nothing problem
9	ham	

3. Lựa chọn và trích xuất đặc trưng

Tập thư sau khi qua bước tiền xử lý sẽ được biến đổi để tạo ra tập từ khóa (vocabulary). Theo cách thông thường, tập từ khóa sẽ là tập các từ riêng biệt xuất hiện trong tập thư. Như vậy có nghĩa là một từ chỉ cần xuất hiện tối thiểu một lần trong tập thư là sẽ được đưa vào tập từ khóa và đóng góp vào việc phân loại thư rác. Với tập dữ liệu ở mức vừa, làm theo cách như vậy thường sẽ tạo ra một tập từ khóa khoảng vài nghìn thuộc tính (từ riêng biệt). Nhưng khi tham khảo một tập dữ liệu thư từ UCI (University of California Irvine), tập dữ liệu này lại không được biểu diễn dưới dạng văn bản thư từ thông thường mà lại được biểu diễn dưới dạng các vector gồm 48 thuộc tính như sau…

	word_freq_make	word_freq_address	word_freq_all	word_freq_3d	word_freq_our	word_freq_over	word_freq_remove	word_freq
0	0.00	0.64	0.64	0.0	0.32	0.00	0.00	
1	0.21	0.28	0.50	0.0	0.14	0.28	0.21	
2	0.06	0.00	0.71	0.0	1.23	0.19	0.19	
3	0.00	0.00	0.00	0.0	0.63	0.00	0.31	
4	0.00	0.00	0.00	0.0	0.63	0.00	0.31	
...	...	...	...	...	...	...	...	
4596	0.31	0.00	0.62	0.0	0.00	0.31	0.00	
4597	0.00	0.00	0.00	0.0	0.00	0.00	0.00	
4598	0.30	0.00	0.30	0.0	0.00	0.00	0.00	
4599	0.96	0.00	0.00	0.0	0.32	0.00	0.00	
4600	0.00	0.00	0.65	0.0	0.00	0.00	0.00	

4601 rows × 46 columns

Nhìn qua, ta có thể thấy mỗi vector sẽ thể hiện tần suất xuất hiện của 48 từ khác nhau trong mỗi email. Nghĩa là khi huấn luyện mô hình bằng tập dữ liệu này thì mô hình đó sẽ chỉ cần dùng 48 từ khóa duy nhất để phân loại thư rác. Điều đó đặt ra một câu hỏi rằng để đánh giá một email là thư rác thì sẽ cần phải dùng rất nhiều (vài nghìn) từ khóa hay chỉ cần dùng một số lượng nhỏ (vài chục) các từ khóa quan trọng là đã đủ cho việc phân loại?

Để trả lời cho câu hỏi đó, nhóm sẽ dùng cả hai tập từ khóa này để huấn luyện hai mô hình khác nhau và kiểm tra bằng thực nghiệm xem cách nào sẽ đem lại hiệu quả và độ chính xác cao hơn. Hiển nhiên để đảm bảo tính công bằng ta sẽ giữ nguyên tập dữ liệu huấn luyện cũng như thuật toán và chỉ thay đổi hai tập từ khóa (vocabulary) khác nhau để huấn luyện mô hình. Có như vậy thì khi so sánh kết quả và đánh giá ta mới đưa ra được những kết luận chính xác nhất được.

Trích xuất đặc trưng: Biến đổi các thư thành các vector

Nhóm sử dụng phương pháp túi từ (bag of word), là một trong những phương pháp trích xuất đặc trưng phổ biến trong lọc thư rác. Phương pháp này không quan tâm đến vị trí từ trong thư cũng như không quan tâm đến sự phụ thuộc giữa các từ.

Mỗi văn bản được chuyển đổi thành vector n chiều $< x_1, x_2, ... , x_n >$ và quan sát xem với n là số lượng từ trong tập từ khóa (vocabulary). Giá trị xi là số lần xuất hiện của từ ti trong văn bản, với ti ở trong tập từ khóa.

Sau bước lựa chọn và trích xuất đặc trưng, ta nhận được tập training gồm các vector với số chiều bằng số lượng từ trong từ khóa, giá trị các phần tử tương ứng với số lần xuất hiện các từ thuộc tập từ khóa trong mỗi văn bản.

Ví dụ đầu ra khi sử dụng tập từ khóa thông thường:

	label_text	mail_text	refund	birth	searching	sunlight	alone	crazy	marriage	favor	...
0	ham	[yep, pretty, sculpture]	0	0	0	0	0	0	0	0	...
1	ham	[yes, make, moan]	0	0	0	0	0	0	0	0	...
2	ham	[apparently, retired]	0	0	0	0	0	0	0	0	...
3	ham	[havent]	0	0	0	0	0	0	0	0	...
4	ham	[forgot, 's, present, lei, want, write, sign]	0	0	0	0	0	0	0	0	...

5 rows × 3290 columns

Ví dụ đầu ra khi sử dụng tập từ khóa rút gọn (48 thuộc tính):

	label_text	mail_text	make	address	all	3d	our	over	remove	addresses	...
0	ham	[yep, pretty, sculpture]	0	0	0	0	0	0	0	0	...
1	ham	[yes, make, moan]	1	0	0	0	0	0	0	0	...
2	ham	[apparently, retired]	0	0	0	0	0	0	0	0	...
3	ham	[havent]	0	0	0	0	0	0	0	0	...
4	ham	[forgot, 's, present, lei, want, write, sign]	0	0	0	0	0	0	0	0	...

5 rows × 50 columns

4. Huấn luyện

Với điều kiện các từ khóa là độc lập với nhau, để có thể phân loại các email là thư thường hay thư rác thì thuật toán Naive Bayes cần phải xác định hai xác suất sau (với wi là các từ khóa có trong tập từ khóa - vocabulary):

$$\mathbf{P(Spam|w_1, w_2, ..., w_n) = P(Spam) \cdot \prod_{i=1}^{n} P(w_i|Spam)}$$

$$\mathbf{P(Ham|w_1, w_2, ..., w_n) = P(Ham) \cdot \prod_{i=1}^{n} P(w_i|Ham)}$$

Ta nhận thấy các giá trị thành phần trong 2 công thức trên đều là hằng số nên quá trình huấn luyện chính là việc ta đi tính toán các giá trị này.

Để tính toán $P(w_i|Spam)$ and $P(w_i|Ham)$ bên trong các công thức ở trên, chúng ta sẽ cần sử dụng các công thức sau:

$$\mathbf{P(w_i|Spam) = \frac{N_{w_i|Spam} + 1}{N_{Spam} + N_{Vocabulary}}}$$

$$\mathbf{P(w_i|Ham) = \frac{N_{w_i|Ham} + 1}{N_{Ham} + N_{Vocabulary}}}$$

- N_{Spam} là tổng số lần xuất hiện tất cả các từ trong tập thư rác

- N_{Ham} là tổng số lần xuất hiện tất cả các từ trong tập thư thường
- $N_{Vocabulary}$ là tổng số từ trong tập từ điển

Lưu ý: Để khắc phục hạn chế tính toán của máy tính ta sẽ sử dụng hàm logarit cho các giá trị xác suất trên như sau

$$P(\text{Spam} \mid w_1, w_2, ..., w_n) = \log P(\text{Spam}) + \sum_{i=1}^{n} \log P(w_i \mid \text{Spam})$$

$$P(\text{Ham} \mid w_1, w_2, ..., w_n) = \log P(\text{Ham}) + \sum_{i=1}^{n} \log P(w_i \mid \text{Ham})$$

5. Test

Sau khi huấn luyện mô hình xong, ta có thể bắt đầu việc phân loại thư theo các bước sau:

- Với mỗi email đầu vào, đưa nó về dạng $(w_1, w_2, \ldots, w_n)$
- Tính toán $P(w_i|\text{Spam})$ and $P(w_i|\text{Ham})$.
- So sánh hai giá trị $P(w_i|\text{Spam})$ và $P(w_i|\text{Ham})$, khi đó:
  - Nếu $P(w_i|\text{Spam}) > P(w_i|\text{Ham})$ thì thư đó là thư rác.
  - Nếu $P(w_i|\text{Spam}) < P(w_i|\text{Ham})$ thì thư đó là thư thường.
  - Nếu $P(w_i|\text{Spam}) = P(w_i|\text{Ham})$ thì nghĩa là thuật toán không phân loại được thư này và cần sự trợ giúp của con người.

Lưu ý rằng một số email mới có thể sẽ chứa các từ không nằm trong tập từ vựng. Khi đó, ta chỉ cần bỏ qua những từ trong việc tính xác suất.

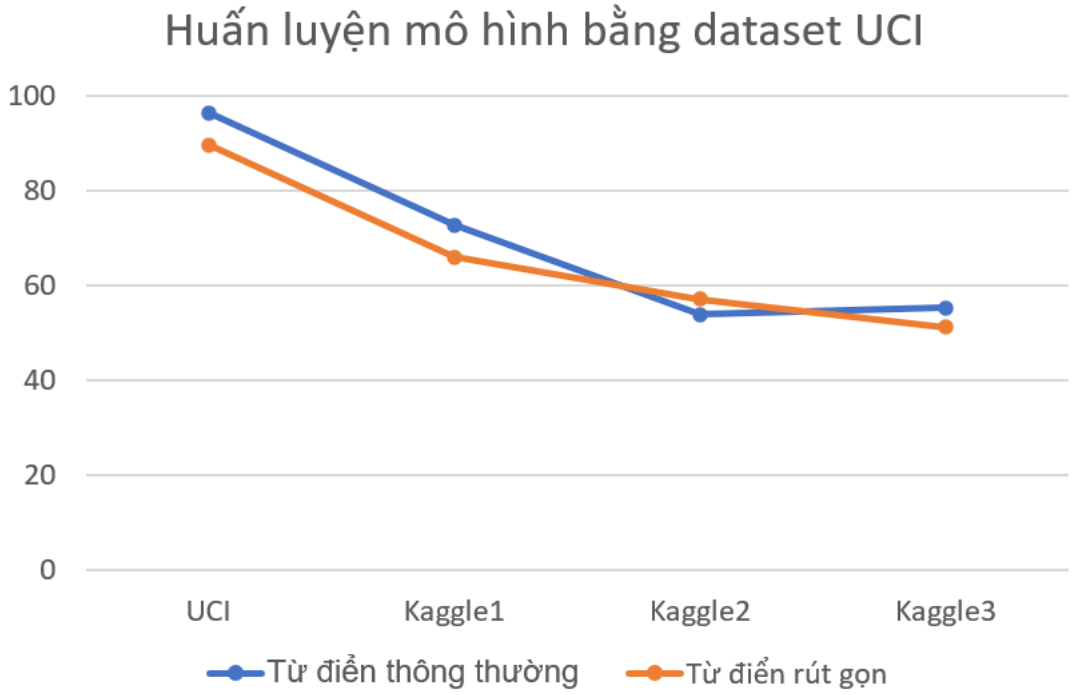
Áp dụng các bước trên với lần lượt tất cả các email trong tập dữ liệu kiểm thử. Độ chính xác sẽ được tính bằng công thức:

$$\text{Độ chính xác} = \frac{\text{Số email phân loại đúng}}{\text{Tổng số email đã phân loại}}$$

Ban đầu, nhóm sử dụng tập dữ liệu UCI.csv (tham khảo nguồn phía dưới) để chia hai phần là tập dữ liệu huấn luyện và tập dữ liệu kiểm thử. Khi tính toán độ chính xác của hai mô hình thông thường và rút gọn ta được kết quả lần lượt là 96,41% và 89.67%.

Để tăng tính thuyết phục, nhóm cũng sẽ tính độ chính xác của hai mô hình trên với những 3 tập dữ liệu khác là kaggle1.csv, kaggle2.csv, kaggle3.csv (tham khảo nguồn phía dưới). Làm như vậy để tránh việc

mô hình xây dựng bằng từ điển thông thường có độ chính xác cao hơn nhờ việc mô hình được huấn luyện bằng tập huấn luyện có cùng đặc trưng dữ liệu với tập kiểm thử (do thực chất hai tập được chia từ một tập gốc ban đầu). Kết quả khi test bằng các tập dữ liệu khác nhau được thể hiện bằng đồ thị sau:



Tập dữ liệu test / Cách biểu diễn	UCI (5572 mail)	Kaggle1 (5728 mail)	Kaggle2 (5171 mail)	Kaggle3 (10000 mail)
Sử dụng từ điển thông thường (3290 thuộc tính)	96.41%	72.73%	53.78%	55.22%
Sử dụng từ điển rút gọn (48 thuộc tính)	89.67%	65.95%	57.14%	51.22%

6. Khó khăn gặp phải và đánh giá

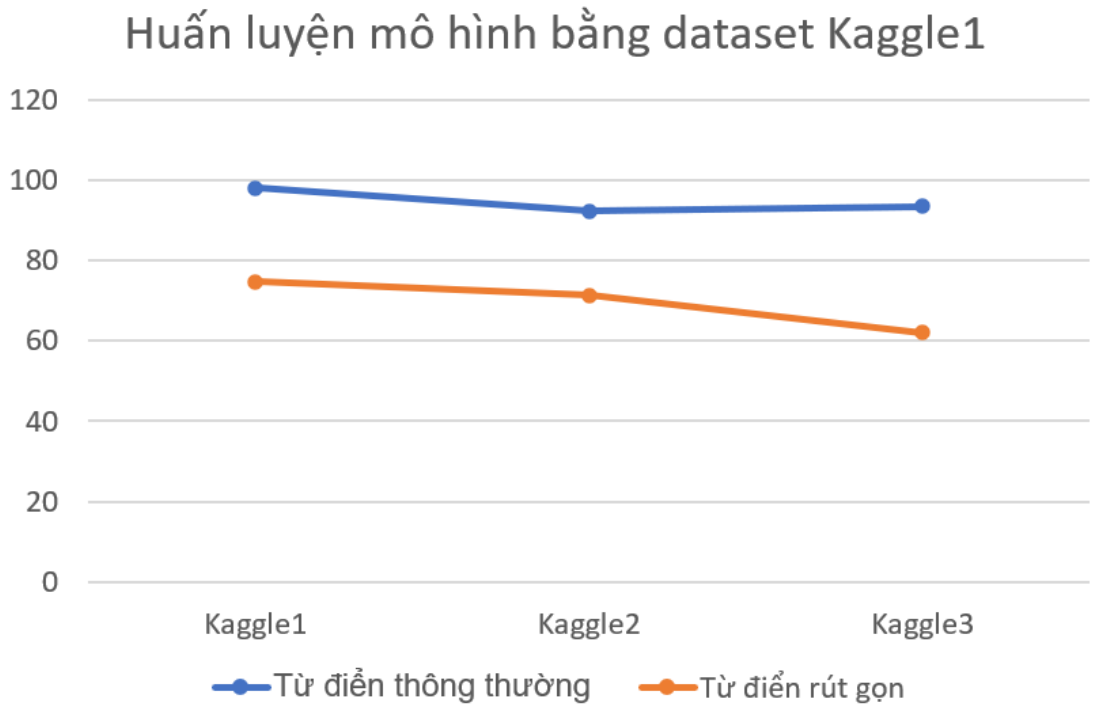
Vấn đề 1: Thông qua so sánh từ đồ thị trên, ta nhận thấy sự tương quan về độ hiệu quả giữa hai mô hình vẫn chưa thực ra rõ ràng. Khi độ chính xác khi test bằng mô hình huấn luyện từ điển rút gọn khá là tương đồng so với độ chính xác của mô hình thông thường (có lúc còn nhỉnh hơn). Vì thế nên ta chưa thể kết luận được mô hình nào tốt hơn.

Vấn đề 2: Khi huấn luyện mô hình bằng tập dữ liệu UCI.csv và test mô hình đó với các tập dữ liệu từ nguồn khác. Ta nhận thấy độ chính

xác bị giảm đi đáng kể, nghĩa là mô hình trên vẫn chưa thực sự tốt. Việc độ chính xác của hai mô hình đều giảm tương tự nhau khi test bằng tập dữ liệu khác càng chứng minh điều đó.

Giải thích cho vấn đề này, việc độ chính xác của cả hai mô hình đều giảm có thể là do dữ liệu huấn luyện từ UCI.csv đã cũ (từ năm 2012). Còn dữ liệu còn lại thì mới hơn (khoảng từ năm 2019-2021). Ta đã biết một mô hình học máy rất khó để giữ độ hiệu quả theo thời gian, chắc chắn sẽ có một thời điểm mà ta phải huấn luyện lại mô hình. Trường hợp này cũng không ngoại lệ, vì các tập thư ở thời điểm gần đây và năm 2012 có văn phong khác nhau nên đặc trưng dữ liệu khác nhau. Đó là lý do tại sao mô hình huấn luyện bằng tập dữ liệu cũ lại trở nên kém hiệu quả như vậy.

Giải pháp: Cả hai vấn đề trên sẽ được giải quyết khi ta huấn luyện mô hình bằng tập dữ liệu mới thay vì UCI.csv, cụ thể là ta sẽ sử dụng tập dữ liệu kaggle1.csv (tham khảo nguồn phía dưới). Sau khi huấn luyện 2 mô hình bằng tập dữ liệu mới và test, ta được kết quả như sau

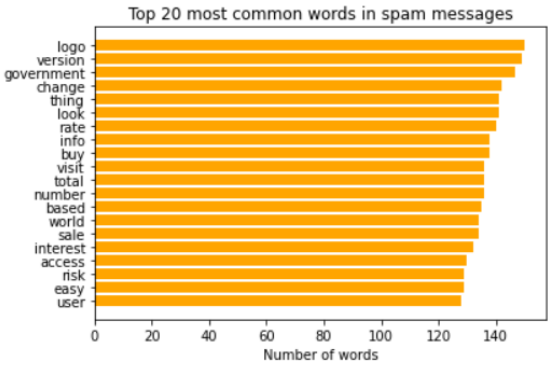


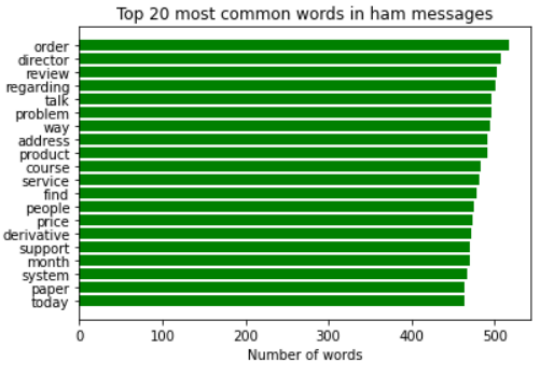
Tập dữ liệu test / Cách biểu diễn	Kaggle1 (5728 mail)	Kaggle2 (5171 mail)	Kaggle3 (10000 mail)
Sử dụng từ điển thông thường (12742 thuộc tính)	98.08%	92.32%	93.61%
Sử dụng từ điển rút gọn (48 thuộc tính)	74.78%	71.37%	62.08%

Sau khi huấn luyện mô hình bằng dữ liệu mới, ta nhận thấy mô hình huấn luyện với từ điển thông thường đã có hiệu quả cao hơn rất nhiều. Còn mô hình huấn luyện với từ điển rút gọn cũng tăng nhưng không đáng kể. Như vậy, ta đã giải quyết được vấn đề 1. Nhưng không thể kết luận là sử dụng từ điển thông thường sẽ hiệu quả hơn từ điển rút gọn vì ta thấy độ chính xác của từ điển rút gọn dù không cao nhưng vẫn khá ổn định. Mặc dù huấn luyện mô hình theo cách thông thường có thể cho độ chính xác tối ưu nhất. Nhưng với cách còn lại, nếu độ chính xác ở mức độ chấp nhận được, thì việc tìm ra tập từ điển rút gọn tối ưu như vậy sẽ rất tốt vì nó giúp giảm rất nhiều thời gian tính toán. Không chỉ thế, nó còn giúp mô hình mang tính khái quát cao hơn, giúp mô hình vẫn có thể hoạt động tốt với những dữ liệu mới trong tương lai. Vậy mục tiêu tiếp theo của nhóm sẽ là tối ưu tập từ điển rút gọn này.

Một ý tưởng đơn giản mà nhóm đã nghĩ ra để tìm tập từ khóa cải tiến đó là dữ liệu huấn luyện sau khi qua bước tiền xử lý, sẽ được tách ra

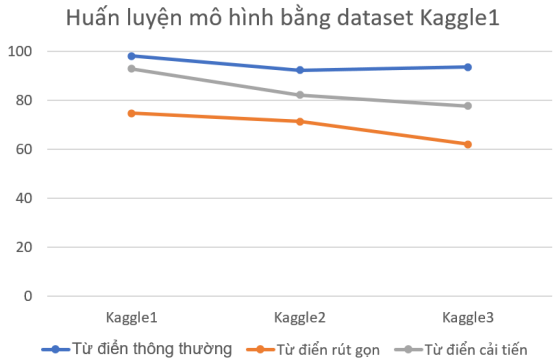
làm hai phần là tập thư spam và tập thư thường. Sau đó tìm top 100 từ có số lần xuất hiện nhiều nhất trong mỗi tập thư rồi gộp lại, loại bỏ các từ trùng nhau và đó chính là tập từ khóa cải tiến cần tìm.





Các bước tiếp theo, ta đem tập từ khóa cải tiến này để đi so sánh độ chính xác với 2 tập từ khóa còn lại tương tự như các bước trước, ta được kết quả như sau:

Cách biểu diễn \ Tập dữ liệu test	Kaggle1 (5728 mail)	Kaggle2 (5171 mail)	Kaggle3 (10000 mail)
Sử dụng từ điển thông thường (12742 thuộc tính)	98.08%	92.32%	93.61%
Sử dụng từ điển rút gọn (48 thuộc tính)	74.78%	71.37%	62.08%
Sử dụng từ điển rút gọn cải tiến (157 thuộc tính)	92.93%	82.17%	77.65%



Ta nhận thấy khi huấn luyện bằng tập cải tiến, độ chính xác đã tăng đáng kể so với khi huấn luyện bằng tập rút gọn (48 từ khóa) mà vẫn giữ số lượng thuộc tính nhỏ (157 từ) nên thời gian tính toán vẫn sẽ nhanh. Sở dĩ độ chính xác của mô hình giảm khi test với các tập dữ liệu khác là do tập từ điển cải tiến này được xây dựng từ kaggle1.csv nên sẽ chỉ tối ưu cho tập đó. Nếu muốn tăng độ chính xác ta chỉ cần tăng số lượng từ khóa trong tập cải tiến nhưng điều đó cũng sẽ làm tăng thời gian chạy.

IV. Kết luận

Project đã trình bày về thuật toán Naive Bayes và áp dụng thuật toán trên vào việc xây dựng mô hình bài toán phân loại thư rác. Qua đây, ta thấy được sức mạnh và khả năng của học máy trong việc xử lý những công việc trong đời sống con người, từ đó có thể áp dụng mô hình để giải quyết bài toán trong tương lai.

Qua quá trình kiểm thử và đánh giá, mô hình đã đáp ứng được yêu cầu lọc thư rác với độ chính xác khá cao. Tuy nhiên, thời gian chạy của

mô hình còn chưa được tối ưu, đặc biệt đối với bộ dữ liệu có số lượng tương đối lớn.

Từ việc so sánh các cách biểu diễn tập từ khóa, ta nhận thấy việc phân loại thư rác thật sự phụ thuộc phần lớn vào một số lượng nhỏ các từ. Việc sử dụng nhiều hay ít từ khóa cho việc phân loại là sự đánh đổi giữa thời gian chạy và độ chính xác. Nếu có cơ hội tạo ra tập cải tiến từ một lượng dữ liệu đủ lớn, đủ đa dạng, đến từ nhiều nguồn thì nhóm tin rằng mô hình được huấn luyện bằng tập từ khóa đó sẽ cân bằng được cả yếu tố thời gian chạy cũng như độ chính xác.

Phương hướng phát triển trong tương lai: Tiếp tục cải thiện thuật toán để tối ưu thêm về độ chính xác cũng như thời gian chạy với bất kỳ cách biểu diễn dữ liệu nào. Ngoài ra, nhóm cũng sẽ tìm hiểu thêm các phương hướng để tối ưu tập từ điển rút gọn để giúp cải thiện hơn nữa độ khái quát và chính xác của mô hình phân loại thư rác.

V. Tài liệu tham khảo

[1] Slide bài giảng "Nhập môn Học máy và Khai phá dữ liệu" - Thầy Nguyễn Nhật Quang, Trường Công nghệ thông tin và truyền thông, Đại học Bách Khoa Hà Nội

[2] https://viblo.asia/p/thuat-toan-phan-lop-naive-bayes-924lJWPm5PM

[3] Code cơ sở: https://www.kdnuggets.com/2020/07/spam-filter-python-naive-bayes-scratch.html

[4] Dataset 48 thuộc tính: https://archive.ics.uci.edu/ml/datasets/spambase

[5] Dataset UCI.csv: https://archive.ics.uci.edu/ml/datasets/sms+spam+collection

[6] Dataset kaggle1.csv:

https://www.kaggle.com/code/harshsinha1234/email-spam-classification-nlp/data

[7] Dataset kaggle2.csv: https://www.kaggle.com/datasets/venky73/spam-mails-dataset

[8] Dataset kaggle1.csv

https://www.kaggle.com/datasets/nitishabharathi/email-spam-dataset