

BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC KINH TẾ TP. HỒ CHÍ MINH
TRƯỜNG CÔNG NGHỆ VÀ THIẾT KẾ
KHOA CÔNG NGHỆ THÔNG TIN KINH DOANH



ĐỒ ÁN MÔN HỌC
DỮ LIỆU LỚN VÀ ỨNG DỤNG

Đề tài:

**PHÂN TÍCH, DỰ ĐOÁN CHỦ ĐỀ CỦA CÁC VIDEO TRÊN NỀN TẢNG
YOUTUBE, XÂY DỰNG HỆ THỐNG GỢI Ý VIDEO**

Nhóm sinh viên:

Họ và tên	MSSV	Lớp
Trần Phạm Hải Nam	31211027651	DS001
Lý Minh Nguyên	31211027653	DS001
Trần Duy Tuấn	31211027683	DS001
Nguyễn Nhật Thảo Vy	31211025542	DS001

Giảng viên: Ts. Đặng Nhân Cách

Thành phố Hồ Chí Minh, ngày 30 tháng 03 năm 2024

MỤC LỤC

CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI.....	8
1.1 Giới thiệu đề tài.....	8
1.2 Mục tiêu nghiên cứu	8
1.3 Phương pháp nghiên cứu:	9
1.4 Tài nguyên sử dụng.....	9
CHƯƠNG 2: CƠ SỞ LÝ THUYẾT	10
2.1 Xử lý ngôn ngữ tự nhiên.....	10
2.2 Phân loại văn bản.....	10
2.3 Phân loại Naive Bayes.....	10
2.3.1 Định lý Bayes	10
2.3.2 Bộ phân lớp Naive Bayes:	11
2.4 Hồi quy Logistic	11
2.5 Mô hình LSTM (Long Short-Term Memory)	13
2.5.1 Cơ sở logic của LSTM.....	13
2.6 Các phương pháp Vec tơ hóa văn bản:.....	14
2.6.1 Mô hình Word2Vec:	14
2.6.2 Mô hình Bag of words	15
2.6.3 Mô hình TF-IDF:	15
2.7 Recommender System	15
2.7.1 Khái niệm	15
2.7.2 Phân loại	16
2.7.2.1 Simple Recommenders.....	16
2.7.2.2 Lọc dựa trên nội dung (Content-based filtering)	16

2.7.2.3	Lọc cộng tác (Collaborative filtering)	16
2.7.2.4	Phương pháp lai ghép (Hybrid Method)	16
2.8	Framework Flask	17
 CHƯƠNG 3: PHƯƠNG PHÁP THU THẬP DỮ LIỆU VÀ TỔNG QUAN BỘ DỮ LIỆU		
		18
3.1.	Sơ lược bộ dữ liệu	18
3.2	Phương pháp thu thập dữ liệu	18
3.2.1	Youtube Data API	18
3.2.2	Thu thập dữ liệu	19
3.2.3	Mô tả thuộc tính bộ dữ liệu:	25
 CHƯƠNG 4: TIỀN XỬ LÝ, KHÁM PHÁ VÀ PHÂN TÍCH DỮ LIỆU		
		27
4.1	Phân tích khám phá dữ liệu (Exploratory Data Analysis – EDA)	27
4.2	Tiền xử lý dữ liệu	30
4.3	Trực quan hóa kết quả sau tiền xử lý	31
 CHƯƠNG 5: HUẤN LUYỆN DỮ LIỆU DỰA TRÊN CÁC MÔ HÌNH		
		35
5.1	Mô hình Naive Bayes	35
5.1.1	Chuyển đổi văn bản và tách tập dữ liệu huấn luyện	35
5.1.2	Huấn luyện mô hình và đánh giá chung	36
5.2	Mô hình Logistic Regression	37
5.2.1	Chuyển đổi văn bản và tách tập dữ liệu huấn luyện	37
5.2.2	Huấn luyện mô hình và đánh giá chung	38
5.3	Mô hình LSTM	39
5.3.1	Chuẩn bị dữ liệu huấn luyện	39
5.3.2	Xây dựng mô hình Word2Vec	40
5.3.3	Huấn luyện mô hình và đánh giá chung	41

5.3.4 Đánh giá chung	44
5.4 So sánh kết quả.....	44
CHƯƠNG 6: XÂY DỰNG HỆ THỐNG GỢI Ý VÀ DEMO GIAO DIỆN NGƯỜI DÙNG (RECOMMENDATION SYSTEM & UI DEMO).....	47
6.1 Hệ thống gợi ý video	47
6.1.1 Dữ liệu gợi ý.....	47
6.1.2 Xây dựng hệ thống gợi ý	48
6.1.2.1 Quy tắc hoạt động.....	48
6.1.2.2 Xây dựng mô hình vector hóa và hàm tính toán độ tương đồng	48
6.2 Xây dựng demo giao diện người dùng	50
6.2.1 Đoạn mã thực hiện.....	50
6.2.2 Hướng dẫn khởi chạy chương trình.....	53
6.2.3 Minh họa kết quả thực hiện	54
6.2.4 Các lỗi thường gặp và cách khắc phục (Poka-yoke)	57
6.2.4.1 Lỗi đường dẫn.....	57
6.2.4.2 Lỗi thiếu thư viện.....	57
CHƯƠNG 7: KẾT LUẬN VÀ ĐÁNH GIÁ.....	59
7.1 Kết quả đạt được.....	59
7.2 Hạn chế	59
7.3 Các phương pháp cải tiến đề xuất	60
PHỤ LỤC	62
BẢNG PHÂN CÔNG.....	62
TÀI LIỆU THAM KHẢO	63

DANH MỤC HÌNH ẢNH

Hình 1. Phát biểu định lý Bayes	10
Hình 2. Minh họa cách thức hoạt động của mô hình Hồi quy Logistic	12
Hình 3. Hoạt động bên trong mạng LSTM.....	13
Hình 4. Ba cổng thông tin bên trong mạng LSTM.....	14
Hình 5. Giao diện tổng quan của Google Cloud console sau khi truy cập vào “APIs & Services”	19
Hình 6. Giao diện tạo project của Google Cloud console	20
Hình 7. Sử dụng thanh tìm kiếm cho thư viện “YouTube Data API v3”	20
Hình 8. API key cuối cùng để sử dụng cào dữ liệu.	21
Hình 9. Hàm python thực hiện cào dữ liệu.....	22
Hình 10. Tùy chỉnh các tham số và thực hiện chạy hàm.....	23
Hình 11. Thống kê mô tả cột ‘word count’	27
Hình 12. Top 25 từ xuất hiện nhiều nhất.....	28
Hình 13. Top 25 từ xuất hiện ít nhất	29
Hình 14. Kết quả sau Lemmatization và Tokenization	31
Hình 15. Top 25 từ xuất hiện nhiều nhất (sau tiền xử lý)	32
Hình 16. Top 25 từ xuất hiện ít nhất (sau tiền xử lý)	32
Hình 17. Word Cloud kết quả cột 'Title_Description'	33
Hình 18. Mô hình Bigram.....	34
Hình 19. Ánh xạ giữa nhãn đã mã hóa và nhãn ban đầu	36
Hình 20. Điểm số đánh giá trên các lớp của mô hình Naive Bayes	37
Hình 21. Pipeline của mô hình Logistic Regression	38
Hình 22. Điểm số đánh giá trên các lớp của mô hình Logistic Regression	39
Hình 23. Cấu trúc mô hình LSTM.....	42
Hình 24. Điểm số đánh giá trên các lớp của mô hình LSTM.....	44
Hình 25. Biểu đồ cột so sánh 4 điểm số đánh giá của 3 mô hình.....	45
Hình 26. Ma trận nhầm lẫn của 3 mô hình	45
Hình 27. Thời gian huấn luyện của 3 mô hình	46
Hình 28. Kết quả gợi ý của hệ thống với một video bất kỳ.....	50
Hình 29. Preview tệp tin “app.py” thực hiện xây dựng backend cho trang web thực nghiệm	51
Hình 30. Preview tệp tin “index.html” thực hiện xây dựng giao diện mặc định cho trang web thực nghiệm	52
Hình 31. Preview tệp tin “result.html” thực hiện xây dựng giao diện kết quả cho trang web thực nghiệm	52
Hình 32. Giới thiệu các bước thực hiện khởi chạy ứng dụng thực nghiệm	53
Hình 33. Thực hiện khởi chạy development server đã xây dựng bằng Flask	54
Hình 34. Trang web thực nghiệm ban đầu được xây dựng bằng HTML và CSS	55

Hình 35. Trang web Wikipedia nội dung đoạn văn nhóm sử dụng để thực nghiệm mô hình	55
Hình 36. Kết quả phân lớp của trang web	56
Hình 37. Phần nội dung về Hệ thống gợi ý sử dụng Độ tương đồng Cosin.....	56
Hình 38. Ví dụ minh họa về lỗi đường dẫn có thể gặp trong quá trình khởi chạy tệp tin	57
Hình 39. Ví dụ minh họa về lỗi thiếu thư viện có thể gặp trong quá trình khởi chạy tệp tin	58

DANH MỤC BẢNG BIỂU

Bảng 1. Kết quả cào các tập dữ liệu con bằng API từ nền tảng Youtube.	24
Bảng 2. Data dictionary (mô tả biến thuộc tính)	26

CHƯƠNG 1: TỔNG QUAN ĐỀ TÀI

1.1 Giới thiệu đề tài

Trong thời đại thông tin bùng nổ, cánh cửa cho những khám phá và ứng dụng không giới hạn được mở ra. Đằng sau cánh cửa ấy là dữ liệu lớn, một nguồn tài nguyên quý giá mà nhiều người muốn khai thác, đặc biệt là dữ liệu từ các nền tảng phương tiện truyền thông xã hội. Hiện nay, Youtube là nền tảng mạng xã hội chia sẻ video lớn nhất thế giới với lượng dữ liệu khổng lồ; việc khai thác dữ liệu video của Youtube có tiềm năng ứng dụng cao, hứa hẹn nhiều cơ hội và phương pháp công nghệ mới trong nhiều lĩnh vực. Với hàng triệu video được tải lên mỗi ngày, việc phân tích và dự đoán chủ đề trở nên thiết yếu, giúp người dùng tiếp cận nội dung phù hợp, các nhà quảng cáo định hình chiến lược marketing cũng như giúp các nhà sáng tạo nội dung phần nào nắm bắt được xu hướng của nền tảng.

Đề tài *'Phân tích, dự đoán chủ đề của các video trên nền tảng Youtube, xây dựng hệ thống gợi ý video'* là một ý tưởng nhằm khai thác và phân tích lượng dữ liệu khổng lồ từ Youtube - nền tảng chia sẻ video phổ biến nhất thế giới. Mục tiêu của đề tài là triển khai phương pháp khai thác dữ liệu hiệu quả để thu thập thông tin từ video, từ đó dự đoán chủ đề chính cho nội dung của video dựa trên các phân tích tiêu đề và mô tả video tương ứng. Mở rộng hơn nữa, đề tài cũng xây dựng một hệ thống gợi ý video thông minh, với hy vọng tối ưu trải nghiệm cá nhân hóa của người dùng. Đề tài này không chỉ là trải nghiệm của nhóm trong việc tiếp xúc với dữ liệu lớn, nó còn có ý nghĩa trong việc khám phá, kết hợp và ứng dụng những kiến thức đã học như khai phá dữ liệu hay trí tuệ nhân tạo để khai thác và phân tích dữ liệu lớn. Là một dự án quan trọng trong việc hiểu biết và tận dụng hiệu quả nguồn dữ liệu phong phú này, đồng thời mở ra những tiềm năng nghiên cứu mới. Trên cơ sở này, đề tài này không chỉ đem lại sự hiểu biết cơ bản về tài nguyên dữ liệu của YouTube mà còn đóng góp vào việc phát triển các kỹ thuật và ứng dụng trong các lĩnh vực khai thác dữ liệu lớn.

1.2 Mục tiêu nghiên cứu

Mục đích chính của đề tài là khai thác được dữ liệu của Youtube, sau đó phân tích các thông tin thu thập được để phục vụ cho mục tiêu dự đoán chủ đề và xây dựng hệ thống gợi ý nội dung:

- Thu thập và xử lý dữ liệu: Mục tiêu này đặt ra để thu thập thông tin về video trên YouTube một cách hiệu quả và tự động. Từ đó tạo ra một tập dữ liệu đủ lớn và đa dạng để phân tích và dự đoán chủ đề cũng như xây dựng hệ thống gợi ý.
- Phân tích và dự đoán chủ đề: Sử dụng các kỹ thuật xử lý ngôn ngữ tự nhiên, máy học và học sâu, mục đích là để phân tích tiêu đề và mô tả của các video để dự

đoán chủ đề của video đó. Qua đó, chúng ta có thể hiểu rõ hơn về nội dung và một số chủ đề nhất định của các video trên nền tảng YouTube.

- Xây dựng hệ thống gợi ý video: Mục tiêu này là phát triển một hệ thống gợi ý nội dung, dựa trên dự đoán chủ đề và phân tích trước đó. Hệ thống này được kỳ vọng sẽ giúp cải thiện trải nghiệm cá nhân hóa của người dùng, giúp họ tìm kiếm và khám phá những video phù hợp với sở thích và nhu cầu của mình.

1.3 Phương pháp nghiên cứu:

Để đạt được những mục tiêu kể trên, nhóm tiến hành nhiều bước nghiên cứu và ứng dụng nhiều kiến thức, kỹ thuật và kỹ năng, gồm:

- Thu thập dữ liệu: nhóm sẽ sử dụng các API (Giao diện lập trình ứng dụng) của YouTube để thu thập dữ liệu về các video, bao gồm tiêu đề, mô tả, nhãn (tags), số lượt xem, số lượt thích, v.v... Dữ liệu này sẽ là nguồn cung cấp chính cho mục tiêu phân tích và dự đoán chủ đề cũng như xây dựng hệ thống gợi ý.
- Tiền xử lý dữ liệu và khám phá dữ liệu: Trước khi tiến hành phân tích, cần thực hiện các bước tiền xử lý dữ liệu như loại bỏ dữ liệu trùng lặp, xử lý dữ liệu thiếu, và chuẩn hóa dữ liệu v.v... để đảm bảo tính nhất quán và chất lượng của dữ liệu. Sau đó, trực quan hóa để khám phá, truy xuất và khai thác những thông tin có giá trị để có cái nhìn sơ lược về bộ dữ liệu.
- Phân tích và dự đoán Chủ đề: Nhóm sẽ sử dụng các mô hình học máy, học sâu và xử lý ngôn ngữ tự nhiên để phân tích và dự đoán chủ đề của các video. Cụ thể, nhóm sử dụng các kỹ thuật gồm Count Vectorizer, Word2Vec, Naive Bayes, Linear Regression, LSTM để xác định chủ đề chính của mỗi video dựa trên tiêu đề và mô tả tương ứng.
- Xây dựng hệ thống gợi ý video: Sau khi đã dự đoán chủ đề cho các video, chúng tôi sẽ xây dựng một hệ thống gợi ý video. Hệ thống này sẽ sử dụng các kỹ thuật như TF-IDF, Logistic Regression để đề xuất các video phù hợp dựa trên sở thích và hành vi trước đó của người dùng theo hướng lọc cộng tác (Collaborative filtering).

1.4 Tài nguyên sử dụng

- Ngôn ngữ sử dụng: Ngôn ngữ lập trình Python, HTML, CSS
- Bộ dữ liệu: [Personality-Youtube data](#)

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT

Để hiểu rõ hơn về đề tài mà nhóm triển khai, sau đây sẽ là phần giới thiệu sơ bộ về cơ sở lý thuyết chung về các nội dung và đặc biệt là những mô hình học máy và cách thức xử lý ngôn ngữ nhóm sẽ tiến hành thực hiện trong phần sau của đồ án.

2.1 Xử lý ngôn ngữ tự nhiên

Công nghệ xử lý ngôn ngữ tự nhiên (NLP) là một phần của máy học, cho phép máy tính hiểu, tương tác và diễn giải ngôn ngữ con người. Trong thời đại hiện nay, các tổ chức sở hữu một lượng lớn dữ liệu bao gồm các cuộc trò chuyện và văn bản từ nhiều nguồn khác nhau như email, tin nhắn, mạng xã hội, video, âm thanh, v.v. Họ tận dụng phần mềm NLP để tự động xử lý dữ liệu này, phân tích ý định và cảm xúc trong các tin nhắn, cũng như cung cấp phản hồi thời gian thực từ con người.

2.2 Phân loại văn bản

Trong NLP, phân loại văn bản theo chủ đề (Topic classification) là quá trình đánh nhãn hoặc gán một nhãn cho một tài liệu văn bản dựa trên nội dung của nó. Mục tiêu của phân loại văn bản theo chủ đề là xác định chủ đề chính hoặc danh mục mà tài liệu thuộc về, giúp tổ chức và quản lý thông tin hiệu quả. Các phương pháp phân loại văn bản có thể sử dụng các kỹ thuật máy học, đặc biệt là học máy có giám sát, để huấn luyện mô hình dự đoán chủ đề của các văn bản dựa trên các đặc trưng và mẫu đã được xác định trước.

2.3 Phân loại Naive Bayes

2.3.1 Định lý Bayes

Với $P(A) > 0$ và $\{B_1, B_2, \dots, B_n\}$ là một hệ đầy đủ các biến cố:

□ Tổng xác suất của hệ bằng 1:

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

□ Từng đôi một xung khắc:

$$P(B_i \cap B_j) = 0$$

Khi đó ta có:

$$\begin{aligned} P(B_k | A) &= \frac{P(A | B_k) P(B_k)}{P(A)} \\ &= \frac{P(A | B_k) P(B_k)}{\sum_{i=1}^n P(A | B_i) P(B_i)} \end{aligned}$$

Hình 1. Phát biểu định lý Bayes

Trong đó ta gọi A là một chứng cứ (evidence) (trong bài toán phân lớp A sẽ là một phần tử dữ liệu), B là một giả thiết nào để cho A thuộc về một lớp C nào đó. Trong bài toán phân lớp chúng ta muốn xác định giá trị $P(B/A)$ là xác suất để giả thiết B là đúng với chứng cứ A thuộc vào lớp C với điều kiện ra đã biết các thông tin mô tả A. $P(B/A)$ là một xác suất hậu nghiệm (posterior probability hay posteriori probability) của B với điều kiện A.

2.3.2 Bộ phân lớp Naive Bayes:

Bộ phân lớp Naive bayes hay bộ phân lớp Bayes (simple byes classifier) hoạt động như sau:

Gọi D là tập dữ liệu huấn luyện, trong đó mỗi phần tử dữ liệu X được biểu diễn bằng một vector chứa n giá trị thuộc tính $A_1, A_2, \dots, A_n = \{x_1, x_2, \dots, x_n\}$

Giả sử có m lớp C_1, C_2, \dots, C_m . Cho một phần tử dữ liệu X, bộ phân lớp sẽ gán nhãn cho X là lớp có xác suất hậu nghiệm lớn nhất. Cụ thể, bộ phân lớp Bayes sẽ dự đoán X thuộc vào lớp C_i nếu và chỉ nếu:

$$P(C_i|X) > P(C_j|X) \quad (1 \leq i, j \leq m, i \neq j)$$

Giá trị này sẽ tính dựa trên định lý Bayes.

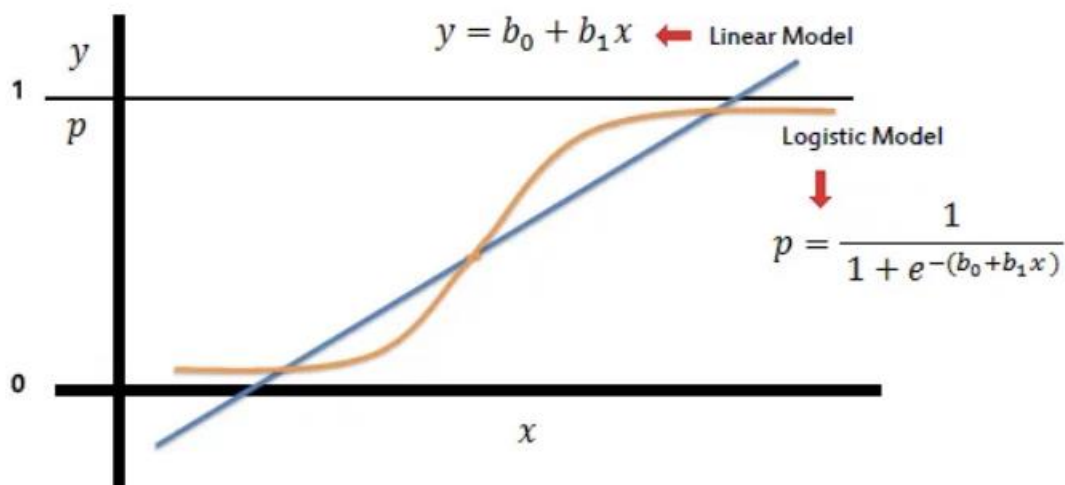
Để tìm xác suất lớn nhất, ta nhận thấy các giá trị $P(X)$ là giống nhau với mọi lớp nên không cần tính. Do đó ta chỉ cần tìm giá trị lớn nhất của $P(X|C_i) * P(C_i)$. Chú ý rằng $P(C_i)$ được ước lượng bằng $|D_i|/|D|$, trong đó D_i là tập các phần tử dữ liệu thuộc lớp C_i . Nếu xác suất tiên nghiệm $P(C_i)$ cũng không xác định được thì ta coi chúng bằng nhau $P(C_1) = P(C_2) = \dots = P(C_m)$, khi đó ta chỉ cần tìm giá trị $P(X|C_i)$ lớn nhất.

Khi số lượng các thuộc tính mô tả dữ liệu là lớn thì chi phí tính toán $P(X|C_i)$ là rất lớn, do đó có thể giảm độ phức tạp của thuật toán Naive Bayes giả thiết các thuộc tính độc lập nhau. Khi đó ta có thể tính:

$$P(X|C_i) = P(x_1|C_i) \dots P(x_n|C_i)$$

2.4 Hồi quy Logistic

Hồi quy Logistic là một phương pháp thống kê phổ biến được áp dụng để phân loại các mẫu vào hai nhóm, được gọi là phân loại nhị phân. Điểm đặc biệt của hồi quy Logistic là sử dụng hàm sigmoid, một hàm phi tuyến tính, để chuyển đổi đầu vào thành xác suất rơi vào một trong hai nhóm.



Hình 2. Minh họa cách thức hoạt động của mô hình Hồi quy Logistic

Hồi quy Logistic hoạt động dựa trên hàm Sigmoid, được biểu diễn như sau:

$$S(z) = 1/(1 + e^{-z})$$

Hàm Sigmoid nhận đầu vào là một giá trị z bất kỳ, và trả về đầu ra là một giá trị xác suất nằm trong khoảng $[0,1]$. Khi áp dụng vào mô hình Hồi quy Logistic với đầu vào là ma trận dữ liệu X và trọng số w , ta có $z = Xw$.

Việc huấn luyện của mô hình là tìm ra bộ trọng số w sao cho đầu ra dự đoán của hàm Sigmoid gần với kết quả thực tế nhất. Để làm được điều này, ta sử dụng hàm mất mát (Loss Function) để đánh giá hiệu năng của mô hình. Mô hình càng tốt khi hàm mất mát càng nhỏ.

Hàm mất mát (Loss Function) là một hàm số được sử dụng để đo lường mức độ lỗi mà mô hình của chúng ta tạo ra khi dự đoán các kết quả từ dữ liệu đầu vào. Trong bài toán Hồi quy Logistic, chúng ta sử dụng hàm mất mát Cross-Entropy (còn gọi là Log Loss) để đánh giá hiệu năng của mô hình.

Hàm mất mát Cross-Entropy được định nghĩa như sau:

$$L(w) = - \frac{1}{n} \sum_{i=1}^n [y_i \log p_i + (1-y_i) \log(1-p_i)] ,$$

Trong đó:

- n : số lượng mẫu dữ liệu trong tập huấn luyện.
- y_i : giá trị thực tế của đầu ra thứ i .
- p_i : xác suất dự đoán thuộc lớp 1 của mô hình cho đầu vào thứ i .

Hàm Cross-Entropy đo lường khoảng cách giữa hai phân phối xác suất y_i và p_i . Khi mô hình dự đoán chính xác, tức là nếu $y_i=1$ thì p_i càng gần 1, và nếu $y_i=0$ thì p_i càng gần 0, sau đó hàm mất mát sẽ tiến gần về 0.

Trong quá trình huấn luyện, chúng ta tìm cách cập nhật bộ trọng số w sao cho giá trị hàm mất mát Cross-Entropy đạt giá trị nhỏ nhất, dẫn đến một mô hình dự đoán tốt nhất.

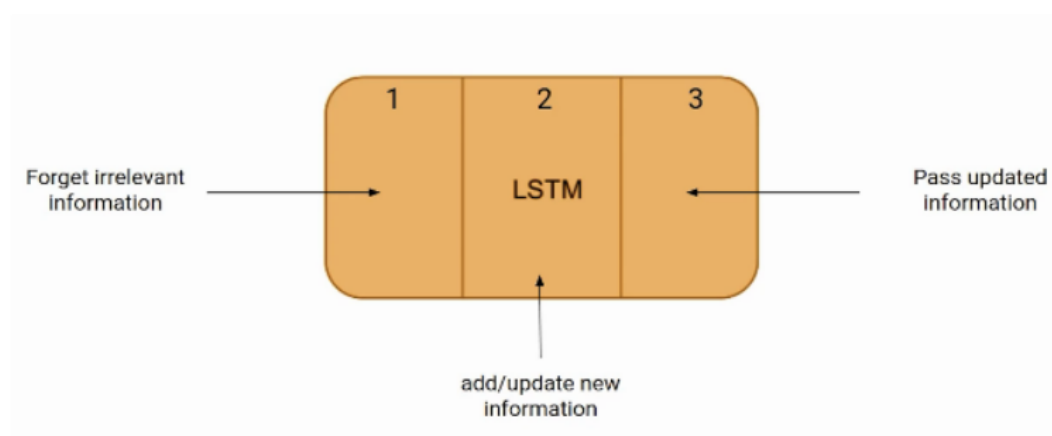
Để tìm giá trị tối ưu cho bộ trọng số w , chúng ta có thể sử dụng kỹ thuật Gradient Descent. Tại mỗi bước lặp, chúng ta cập nhật w theo phương hướng ngược với đạo hàm của hàm mất mát $L(w)$ theo w .

2.5 Mô hình LSTM (Long Short-Term Memory)

LSTM (Long Short-Term Memory) là một kiến trúc mạng nơ-ron tái phát (RNN) rộng rãi được sử dụng trong Học Sâu. Nó xuất sắc trong việc nắm bắt các phụ thuộc dài hạn, làm cho nó lý tưởng cho các nhiệm vụ dự đoán chuỗi.

Khác với các mạng nơ-ron truyền thống, LSTM tích hợp các kết nối phản hồi, cho phép nó xử lý toàn bộ chuỗi dữ liệu, không chỉ là các điểm dữ liệu cá thể. Điều này làm cho nó rất hiệu quả trong việc hiểu và dự đoán các mẫu trong dữ liệu tuần tự như chuỗi thời gian, văn bản và âm thanh.

Ở mức cao, LSTM hoạt động rất giống như một ô RNN. Dưới đây là cách hoạt động nội bộ của mạng LSTM. Kiến trúc mạng LSTM bao gồm ba phần, như được hiển thị trong hình dưới đây, và mỗi phần thực hiện một chức năng cá nhân.



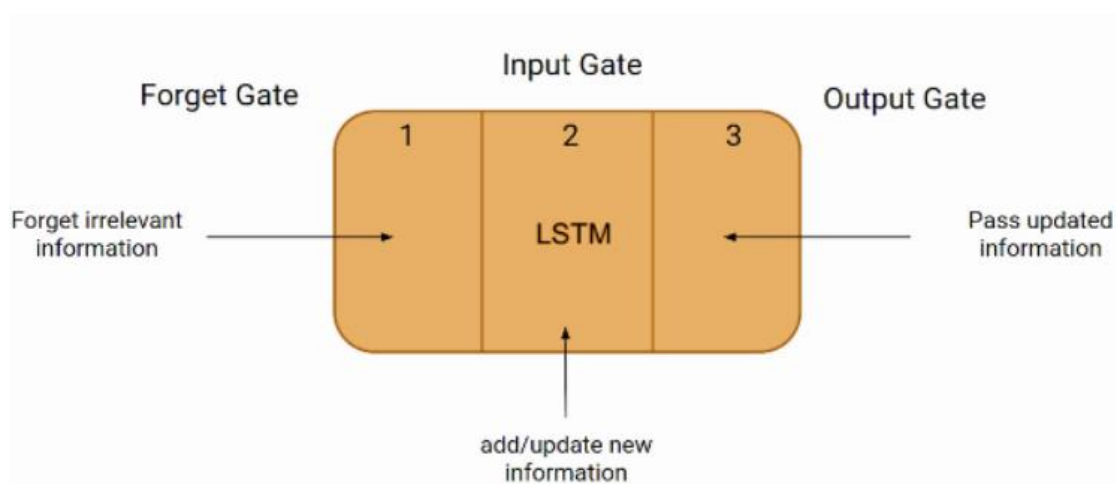
Hình 3. Hoạt động bên trong mạng LSTM.

2.5.1 Cơ sở logic của LSTM

Phần đầu tiên quyết định liệu thông tin từ thời điểm trước đó có nên được ghi nhớ hay là không quan trọng và có thể quên đi. Trong phần thứ hai, ô LSTM cố gắng học thông

tin mới từ đầu vào của ô này. Cuối cùng, trong phần thứ ba, ô chuyển thông tin được cập nhật từ thời điểm hiện tại sang thời điểm tiếp theo. Một chu kỳ của LSTM này được coi là một bước thời gian đơn lẻ.

Ba phần này của một đơn vị LSTM được biết đến là các cổng. Chúng điều khiển luồng thông tin vào và ra khỏi ô nhớ hoặc ô LSTM. Cổng đầu tiên được gọi là Cổng Quên, cổng thứ hai được biết đến là Cổng Đầu Vào, và cổng cuối cùng là Cổng Đầu Ra. Một đơn vị LSTM bao gồm ba cổng này và một ô nhớ hoặc ô lstm có thể được coi là một lớp của các nơ-ron trong mạng nơ-ron lan truyền tiến truyền thông, với mỗi nơ-ron có một lớp ẩn và một trạng thái hiện tại



Hình 4. Ba cổng thông tin bên trong mạng LSTM

2.6 Các phương pháp Vec tơ hóa văn bản:

2.6.1 Mô hình Word2Vec:

Mô hình Word2Vec là một phương pháp quan trọng trong lĩnh vực xử lý ngôn ngữ tự nhiên và học sâu. Mục tiêu của Word2Vec là biểu diễn từng từ trong văn bản dưới dạng các vector số học trong không gian nhiều chiều sao cho các từ có ý nghĩa tương tự sẽ có biểu diễn vector gần nhau trong không gian này.

Mô hình Word2Vec thường được huấn luyện trên một lượng lớn dữ liệu văn bản. Có hai phương pháp chính để huấn luyện Word2Vec: Skip-gram và Continuous Bag of Words (CBOW). Trong phương pháp Skip-gram, mô hình cố gắng dự đoán các từ xung quanh một từ cho trước từ một từ đầu vào, trong khi CBOW dựa trên ngược lại, cố gắng dự đoán từ một từ bằng cách sử dụng các từ xung quanh nó.

Kết quả của quá trình huấn luyện Word2Vec là một tập hợp các vector từ, trong đó mỗi từ trong từ điển được biểu diễn bằng một vector. Đặc biệt, các vector này có thể biểu

diễn mối quan hệ ngữ nghĩa giữa các từ, ví dụ, các vector của các từ có ý nghĩa tương tự sẽ gần nhau trong không gian vector.

2.6.2 Mô hình Bag of words

Mô hình túi từ là một cách đơn giản để chuyển đổi từ thành biểu diễn số trong xử lý ngôn ngữ tự nhiên. Mô hình này là một kỹ thuật nhúng tài liệu đơn giản dựa trên tần số của từ. Theo quan niệm, chúng ta coi toàn bộ tài liệu như một "túi" các từ, thay vì một chuỗi. Chúng ta đại diện cho tài liệu chỉ bằng tần số của mỗi từ. Sử dụng kỹ thuật này, chúng ta có thể nhúng một tập hợp toàn bộ các tài liệu và đưa chúng vào một loạt các thuật toán học máy khác nhau.

2.6.3 Mô hình TF-IDF:

Mô hình TF-IDF (Term Frequency-Inverse Document Frequency) là một phương pháp quan trọng trong xử lý ngôn ngữ tự nhiên và trích xuất thông tin từ văn bản. Mô hình này đo lường tầm quan trọng của một từ trong một văn bản dựa trên hai yếu tố chính: tần suất xuất hiện của từ trong văn bản (TF) và tần suất xuất hiện của từ trong toàn bộ tập hợp các văn bản (IDF).

Trong mô hình TF-IDF, giá trị TF đại diện cho tần suất xuất hiện của một từ trong một văn bản cụ thể. Tuy nhiên, để tránh việc các từ phổ biến như "the", "a" có trọng số cao, chúng ta sử dụng giá trị IDF để điều chỉnh giá trị này bằng cách đo lường tần suất xuất hiện của từ đó trong toàn bộ tập hợp các văn bản. Từ đó, một từ có giá trị IDF cao nếu nó xuất hiện ít trong toàn bộ tập hợp các văn bản, làm tăng tầm quan trọng của từ đó khi xuất hiện trong một văn bản cụ thể.

2.7 Recommender System

2.7.1 Khái niệm

Recommender System (Hệ thống gợi ý) là một trong những ứng dụng phổ biến nhất của khoa học dữ liệu ngày nay. Recommender System là các hệ thống thông tin hoặc công nghệ phần mềm giúp dự đoán và gợi ý các mục (sản phẩm, dịch vụ, nội dung) mà một người dùng có thể quan tâm, dựa trên lịch sử tương tác hoặc các thông tin về họ. Các hệ thống này đóng vai trò quan trọng trong việc cải thiện trải nghiệm người dùng trên nhiều nền tảng, từ các trang web mua sắm đến các dịch vụ phát trực tuyến và các ứng dụng giải trí.

Recommender System là 1 nhánh con của hệ thống lọc thông tin (Information filtering system). Chúng chủ yếu được dùng trong các ứng dụng thương mại điện tử.

Thực chất, vấn đề của hệ gợi ý là xác định ánh xạ $(u, i) \rightarrow R$, trong đó u là biểu diễn cho 1 người dùng, i biểu diễn cho 1 sản phẩm và R là đánh giá của u lên i . Sau đó, sắp xếp

các đánh giá của người dùng u tương ứng với tất cả các sản phẩm i , và lấy N sản phẩm có đánh giá cao nhất để đưa ra gợi ý cho người dùng u .

2.7.2 Phân loại

Thông thường, hệ thống Recommender System có thể được chia thành 4 loại chính:

2.7.2.1 Simple Recommenders

Hệ thống tiến hành đưa ra các đề xuất tổng quát cho mọi người dùng, dựa trên mức độ phổ biến và/hoặc thể loại phim. Ví dụ: IMDB Top 250.

2.7.2.2 Lọc dựa trên nội dung (Content-based filtering)

Phương pháp lọc dựa trên nội dung dựa vào tổng số các item (đối với user) và profile về đánh giá của người dùng. Nó phù hợp nhất với dữ liệu đã biết về item và cách người dùng đã tương tác trước đây với hệ gợi ý, nhưng thiếu thông tin người dùng.

Thực chất, việc dự đoán đánh giá của user u lên item i chính là dựa trên các đánh giá của người dùng đó cho các item khác trước đây.

Hạn chế: Dữ liệu thừa thớt, dựa trên bộ nhớ hay các mô hình thì cả 2 đều tận dụng tương tác lịch sử của người dùng với hệ gợi ý. Vì vậy, đối với những người mới bắt đầu sử dụng hệ thống (Cold Start) thì hệ gợi ý có thể hoạt động chưa chính xác.

2.7.2.3 Lọc cộng tác (Collaborative filtering)

Lọc cộng tác phù hợp với các kiểu dữ liệu đã biết về người dùng nhưng thiếu dữ liệu cho các item hoặc khó thực hiện việc trích xuất tính năng cho các item quan tâm.

Không giống như lọc dựa trên nội dung, lọc cộng tác cố gắng dự đoán đánh giá của người dùng u cho 1 item i dựa trên các đánh giá của người dùng khác đối với item đó.

Hạn chế: Dữ liệu thừa thớt, đối với các item ít phổ biến hay ít đánh giá hơn, lọc cộng tác khó đưa ra kết quả chính xác.

2.7.2.4 Phương pháp lai ghép (Hybrid Method)

Do cả 2 phương pháp trên đều có các hạn chế riêng, đồng thời phương pháp này có thể giải quyết được hạn chế của phương pháp còn lại, vì vậy có thể lai ghép chúng với nhau.

Các cách kết hợp:

- Triển khai 2 phương pháp 1 cách riêng biệt và kết hợp các dự đoán của chúng. Đây thực chất là 1 mô hình tổng hợp.
- Kết hợp các đặc điểm dựa trên nội dung và lọc cộng tác. Cách làm là tận dụng profile của người dùng để đo độ tương đồng giữa 2 người dùng, và sử dụng độ tương đồng này làm trọng số trong bước tổng hợp 2 phương pháp.
- Một mô hình nhiều người dùng và nhiều mục. Điều này là để xây dựng một mô hình có cả tính năng mục và tính năng người dùng làm đầu vào, chẳng hạn như mô hình hồi quy tuyến tính, mô hình cây, mô hình mạng nơ ron, ...

2.8 Framework Flask

Flask là loại framework web phổ biến được viết bằng ngôn ngữ lập trình Python. Công nghệ thường được sử dụng để xây dựng trang web từ những ứng dụng đơn giản đến những hệ thống phức tạp hơn. Flask được tạo ra bởi Armin Ronacher và được phát triển dựa trên Werkzeug và Jinja2. Flask được thiết kế để đơn giản, dễ sử dụng và tùy biến, giúp các nhà phát triển xây dựng các ứng dụng web nhanh chóng và dễ dàng.

Flask cung cấp nhiều tính năng quan trọng để phát triển các ứng dụng web hiệu quả:

- *Nhẹ và dễ sử dụng*: Flask được thiết kế nhẹ nhàng và mã nguồn dễ đọc, giúp người phát triển dễ dàng tiếp cận và tùy chỉnh theo nhu cầu cụ thể của họ.
- *Định tuyến linh hoạt*: Flask cung cấp cơ chế hoạt động định tuyến (routing mechanism), cho phép người phát triển xác định các mẫu URL và phân bổ chúng cho các hàm xử lý tương ứng. Điều này giúp quản lý và xử lý yêu cầu HTTP một cách hiệu quả.
- *Công cụ mẫu*: Flask tích hợp Jinja2, một công cụ biên dịch mẫu mạnh mẽ cho phép tạo ra các giao diện người dùng linh hoạt và dễ dàng.
- *Được mở rộng rộng rãi*: Mặc dù là một framework nhỏ gọn, nhưng Flask vẫn có khả năng mở rộng mạnh mẽ thông qua việc sử dụng các tiện ích và thư viện của cộng đồng. Người dùng có thể tích hợp các tính năng như xác thực, đăng nhập, điều hướng, tương tác với cơ sở dữ liệu và nhiều tính năng khác.
- *Máy chủ phát triển tích hợp*: Flask cung cấp máy chủ phát triển tích hợp, giúp người phát triển dễ dàng kiểm tra và phát triển ứng dụng mà không cần cấu hình bổ sung.
- *Gửi yêu cầu RESTful*: Flask hỗ trợ xây dựng API và các ứng dụng RESTful theo cách hoạt động và hiệu quả.
- *Cộng đồng lớn và tích cực*: Flask có một cộng đồng người dùng đông đảo và tích cực, nhận được sự hỗ trợ mạnh mẽ từ cộng đồng Python. Điều này giúp người phát triển tìm kiếm thông tin và tài liệu một cách dễ dàng.

CHƯƠNG 3: PHƯƠNG PHÁP THU THẬP DỮ LIỆU VÀ TỔNG QUAN BỘ DỮ LIỆU

3.1. Sơ lược bộ dữ liệu

Bộ dữ liệu **Personality-Youtube_data.xlsx** là tổng hợp từ các tập dữ liệu nhỏ hơn liên quan tới các chủ đề trên nền tảng Youtube, những tập nhỏ hơn này được nhóm thực hiện cào dữ liệu trực tiếp từ 26/02/2024 đến 29/02/2024. Vào thời điểm nhóm thực hiện trích xuất dữ liệu và tổng hợp, bộ dữ liệu tổng hợp đã bao gồm 139012 quan sát với số lượng chủ đề là 12 mục.

Với mục tiêu phân tích khám phá kết hợp xây dựng mô hình máy học xoay quanh về nền tảng chia sẻ video hàng đầu thế giới này, nhóm đã thực hiện cào dữ liệu tổng cộng 14 biến thuộc tính. Chi tiết ý nghĩa các thuộc tính và phương pháp trích xuất dữ liệu sẽ được nhóm diễn giải cụ thể ở các phần tiếp theo.

3.2 Phương pháp thu thập dữ liệu

Ban đầu, nhóm xem xét tham khảo các trang web thứ ba như Apify hoặc Trackmyhashtag để thực hiện cào dữ liệu, cuối cùng, nhóm quyết định chọn phương pháp cào dữ liệu bằng python và sử dụng API cấp phép giới hạn từ Google Cloud console.

3.2.1 Youtube Data API

YouTube Data API là một dịch vụ web cung cấp bởi Google, cho phép tương tác với dữ liệu trên YouTube. Nó cho phép chúng ta truy cập vào nhiều loại thông tin từ YouTube, bao gồm video, danh sách phát, kênh, v.v...

Để sử dụng YouTube Data API, cần một API key, API này có thể lấy được từ Google Cloud Console. Cần có 1 tài khoản Google Cloud Console, sau đó khởi tạo project mới và gửi yêu cầu 1 API key cho project.

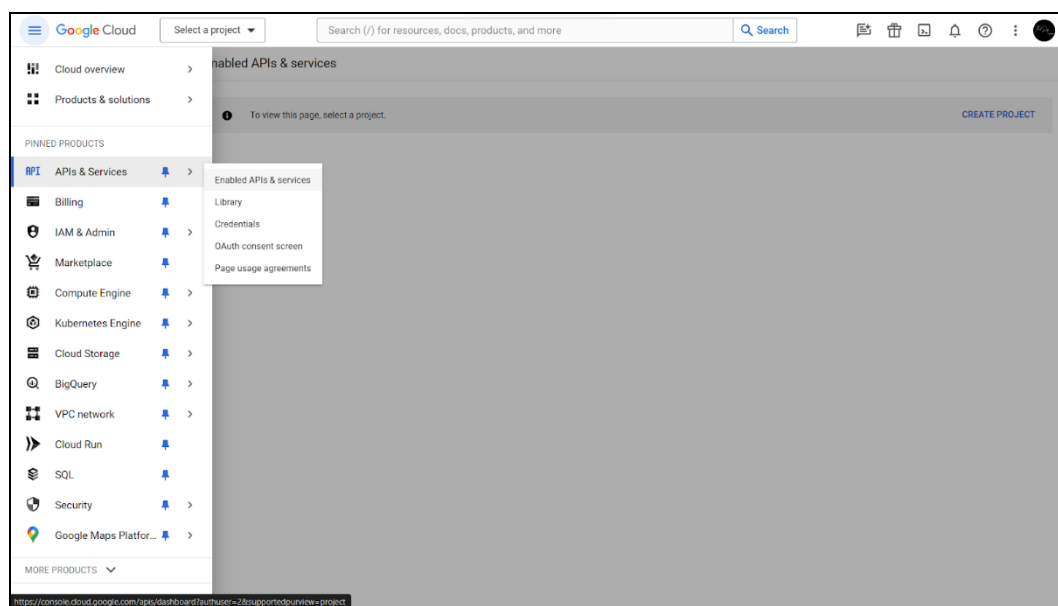
YouTube Data API có một hạn ngạch (quota) nhất định. Tất cả các yêu cầu API, kể cả các yêu cầu không hợp lệ, đều được tính tiêu hao một đơn vị quota. Chúng ta có thể kiểm tra quota có sẵn cho project cũng như mức tiêu thụ của mình trong API Console. Các dự án kích hoạt API của YouTube có hạn mức phân bổ mặc định là 10.000 đơn vị mỗi ngày. Hạn ngạch mặc định, có thể thay đổi, giúp nhà cung cấp dịch vụ tối ưu hóa việc phân bổ hạn ngạch và mở rộng cơ sở hạ tầng theo cách có ý nghĩa hơn đối với người dùng API của họ.

3.2.2 Thu thập dữ liệu

Ban đầu, nhóm xem xét tham khảo các trang web thứ ba như *Apify* hoặc *Trackmyhashtag* để thực hiện cào dữ liệu, cuối cùng, nhóm quyết định chọn phương pháp cào dữ liệu bằng *python* và sử dụng *API* cấp phép giới hạn từ *Google Cloud console*.

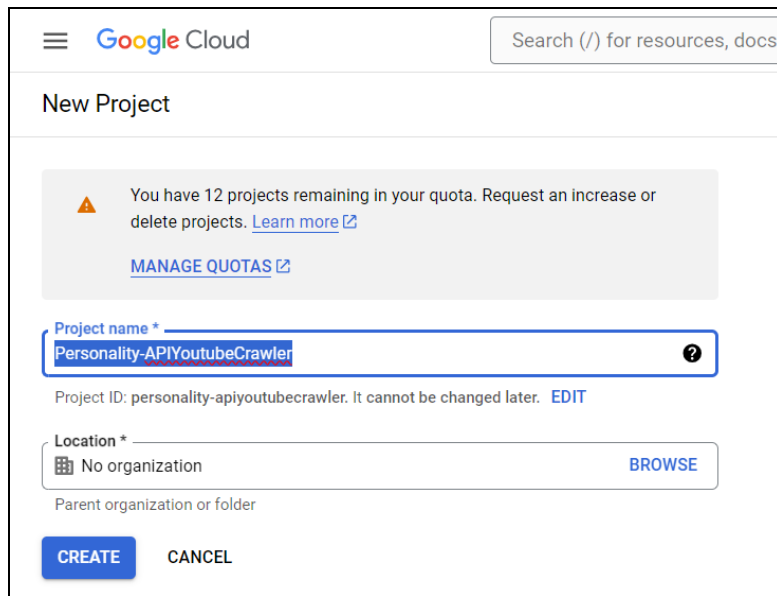
Ý tưởng thực hiện sẽ là sử dụng nền tảng này **tạo một token API** và cấp các quyền cơ bản hỗ trợ công việc cào dữ liệu, sau đó, nhóm sẽ viết một đoạn mã python để thực hiện tìm kiếm các video trên Youtube theo một danh sách từ khóa được xác định sẵn, tiếp đó, nhóm **sử dụng mã API đã tạo để thực hiện truy xuất các thông tin liên quan** về các video tìm được và lưu vào dataframe.

Vậy dựa trên hướng đi đó, đầu tiên, nhóm sẽ truy cập vào trang console của Google Cloud và sử dụng tài khoản Google bất kỳ để đăng nhập vào Product “APIs & Services” của hệ thống.



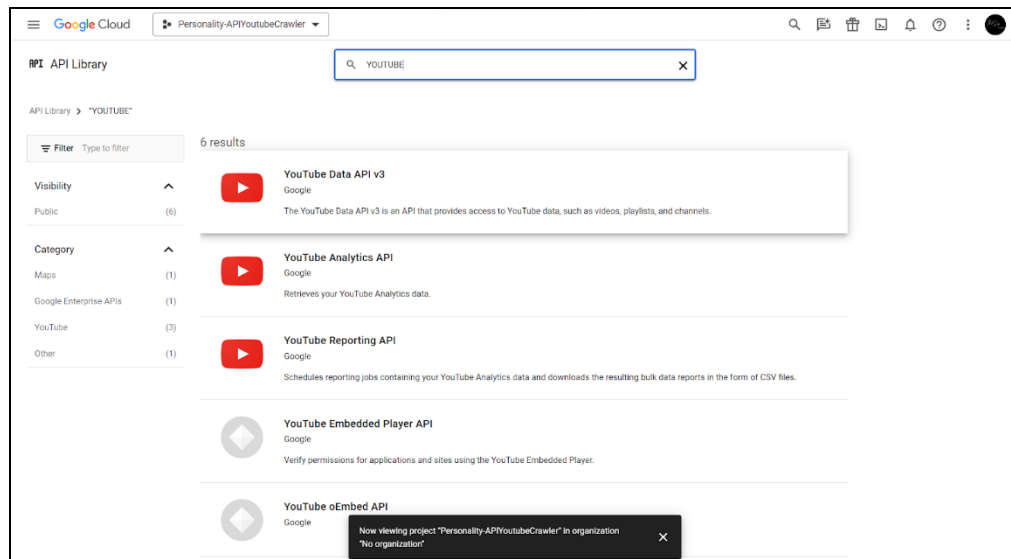
Hình 5. Giao diện tổng quan của Google Cloud console sau khi truy cập vào “APIs & Services”

Tiếp theo, nhóm sẽ tạo một project bất kỳ bằng cách chọn “CREATE PROJECT” trong giao diện tổng quan này và chỉnh sửa các tiện ích tùy thích, sau đó ấn “CREATE” để thực hiện tạo sau khi hoàn tất các tùy chỉnh.



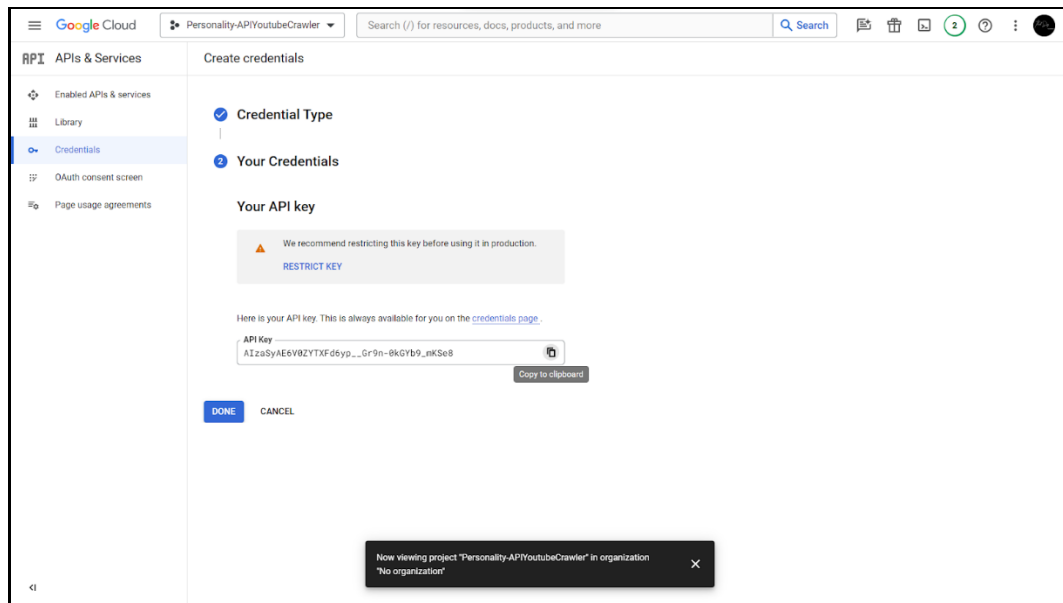
Hình 6. Giao diện tạo project của Google Cloud console

Sau khi khởi tạo thành công, nhóm truy cập vào tab “Library” ở thanh bên trái để tiến hành cấp một số quyền cho project này. Ở thanh tìm kiếm của giao diện này, nhóm **tiến hành tìm từ khóa “YOUTUBE”** để tìm **Library “YouTube Data API v3”** - đây là một thư viện cho phép các nhà phát triển truy cập và tương tác với dữ liệu thống kê của YouTube như thông tin về video, kênh, lượt tương tác v.v...



Hình 7. Sử dụng thanh tìm kiếm cho thư viện “YouTube Data API v3”

Sau khi truy cập vào trang thư viện này, tiến hành chọn nút “Enable” để thực hiện quá trình kích hoạt. Tiếp theo, nhóm sẽ tạo một key cho project để sử dụng API này bằng cách nhấn vào nút “CREATE CREDENTIALS” với tùy chọn *Credential Type* là “Public data”. Vậy là quá trình khởi tạo API đã hoàn tất và nhóm đã có thể sử dụng key này để thực hiện cào dữ liệu bằng mã *python*.



Hình 8. API key cuối cùng để sử dụng cào dữ liệu.

Quá trình tiếp theo này, nhóm sẽ sử dụng ngôn ngữ python để viết hàm cào dữ liệu các thông tin cần thiết từ các thuộc tính javascript của nền tảng tham khảo từ trang web “YouTube Player API Reference for iframe Embeds”.

```

1 import requests
2 import pandas as pd
3
4 def search_videos(search_terms, api_key, max_results):
5     search_url = "https://www.googleapis.com/youtube/v3/search"
6     data = []
7     total_results = 0
8
9     for search_term in search_terms:
10         page_token = None
11
12         while total_results < max_results:
13             # try:
14             search_params = {
15                 "part": "snippet",
16                 "q": search_term,
17                 "key": api_key,
18                 "maxResults": min(5000 - total_results, 50),
19                 "type": "video",
20                 "pageToken": page_token
21             }
22
23             r = requests.get(search_url, params=search_params)
24             results = r.json()
25
26             if 'error' in results:
27                 print("Đã vượt quá hạn mức API. Dừng chương trình.")
28                 break
29
30             for result in results["items"]:
31                 video_id = result["id"]["videoId"]
32                 video_info_url = f"https://www.googleapis.com/youtube/v3/videos?id={video_id}&part=snippet,contentDetails,statistics,status&key={api_key}"
33                 video_info_response = requests.get(video_info_url)
34                 video_info = video_info_response.json()
35
36                 channel_title = result["snippet"]["channelTitle"]
37                 video_title = result["snippet"]["title"]
38                 video_description = result["snippet"]["description"]
39                 video_url = "https://www.youtube.com/watch?v=" + video_id
40                 subscribers_count = video_info["items"][0]["statistics"].get("subscriberCount", 0)
41                 channel_url = "https://www.youtube.com/channel/" + video_info["items"][0]["snippet"]["channelId"]
42                 monetized = video_info["items"][0]["status"].get("monetizationStatus", "Not monetized")
43                 comments_count = video_info["items"][0]["statistics"].get("commentCount", 0)
44                 comments_turned_off = video_info["items"][0]["status"].get("comments", "Comments turned on")
45                 publish_date = video_info["items"][0]["snippet"]["publishedAt"]
46                 duration = video_info["items"][0]["contentDetails"]["duration"]
47                 view_count = video_info["items"][0]["statistics"].get("viewCount", 0)
48                 like_count = video_info["items"][0]["statistics"].get("likeCount", 0)
49                 data.append([search_term, channel_title, subscribers_count, channel_url, video_id, monetized, comments_count,
50                             comments_turned_off, publish_date, duration, view_count, like_count, video_title, video_description])
51                 total_results += 1
52                 page_token = results.get("nextPageToken")
53             if not page_token:
54                 print("Đã vượt quá hạn mức API.")
55                 break
56             # except Exception as e:
57             #     print(f"An error occurred: {e}")
58             #     break
59         if total_results >= 5000:
60             break
61 df = pd.DataFrame(data, columns=["Topic", "Channel", "Subscribers", "Channel URL", "Video ID", "Monetized", "Comments Count",
62                                "Comments Turned Off", "Publish Date", "Duration", "View Count", "Like Count", "Title", "Description"])
63 return df

```

Hình 9. Hàm python thực hiện cào dữ liệu

Trong đoạn mã trên, hàm `search_videos` được xây dựng để tìm kiếm video trên YouTube dựa trên một danh sách các từ khóa tìm kiếm (`search_terms`), một API key (`api_key`), và số lượng kết quả tối đa (`max_results`). Hàm này sử dụng YouTube Data API nêu trên để thực hiện các truy vấn tìm kiếm.

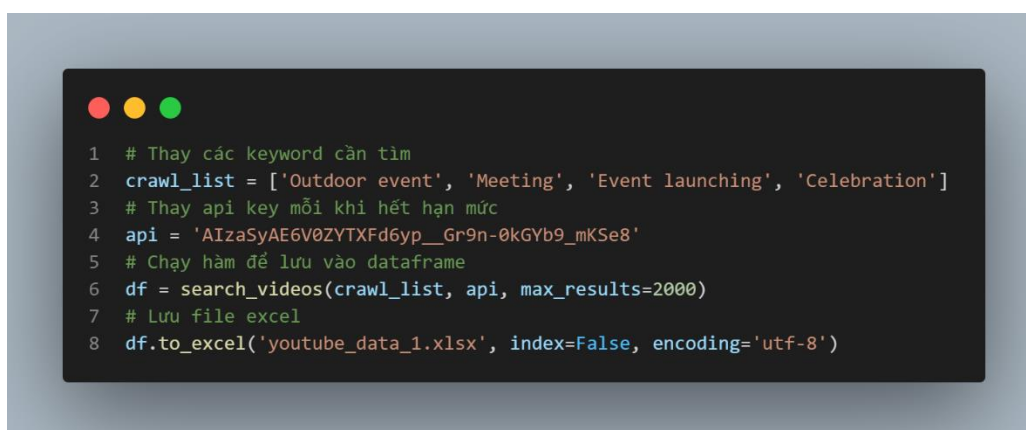
Đầu tiên là khởi tạo các biến, bao gồm URL cho API tìm kiếm YouTube (`search_url`), một danh sách để lưu trữ dữ liệu (`data`), và hai biến để xác định số lượng kết quả tối đa cho mỗi từ khóa tìm kiếm (`max_results_per_term` và `remainder`). Chương trình tính toán số lượng kết quả tối đa cho mỗi từ khóa tìm kiếm, cùng với số lượng kết quả còn lại sau khi chia đều cho tất cả các từ khóa.

Sau đó duyệt qua từng từ khóa tìm kiếm trong danh sách `search_terms`. Đối với mỗi từ khóa, thực hiện một loạt các truy vấn tìm kiếm đến YouTube API cho đến khi đạt được số lượng kết quả tối đa cho từ khóa đó (`term_max_results`). Đối với mỗi truy vấn, chương trình sẽ gửi một yêu cầu GET đến `search_url` với một số tham số tìm

kiếm, bao gồm từ khóa tìm kiếm hiện tại, API key, số lượng kết quả tối đa cho mỗi truy vấn (`maxResults`), loại kết quả mong muốn (`type`), và token của trang kết quả tiếp theo (`pageToken`).

Kết quả trả về từ mỗi truy vấn tìm kiếm là một đối tượng JSON chứa thông tin về các video tìm thấy. Sau đó hàm sẽ duyệt qua từng video trong kết quả, thu thập thông tin chi tiết về mỗi video bao gồm: tiêu đề kênh, tiêu đề video, mô tả video, URL video, số lượng người đăng ký, URL kênh, trạng thái kiểm tiền, số lượng bình luận, trạng thái bình luận, ngày xuất bản, thời lượng, số lượng lượt xem, và số lượng lượt thích. Sau đó các thêm thông tin này vào danh sách `data`. Danh sách này gồm các cột chứa các thông tin vừa khai thác được và cột chủ đề (`Topic`) của video (tương ứng với `search_term`).

Cuối cùng, tạo một DataFrame pandas từ danh sách `data`, với mỗi cột trong DataFrame tương ứng với một loại thông tin khác nhau về các video.



```
1 # Thay các keyword cần tìm
2 crawl_list = ['Outdoor event', 'Meeting', 'Event launching', 'Celebration']
3 # Thay api key mỗi khi hết hạn mức
4 api = 'AIzaSyAE6V0ZYTXFd6yp__Gr9n-0kGYb9_mKSe8'
5 # Chạy hàm để lưu vào dataframe
6 df = search_videos(crawl_list, api, max_results=2000)
7 # Lưu file excel
8 df.to_excel('youtube_data_1.xlsx', index=False, encoding='utf-8')
```

Hình 10. Tùy chỉnh các tham số và thực hiện chạy hàm

Gọi hàm `search_videos` và thay các tham số vào để cào các video theo chủ đề với 3 tham số truyền vào gồm các từ khóa tìm kiếm, API key được Google cung cấp và số lượng video tối đa muốn cào được.

Sau đó lưu DataFrame được trả về từ hàm `search_videos` vào file excel.

Vậy cuối cùng qua hai công đoạn chính là tạo API key và chạy hàm cào dữ liệu, nhóm đã thu được một tập dữ liệu con bao gồm 4 từ khóa với số lượng quan sát xấp xỉ trên dưới 2000 dòng. Trên cơ sở đó, nhóm quyết định chọn mốc 10000 quan sát cho một chủ đề để tập dữ liệu cha được đa dạng hóa về số lượng. Kết quả cuối cùng từ việc cào được thể hiện bằng bảng sau đây.

BỘ DỮ LIỆU	SỐ QUAN SÁT	MÔ TẢ
Animals	10157	Bao gồm thông tin về các loài động vật nuôi, thú hoang dã trong môi trường tự nhiên
Education	11785	Bao gồm các video liên quan tới chủ đề về giáo dục
Film & Animation	10190	Bao gồm các video về lĩnh vực phim ảnh như các bộ phim công chiếu nổi tiếng, hoạt hình...
Food	10042	Bao gồm các danh mục chủ đề về đồ ăn, đặc sản ẩm thực...
Gaming	14732	Bao gồm các danh mục xoay quanh thế giới trò chơi điện tử và các giải đấu điện tử chuyên nghiệp
History & Documentary	10346	Bao gồm các thông tin xoay quanh chủ đề trong quá khứ như lịch sử, tài liệu cũ...
Music	9913	Bao gồm các video về lĩnh vực âm nhạc như video âm nhạc, các bài hát hay nghệ sĩ nổi tiếng...
News & Politics	16015	Bao gồm các thông tin xoay quanh lĩnh vực chính trị, là những vấn đề mới nổi xuất hiện trên các bản tin
NonProfit & Activism	12014	Bao gồm những video về các hoạt động thiện nguyện xã hội hay các dự án tổ chức phi lợi nhuận
Science & Technology	10514	Bao gồm các danh mục chủ đề về công nghệ và khoa học hay các lĩnh vực không ngừng đổi mới liên quan
Sports	10174	Bao gồm các video về khoảnh khắc đặc biệt trong thể thao hay các giải vô địch cùng lĩnh vực
Travel & Events	13129	Bao gồm thông tin về các video du lịch, giới thiệu địa điểm mới, các sự kiện trong đời sống

Bảng 1. Kết quả cào các tập dữ liệu con bằng API từ nền tảng Youtube.

3.2.3 Mô tả thuộc tính bộ dữ liệu:

TÊN THUỘC TÍNH	KIỂU DỮ LIỆU	MÔ TẢ	CHÚ THÍCH
Topic	Chuỗi	Chủ đề chính của video	
Channel	Chuỗi	Kênh đăng tải video	
Subscribers	Số nguyên	Số lượng người đăng ký của của kênh này	
Channel URL	Chuỗi	Đường dẫn liên kết tới trang web của kênh này	
Video ID	Chuỗi	Mã video	Đây là mã định danh cho từng video, có thể nhìn thấy ở cuối đường dẫn của video đó
Monetized	Chuỗi	Phân loại chế độ kiếm tiền của video	Gồm 2 giá trị: - “Not monetized” : Video không bật chế độ kiếm tiền - “Monetized” : Video bật chế độ kiếm tiền
Comments Count	Số nguyên	Số lượng bình luận của video	
Comments Turned Off	Chuỗi	Phân loại chế độ ẩn bình luận của video	Gồm 2 giá trị: - “Comments turned on” : Video bật chế độ bình luận - “Comments turned off” : Video tắt chế độ bình luận

Publish Date	Chuỗi	Thời điểm video được phát hành công khai	Gồm ngày và thời gian phát hành
Duration	Chuỗi	Thời lượng video	
View Count	Số nguyên	Số lượng lượt xem của video	
Like Count	Số nguyên	Số lượng lượt thích của video	
Title	Chuỗi	Tiêu đề video	
Description	Chuỗi	Mô tả video	

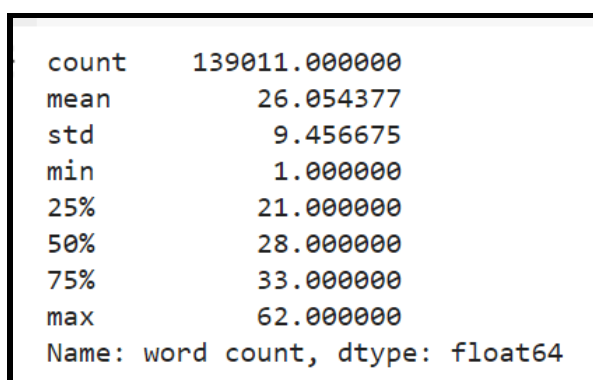
Bảng 2. Data dictionary (mô tả biến thuộc tính)

CHƯƠNG 4: TIỀN XỬ LÝ, KHÁM PHÁ VÀ PHÂN TÍCH DỮ LIỆU

4.1 Phân tích khám phá dữ liệu (Exploratory Data Analysis – EDA)

Nhóm tiến hành khám phá dữ liệu để hiểu và phân tích tính chất của dữ liệu một cách hệ thống trước khi thực hiện các bước xử lý tiếp theo.

Đầu tiên, nhóm áp dụng hàm “word_count” cho cột “Title_Description” trong DataFrame “data” để tạo ra một cột mới có tên là “word count”, lưu trữ số lượng từ trong mỗi mẫu. Sau đó, tóm tắt thống kê về cột 'word count', bao gồm các thống kê như số lượng mẫu, trung bình, độ lệch chuẩn, giá trị tối thiểu, phân vị 25%, 50%, 75% và giá trị tối đa.



count	139011.000000
mean	26.054377
std	9.456675
min	1.000000
25%	21.000000
50%	28.000000
75%	33.000000
max	62.000000
Name: word count, dtype: float64	

Hình 11. Thống kê mô tả cột 'word count'

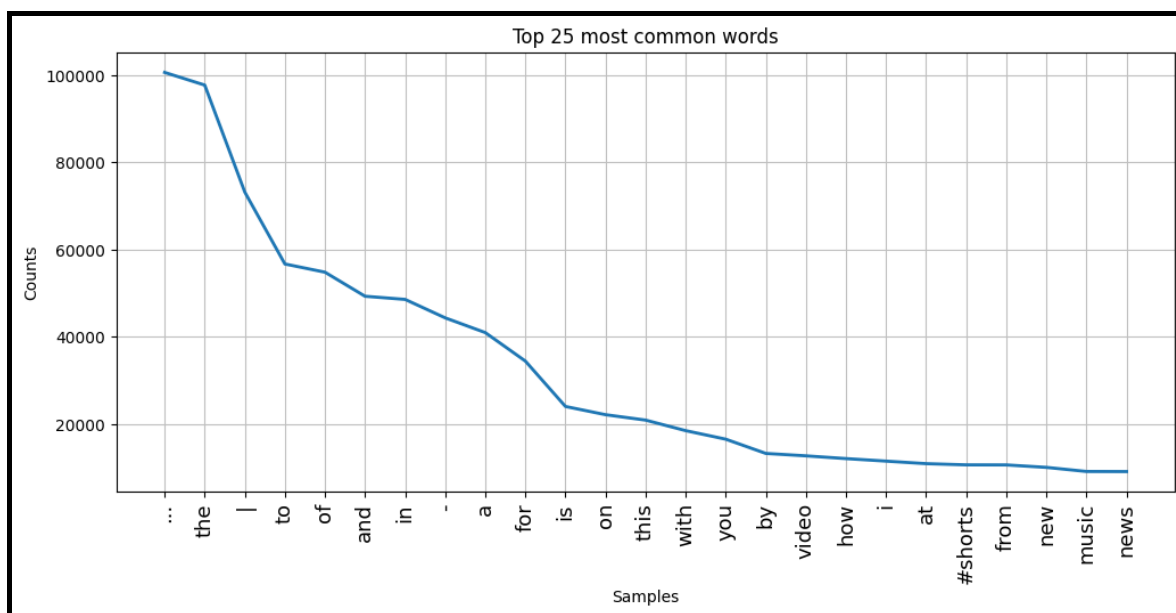
Giải thích các giá trị:

- Count: Có tổng cộng 139,011 mẫu dữ liệu được quan sát trong cột dữ liệu.
- Mean: Trung bình số lượng từ trong mỗi mẫu là khoảng 26.05 từ. Điều này có nghĩa là trung bình mỗi mẫu dữ liệu chứa khoảng 26 từ.
- Std (standard deviation): Độ lệch chuẩn là khoảng 9.46 từ, cho thấy rằng sự biến động trong số lượng từ giữa các mẫu khá lớn.
- Min, Max: Có ít nhất một từ và nhiều nhất 62 từ. Điều này chỉ ra rằng có sự đa dạng lớn về số lượng từ trong các mẫu dữ liệu.
- Phân vị (percentiles):
 - 25% (Q1): 21 từ. Khoảng 25% của các mẫu có số lượng từ ít hơn hoặc bằng 21 từ.

- 50% (Q2) hay trung vị: 28 từ. Khoảng 50% của các mẫu có số lượng từ ít hơn hoặc bằng 28 từ.
- 75% (Q3): 33 từ. Khoảng 75% của các mẫu có số lượng từ ít hơn hoặc bằng 33 từ.

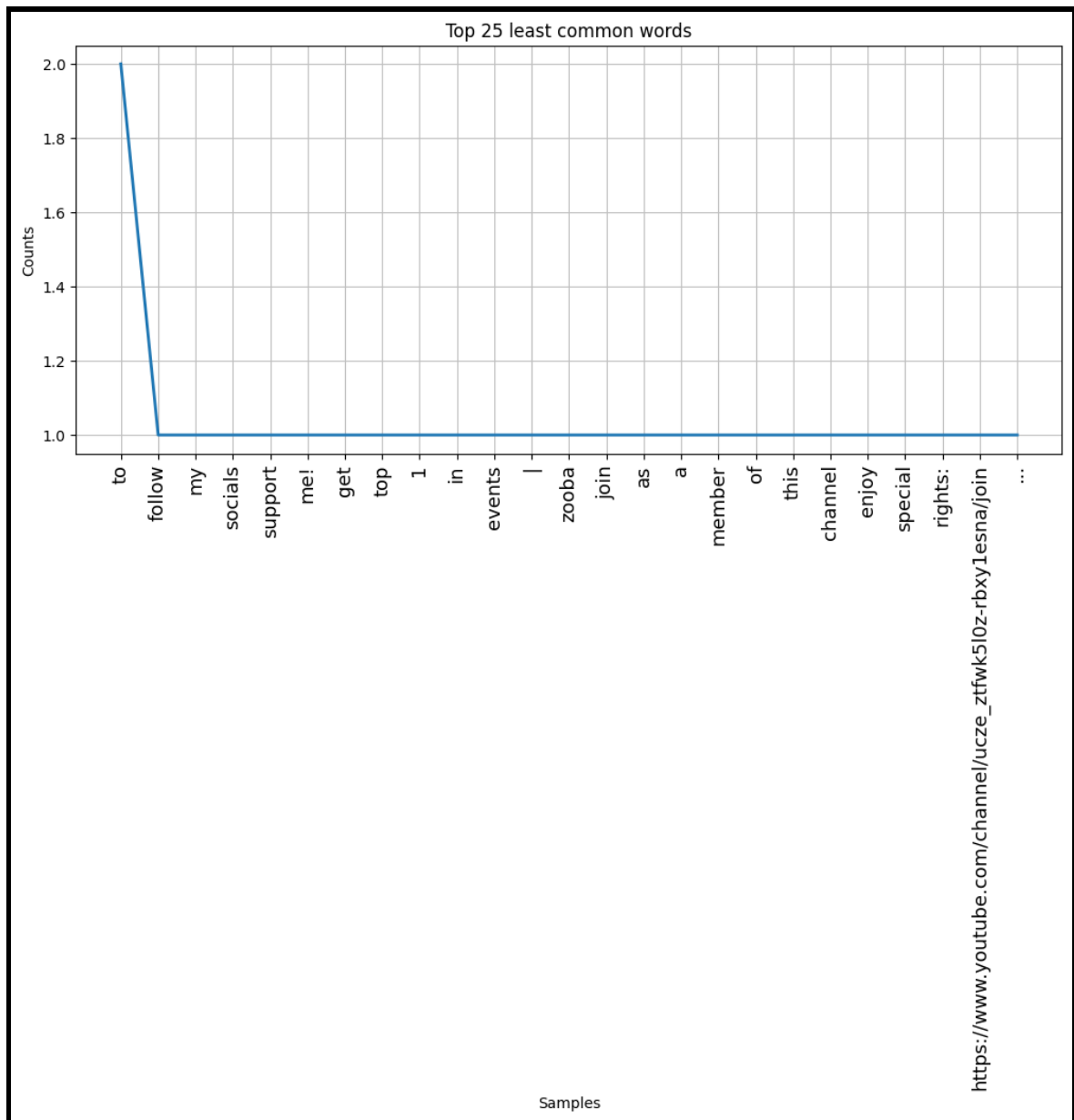
Thông tin này cung cấp một cái nhìn tổng quan về phân phối của số lượng từ trong các mẫu dữ liệu. Có sự biến động đáng kể trong số lượng từ giữa các mẫu.

Tiếp theo, nhóm tiến hành phân tích xem khi chưa qua các bước xử lý, những từ nào sẽ xuất hiện ít nhất và những từ nào sẽ xuất hiện nhiều nhất.



Hình 12. Top 25 từ xuất hiện nhiều nhất

Các từ phổ biến như '...', 'the', 'to', 'of', 'and', 'in', 'a', 'for', '|', và '-' xuất hiện nhiều nhất trong cột 'Title_Description'. Các từ này thường không cung cấp thông tin cụ thể về nội dung của các mẫu dữ liệu và có thể là các từ phổ biến trong ngôn ngữ hoặc ký tự đặc biệt được sử dụng cho mục đích định dạng.



Hình 13. Top 25 từ xuất hiện ít nhất

Những từ trong top 25 từ xuất hiện ít nhất, bao gồm cả các từ kết hợp giữa số và chữ, cũng như các link URL, thường là các từ không mang lại ý nghĩa cụ thể hoặc thông tin chính xác về nội dung của các mẫu dữ liệu. Các từ này có thể xuất hiện ít nhất vì chúng thường là các từ phụ, không mang lại sự diễn giải rõ ràng về nội dung hay không được xem là quan trọng trong ngữ cảnh của dữ liệu.

Các kết quả phân tích ban đầu cho thấy rằng bộ dữ liệu cần được xử lý cụ thể trước khi áp dụng vào xây dựng mô hình.

4.2 Tiền xử lý dữ liệu

Bước tiền xử lý này giúp làm sạch dữ liệu, tập trung vào thông tin quan trọng và cải thiện tính chính xác của các mô hình sau này. Nhóm thực hiện thông qua một số bước cụ thể.

Loại bỏ trùng lặp:

```
data.drop_duplicates(inplace=True)
```

Chuyển đổi các cột cần thiết sang dạng chuỗi:

```
df['Title'] = df['Title'].astype(str)
df['Description'] = df['Description'].astype(str)
```

Xử lý các giá trị NaN (null):

```
df['Description'].fillna('', inplace=True)
df['Description'].isnull().sum()
```

Các bước xử lý văn bản tiếp theo:

- Loại bỏ các thực thể HTML đặc biệt (ví dụ: &).
- Chuyển đổi tất cả các tên người dùng @username.
- Loại bỏ các từ khóa (tickers) như các ký hiệu cổ phiếu (ví dụ: \$AAPL).
- Chuyển đổi tất cả các ký tự trong chuỗi title thành chữ thường.
- Loại bỏ các liên kết web (URLs) từ chuỗi title.
- Loại bỏ các hashtag (ví dụ: #hashtag) từ chuỗi title.
- Loại bỏ các dấu câu và thay thế chúng bằng khoảng trắng trong chuỗi.
- Loại bỏ các từ có 2 hoặc ít hơn ký tự trong chuỗi.
- Loại bỏ khoảng trắng dư thừa trong chuỗi.
- Loại bỏ khoảng trắng ở đầu chuỗi.
- Loại bỏ các ký tự nằm ngoài Phạm vi BMP của Unicode từ chuỗi.

```
import string
def processTitle(title):
    # Remove HTML special entities (e.g. &)
    title = re.sub(r'&\w*;', '', title)
    # Convert @username to AT_USER
    title = re.sub('@[^\s]+', '', title)
    # Remove tickers
    title = re.sub(r'\$\w*', '', title)
    # To lowercase
    title = title.lower()
    # Remove hyperlinks
```

```

title = re.sub(r'https?:\/\/\.*\/\w*', '', title)
# Remove hashtags
title = re.sub(r'#\w*', '', title)
# Remove Punctuation and split 's, 't, 've with a space for filter
title = re.sub(r'['+string.punctuation+']+', ' ', title)
# Remove words with 2 or fewer letters
title = re.sub(r'\b\w{1,2}\b', '', title)
# Remove whitespace (including new line characters)
title = re.sub(r'\s\s+', ' ', title)
# Remove single space remaining at the front of the title.
title = title.lstrip(' ')
# Remove characters beyond Basic Multilingual Plane (BMP) of
Unicode:
title = ''.join(c for c in title if c <= '\uFFFF')
return title

```

Lemmatization và tokenization:

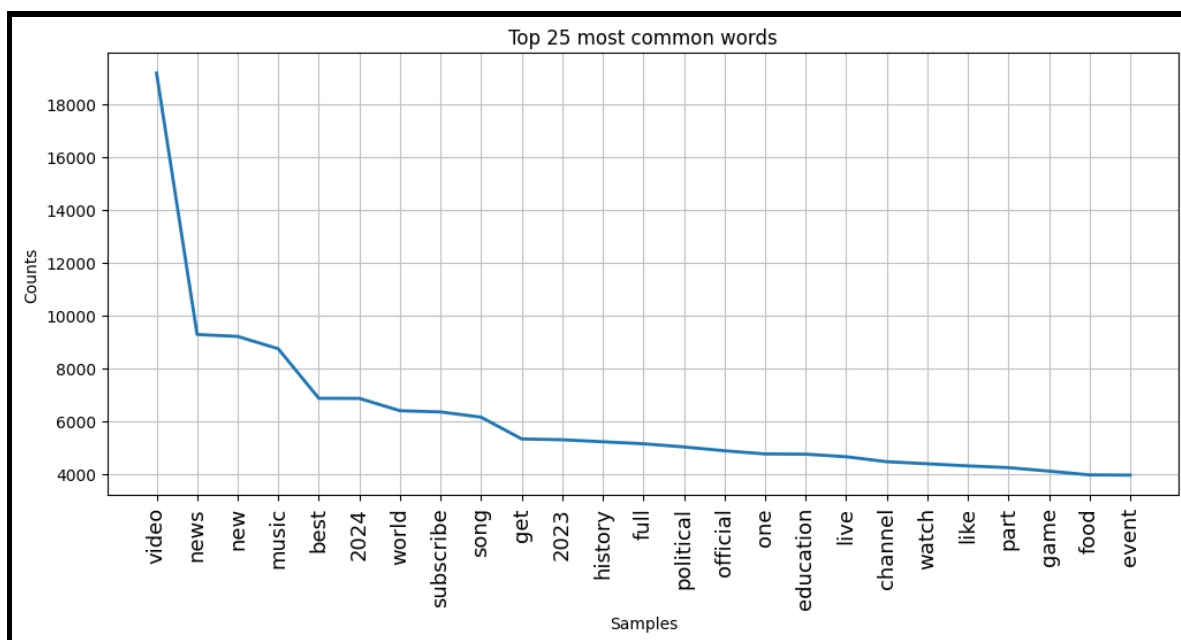
- Tokenization là quá trình chia văn bản thành các đơn vị nhỏ hơn gọi là "token", chẳng hạn như từ hoặc dấu câu. Tokenization giúp chia nhỏ văn bản thành các phần tử có ý nghĩa để tiện cho việc xử lý tiếp theo, như phân tích cú pháp hoặc xây dựng mô hình. Các thư viện phổ biến như NLTK (Natural Language Toolkit) hoặc spaCy cung cấp các công cụ để thực hiện tokenization một cách hiệu quả.
- Lemmatization: Lemmatization là quá trình chuẩn hóa các từ về dạng gốc (lemma) của chúng. Một từ có thể có nhiều dạng khác nhau (ví dụ: "running", "ran", "runs" đều liên quan đến "run"), nhưng lemmatization chuyển đổi tất cả các dạng này về dạng gốc ("run"). Điều này giúp giảm số lượng từ vựng trong dữ liệu và cải thiện tính nhất quán khi xử lý văn bản. Trong Python, thư viện spaCy hoặc NLTK cung cấp các công cụ để thực hiện lemmatization.

	Title_Description	View Count	Like Count	Topic	tokens	clean_title
0	Best Funny Animals Funniest Dogs and Cats Funn...	93719	210	Animals	[best, funny, animal, funniest, dog, cat, funn...	best funny animal funniest dog cat funniest an...
1	Animals Provided to YouTube by Roadrunner Reco...	26738388	240292	Animals	[animal, provided, youtube, roadrunner, record...	animal provided youtube roadrunner record anim...
2	Turf Angriest Animals Animal Fight Night Fins ...	41712	1047	Animals	[turf, angriest, animal, animal, fight, night,...	turf angriest animal animal fight night fin th...
3	Ocean Sea Animals for Beautiful Coral Reef Fis...	1081	396	Animals	[ocean, sea, animal, beautiful, coral, reef, f...	ocean sea animal beautiful coral reef fish aqu...
4	Wildlife The Fascinating World of Wild Animals...	1191645	5874	Animals	[wildlife, fascinating, world, wild, animal, f...	wildlife fascinating world wild animal full se...

Hình 14. Kết quả sau Lemmatization và Tokenization

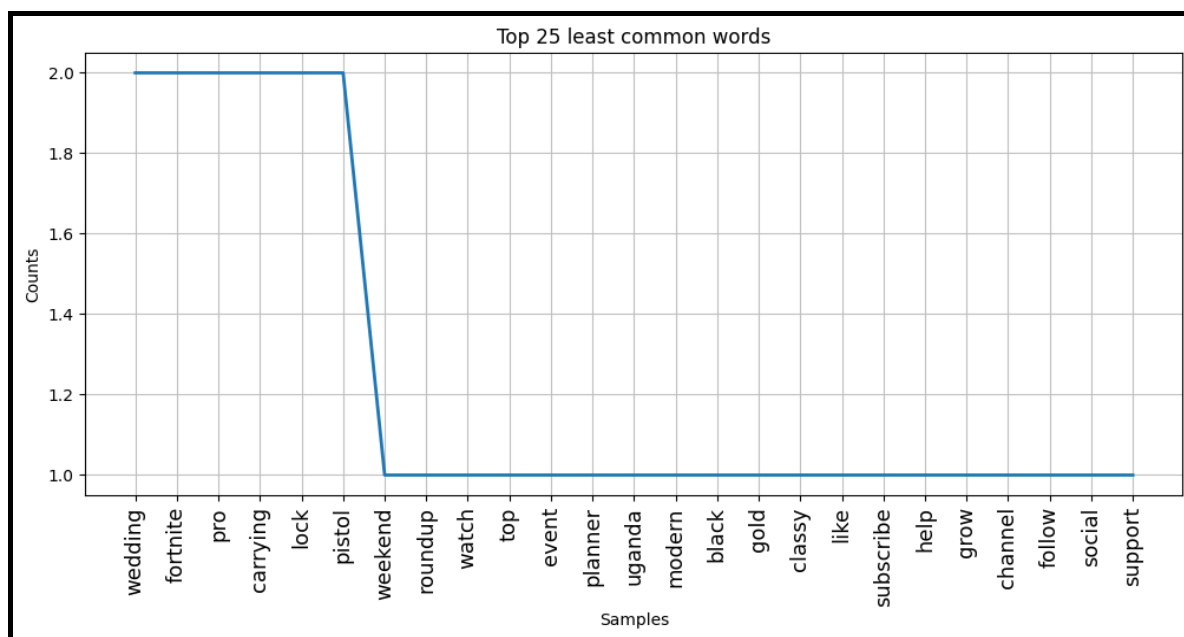
4.3 Trục quan hóa kết quả sau tiền xử lý

Với mục đích kiểm tra lại xem các bước xử lý đã đưa ra được kết quả tốt hơn chưa, nhóm tiến hành trục quan hóa thông qua một số biểu đồ trục quan.



Hình 15. Top 25 từ xuất hiện nhiều nhất (sau tiền xử lý)

Kết quả trả về khi trích xuất 25 từ xuất hiện nhiều nhất là từ vựng đúng dạng trong dữ liệu là một điểm tích cực, vì nó cho thấy các từ này có thể mang lại thông tin quan trọng về nội dung hoặc chủ đề chính của dữ liệu. Việc xem xét ngữ cảnh và sự xuất hiện của các từ này có thể giúp chúng ta hiểu rõ hơn về chủ đề, nội dung hoặc mục đích của dữ liệu. Từ đó, có thể xác định các phương pháp xử lý và phân tích phù hợp để khai thác thông tin từ dữ liệu một cách hiệu quả.



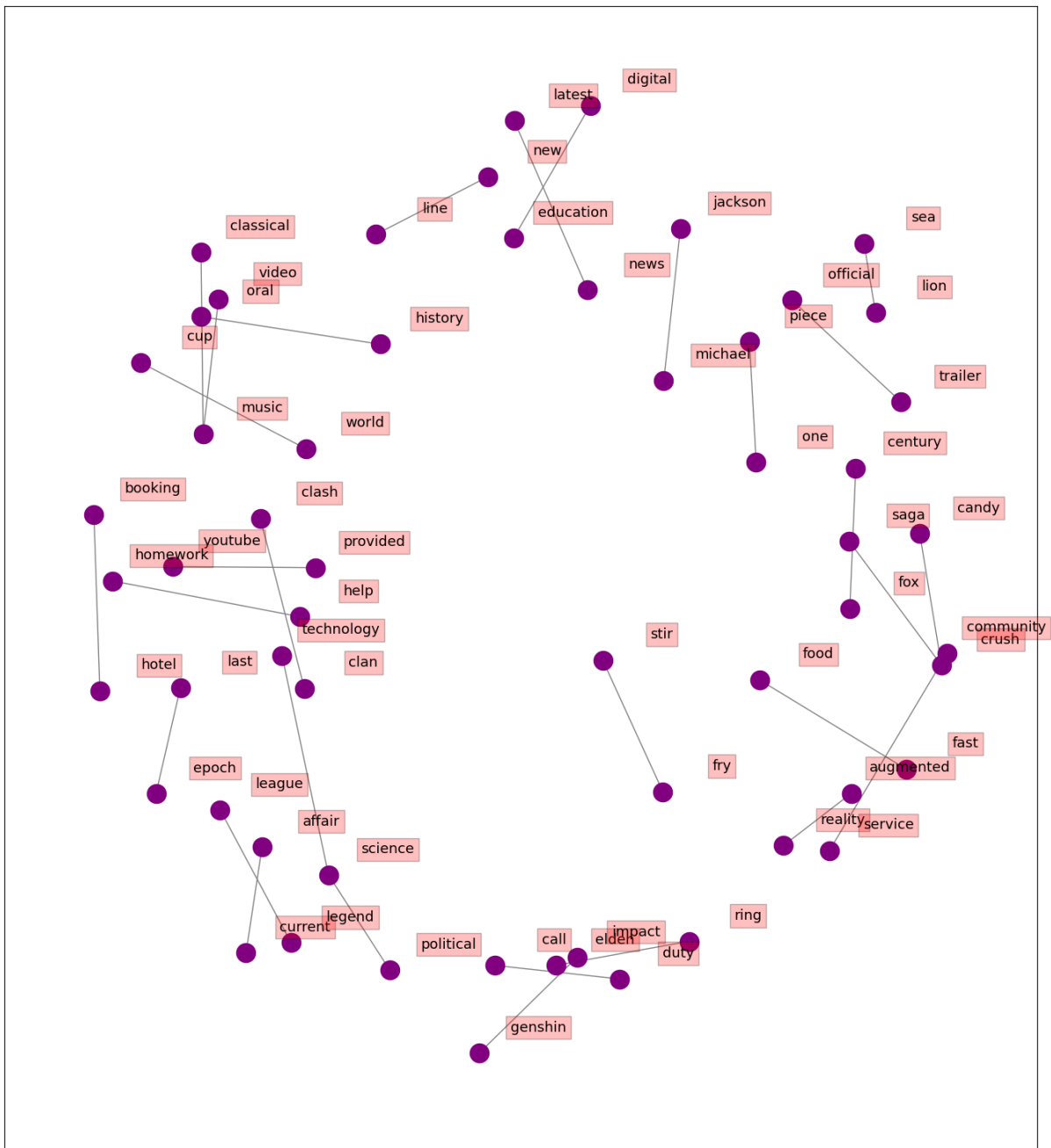
Hình 16. Top 25 từ xuất hiện ít nhất (sau tiền xử lý)

Khi trực quan hóa 25 từ xuất hiện ít nhất, việc trả về các từ đúng dạng thay vì các ký tự hay đường liên kết URL là kết quả tốt trong quá trình tiền xử lý dữ liệu. Điều này cho thấy rằng dữ liệu đã được làm sạch khỏi các phần không mong muốn và tập trung vào các từ mang ý nghĩa. Việc này cải thiện tính chính xác của phân tích và tạo điều kiện hiểu rõ hơn về nội dung và ngữ cảnh của dữ liệu.



Hình 17. Word Cloud kết quả cột 'Title Description'

Các từ như “video” và “subscribe” không mang lại nhiều giá trị cho việc phân tích vì chúng có thể chỉ đơn giản là các từ chung chung liên quan đến nền tảng youtube và hoạt động trên mạng. Tuy nhiên, các từ như “music”, “education”, “history”, “science” và các từ khác liên quan đến các chủ đề cụ thể thì lại vô cùng quan trọng và hữu ích khi phân loại các mẫu vào các chủ đề tương ứng.



Hình 18. Mô hình Bigram

Nhóm tiền hành trực quan kết quả thông qua mô hình Bigram vì đây là công cụ quan trọng trong xử lý ngôn ngữ tự nhiên, giúp phân tích và hiểu ngữ cảnh của từng từ trong văn bản bằng cách xem xét từ liền trước của nó. Bigram cũng được sử dụng để dự đoán từ tiếp theo trong một chuỗi văn bản và hiểu cấu trúc của ngôn ngữ tự nhiên.

CHƯƠNG 5: HUẤN LUYỆN DỮ LIỆU DỰA TRÊN CÁC MÔ HÌNH

Sau khi thực hiện các bước tiền xử lý văn bản và phân tích khám phá thông qua các bước trực quan hóa dữ liệu. Nhóm sẽ tiến hành sử dụng dữ liệu đã được làm sạch để tiến hành huấn luyện mô hình học máy và học sâu đã được đề cập ở chương 2.

```
sentences = data['clean_title']
labels = data['Topic']
tokens = data['tokens']
re_title = data['Title']
```

Nhóm tiến hành trích xuất dữ liệu từ Dataframe “data” thành các Series riêng biệt

- sentences: chứa dữ liệu của cột “clean_title” bao gồm title và description của các video đã được tiền xử lý.
- labels: chứa dữ liệu của cột “Topic” là nhãn của từng dòng trong sentences.
- tokens: chứa dữ liệu của cột “tokens” với từng dòng dữ liệu là list các token được tách từ cột clean_title.
- re_title: chứa dữ liệu của cột “Title” lưu thông tin của các title khi chưa xử lý.

Kế đến nhóm sẽ cài đặt những thư viện hỗ trợ cần thiết cho quá trình chuyển đổi văn bản sang dạng vector và huấn luyện mô hình.

5.1 Mô hình Naive Bayes

5.1.1 Chuyển đổi văn bản và tách tập dữ liệu huấn luyện

```
label_encoder = LabelEncoder()
encoded_labels = label_encoder.fit_transform(labels)

# Initialize CountVectorizer to convert text data into numerical
features
vectorizer = CountVectorizer()

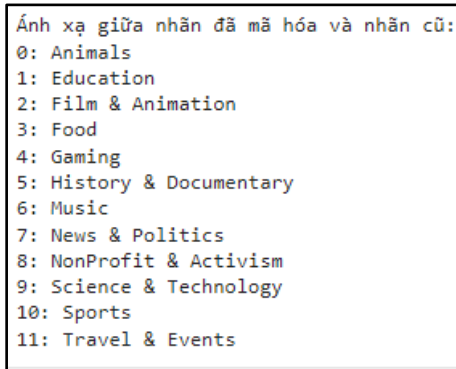
# Convert text data into numerical features
X_NB = vectorizer.fit_transform(sentences)

# Split the data into training and testing sets
X_train_NB, X_test_NB, y_train_NB, y_test_NB = train_test_split(X_NB,
encoded_labels, test_size=0.2, random_state=42)
```

Đầu tiên, nhóm sẽ thực hiện các bước xử lý dữ liệu cần thiết cho phù hợp với yêu cầu kỹ thuật của các mô hình được thiết kế trong các thư viện hỗ trợ. Đối với các bài toán

xử lý văn bản để huấn luyện mô hình phân lớp, cần phải thực hiện vector hóa văn bản để phù hợp với tính chất và công thức toán học của mô hình.

Đối với mô hình Naïve Bayes, nhóm sẽ thực hiện chuyển đổi văn bản thành các đặc trưng số học bằng phương pháp Bag of words bởi vì phương pháp này phù hợp với giả định Naive Bayes, mô hình Naive Bayes giả định rằng các đặc trưng độc lập với nhau khi đã biết lớp. Phương pháp BoW không cần biết về mối quan hệ giữa các từ và tập trung chỉ vào tần suất xuất hiện của chúng, phù hợp với giả định của Naive Bayes. Đồng thời dữ liệu nhãn gán cũng được số hóa bằng phương pháp Label Encoder.



```
Ánh xạ giữa nhãn đã mã hóa và nhãn cũ:
0: Animals
1: Education
2: Film & Animation
3: Food
4: Gaming
5: History & Documentary
6: Music
7: News & Politics
8: NonProfit & Activism
9: Science & Technology
10: Sports
11: Travel & Events
```

Hình 19. Ánh xạ giữa nhãn đã mã hóa và nhãn ban đầu

Kế đến, nhóm thực hiện chia dữ liệu huấn luyện thành các tập training và test với tỷ lệ 8:2. Cài đặt tham số `random_state` để giữ tính nhất quán cho mỗi lần chia tập dữ liệu.

5.1.2 Huấn luyện mô hình và đánh giá chung

```
# Initialize and train Naive Bayes classifier
classifier = MultinomialNB()
classifier.fit(X_train_NB, y_train_NB)

end_time_nb = time.time()
elapsed_time_nb = end_time_nb - start_time_nb
```

Sau khi thực hiện các bước chuyển đổi văn bản, nhóm sẽ tiến hành huấn luyện mô hình thông qua hàm có sẵn của thư viện Scikit-learn. Để tiện cho việc sử dụng mô hình sau này trong phần demo dự án, nhóm sẽ lưu mô hình vào file “naive_bayes_model.pkl”.

Sau khi huấn luyện thành công mô hình, nhóm sẽ tiến hành đánh giá hiệu quả của mô hình thông qua 4 chỉ số accuracy, recall, precision và f1 score trên mỗi nhãn.

Classification Report:				
	precision	recall	f1-score	support
0	0.80	0.81	0.80	1871
1	0.81	0.84	0.82	2123
2	0.83	0.84	0.83	1832
3	0.88	0.83	0.86	1875
4	0.81	0.88	0.84	1840
5	0.84	0.73	0.78	1823
6	0.82	0.83	0.82	1736
7	0.75	0.89	0.82	2805
8	0.77	0.83	0.80	2081
9	0.83	0.73	0.78	1850
10	0.88	0.77	0.82	1740
11	0.82	0.75	0.78	2298
accuracy			0.81	23874
macro avg	0.82	0.81	0.81	23874
weighted avg	0.82	0.81	0.81	23874

Hình 20. Điểm số đánh giá trên các lớp của mô hình Naive Bayes

Kết quả cho thấy điểm số đánh giá chung accuracy là 0.81, điểm số này thật sự không phải quá cao nhưng cũng có thể xem là mô hình vẫn thể hiện tốt với đa số các giá trị trong tập test. Khi xét đến 3 chỉ số còn lại trên mỗi nhãn khác nhau, có thể nhận thấy các giá trị này là tương đối đồng đều ở mọi nhãn.

- Precision có điểm số trải từ [0.75;0.88].
- Recall có điểm số trải từ [0.73;0.89].
- F1-score có điểm số trải từ [0.78;0.86].

Điểm số có sự chênh lệch nhiều nhất là Recall khi giá trị của nhãn 5 History & Documentary) và 9 (Science and Technology) có cùng điểm số thấp nhất là 0.73. Ngoài ra điểm Recall của một số nhãn khác cũng đạt điểm số dưới 0.8.

5.2 Mô hình Logistic Regression

5.2.1 Chuyển đổi văn bản và tách tập dữ liệu huấn luyện

```
# Dữ liệu huấn luyện và nhãn tương ứng
X_LR = sentences # Dữ liệu văn bản huấn luyện
y_LR = labels    # Nhãn tương ứng

# Chia dữ liệu thành tập huấn luyện và tập kiểm tra
X_train_LR, X_test_LR, y_train_LR, y_test_LR = train_test_split(X_LR,
y_LR, test_size=0.2, random_state=42)

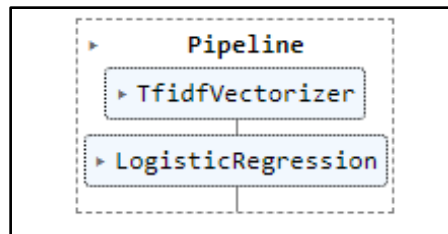
#Xây dựng pipeline cho mô hình
pipeline = Pipeline([
    ('tfidf', TfidfVectorizer()), # Chuyển đổi văn bản thành vector TF-
IDF
```

```
('clf', LogisticRegression()), # Sử dụng mô hình Logistic
Regression
])
```

Thực hiện chia tập dữ liệu như các bước đã thực hiện ở mô hình Naive Bayes. Tuy nhiên, nhóm sẽ sử dụng phương pháp TF-IDF để thực hiện vector hóa dữ liệu văn bản huấn luyện thay vì sử dụng mô hình BoW. Lý do nhóm chọn TF-IDF là vì mô hình tạo ra một vector đặc trưng có kích thước cố định dựa trên toàn bộ tập dữ liệu, với mỗi phần tử trong vector đại diện cho trọng số của từ tương ứng trong từ điển. Điều này tạo ra các đặc trưng đa dạng, giúp mô hình Logistic Regression có thêm thông tin để học từ dữ liệu và sự kết hợp giữa mô hình TF-IDF và Logistic Regression thường cho hiệu suất tốt trên nhiều bài toán phân loại văn bản, như phân loại cảm xúc trong bình luận, phát hiện spam trong email, hay phân loại chủ đề của các văn bản.

5.2.2 Huấn luyện mô hình và đánh giá chung

```
# Huấn luyện mô hình
pipeline.fit(X_train_LR, y_train_LR)
```



Hình 21. Pipeline của mô hình Logistic Regression

```
# Lưu model thành file pickle
joblib.dump(pipeline, 'logistic_regression_model.pkl')
```

Thực hiện các bước tương tự như với mô hình Naive Bayes, sau khi huấn luyện và lưu mô hình thành công nhóm sẽ in các chỉ số điểm đánh giá để nhận xét mô hình.

Classification Report:				
	precision	recall	f1-score	support
Animals	0.90	0.85	0.88	1871
Education	0.90	0.86	0.88	2123
Film & Animation	0.89	0.88	0.89	1832
Food	0.92	0.89	0.90	1875
Gaming	0.90	0.89	0.90	1840
History & Documentary	0.89	0.85	0.87	1823
Music	0.84	0.88	0.86	1736
News & Politics	0.89	0.92	0.90	2805
NonProfit & Activism	0.88	0.83	0.85	2081
Science & Technology	0.84	0.85	0.84	1850
Sports	0.72	0.87	0.79	1740
Travel & Events	0.87	0.85	0.86	2298
accuracy			0.87	23874
macro avg	0.87	0.87	0.87	23874
weighted avg	0.87	0.87	0.87	23874

Hình 22. Điểm số đánh giá trên các lớp của mô hình Logistic Regression

Điểm Accuracy cho thấy mô hình hoạt động tương đối hiệu quả trên tập test khi có đến 87% các dòng dữ liệu được gán nhãn chính xác. Các chỉ số điểm còn lại cũng cho thấy sự đồng đều trên các nhãn khác nhau tuy nhiên chỉ có nhãn Sports lại có điểm số không tốt ở điểm Precision. Nhưng xét về tổng thể mô hình Logistic Regression thể hiện xuất sắc hơn hẳn so với mô hình Naive Bayes.

5.3 Mô hình LSTM

5.3.1 Chuẩn bị dữ liệu huấn luyện

```
data_LSTM = pd.DataFrame({
    'clean_title': sentences,
    'Topic': encoded_labels,
    'tokens': tokens
})
# Chia tập train (70%) và tập còn lại (30%)
train, temp_data = train_test_split(data_LSTM, test_size=0.2,
random_state=42)

# Chia tập còn lại thành tập validation (50%) và tập test (50%)
val, test = train_test_split(temp_data, test_size=0.5, random_state=42)

train_x = train['clean_title'].tolist()
val_x = val['clean_title'].tolist()
test_x = test['clean_title'].tolist()

word_train = [token for token in train.tokens]
word_valid=[token for token in val.tokens]
word_test = [token for token in test.tokens]
```

Đầu tiên nhóm sẽ khởi tạo dataframe “data_LSTM” chứa các cột:

- 'clean_title': Chứa các câu đã được xử lý để loại bỏ nhiễu và làm sạch.
- 'Topic': Chứa nhãn của mỗi câu, đã được mã hóa.
- 'tokens': Chứa danh sách các từ được tách từ các câu.

Kế đến tiến hành chia dữ liệu thành ba phần: tập huấn luyện (train), tập validation (val) và tập kiểm tra (test) với tỷ lệ là 7:1,5:1,5. Ở đây tập validation và tập kiểm tra được chia đều từ phần còn lại của dữ liệu, mỗi phần chiếm 15%. Tập validation được sử dụng để đánh giá hiệu suất của mô hình trong quá trình huấn luyện và đo lường khả năng tổng quát hóa của mô hình trên dữ liệu mà nó chưa từng thấy.

5.3.2 Xây dựng mô hình Word2Vec

Kế đến nhóm sẽ tiến hành xây dựng mô hình W2V để chuyển đổi dữ liệu huấn luyện sang dạng vector. Mô hình W2V tạo ra các vector biểu diễn cho từng từ trong từ vựng sao cho các từ có ý nghĩa tương tự sẽ có vector gần nhau trong không gian vector. Điều này giúp mô hình LSTM hiểu được ý nghĩa của từng từ trong ngữ cảnh của nó.

```
# Cài đặt các chỉ số
min_count=2
window=5
vector_size=100
alpha=1e-3
min_alpha=1e-4
negative=5

# Tạo mô hình Word2Vec
w2v_model = Word2Vec(min_count=min_count, window=window,
vector_size=vector_size, sg=0)
# Xây dựng từ điển cho tập dữ liệu
w2v_model.build_vocab(word_train)

#Huấn luyện mô hình
w2v_model.train(word_train, total_examples=w2v_model.corpus_count,
epochs=100, report_delay=1,compute_loss=True)
```

Tiến hành thực hiện các bước xây dựng mô hình như sau:

Đầu tiên cài đặt các chỉ số cho mô hình:

- min_count: Số lần xuất hiện tối thiểu của một từ trong tập dữ liệu để được đưa vào từ điển.

- window: Số lượng từ xung quanh mỗi từ mục tiêu trong một câu được xem xét trong quá trình huấn luyện.
- vector_size: Kích thước của các vector từ sau khi được mã hóa bằng mô hình Word2Vec.
- alpha: Tốc độ học ban đầu.
- min_alpha: Tốc độ học tối thiểu.
- negative: Số lượng mẫu âm được sử dụng trong phương pháp lấy mẫu âm tích cực (negative sampling).

Tiếp theo nhóm tạo mô hình Word2Vec sử dụng các tham số được cài đặt ở trên và xây dựng từ điển cho tập dữ liệu. Cuối cùng là huấn luyện mô hình bằng phương thức train() để huấn luyện mô hình với tập dữ liệu huấn luyện với các tham số:

- total_examples=w2v_model.corpus_count: Số lượng ví dụ trong tập dữ liệu huấn luyện.
- epochs=100: Số lần lặp lại qua toàn bộ tập dữ liệu huấn luyện.
- report_delay=1: Tần suất báo cáo tiến trình huấn luyện (số epoch) trong quá trình huấn luyện.
- compute_loss=True: Tính toán và báo cáo giá trị loss trong quá trình huấn luyện.

5.3.3 Huấn luyện mô hình và đánh giá chung

Sau khi đã chuẩn bị dữ liệu và mô hình vector hóa, nhóm sẽ tiến hành xây dựng mô hình LSTM và huấn luyện mô hình dựa vào dữ liệu được chuẩn bị.

```
# # Kích thước của từ điển
# vocab_size = len(w2v_model.wv.key_to_index) + 1

# Kích thước của vector nhúng
embedding_dim = w2v_model.vector_size

# Xây dựng mô hình LSTM
model_LSTM_topic = Sequential()
model_LSTM_topic.add(LSTM(units=100, return_sequences=True,
input_shape=(1, embedding_dim)))
model_LSTM_topic.add(Dropout(0.2))
model_LSTM_topic.add(LSTM(units=75, return_sequences=True))
model_LSTM_topic.add(Dropout(0.2))
model_LSTM_topic.add(LSTM(units=50, return_sequences=True))
model_LSTM_topic.add(Dropout(0.2))
model_LSTM_topic.add(LSTM(units=30))
model_LSTM_topic.add(Dropout(0.2))
model_LSTM_topic.add(Dense(units=12, activation='softmax'))
```

```
# Biên dịch mô hình
model_LSTM_topic.compile(optimizer='adam',
loss='categorical_crossentropy', metrics=['accuracy'])

print(model_LSTM_topic.summary())
```

Mô hình LSTM được xây dựng với một loạt các lớp LSTM được xếp chồng lên nhau, kết hợp với các lớp Dropout để giảm overfitting. Lớp LSTM đầu tiên có units=100 và return_sequences=True để trả về dãy đầy đủ của các giá trị trạng thái ẩn cho mỗi time step. Các lớp LSTM tiếp theo giảm dần số units và không cần trả về dãy đầy đủ của các giá trị trạng thái ẩn. Cuối cùng, một lớp Dense với units=12 và activation là 'softmax' được sử dụng để phân loại đầu ra. Mô hình được biên dịch với optimizer 'adam', loss function là 'categorical_crossentropy' (do đây là bài toán phân loại đa lớp), và độ đo hiệu suất là 'accuracy'.

Model: "sequential"		
Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 1, 100)	80400
dropout (Dropout)	(None, 1, 100)	0
lstm_1 (LSTM)	(None, 1, 75)	52800
dropout_1 (Dropout)	(None, 1, 75)	0
lstm_2 (LSTM)	(None, 1, 50)	25200
dropout_2 (Dropout)	(None, 1, 50)	0
lstm_3 (LSTM)	(None, 30)	9720
dropout_3 (Dropout)	(None, 30)	0
dense (Dense)	(None, 12)	372
Total params: 168492 (658.17 KB)		
Trainable params: 168492 (658.17 KB)		
Non-trainable params: 0 (0.00 Byte)		
None		

Hình 23. Cấu trúc mô hình LSTM

```
def get_vector(word_list, model):
    # Khởi tạo một vector 0
    vec = np.zeros(model.vector_size).reshape((1, model.vector_size))
    count = 0.
    for word in word_list:
        # Thêm vector của từ vào vec
        if word in model.wv.key_to_index: # check if the word is in
the model's vocabulary
            vec += model.wv.get_vector(word).reshape((1,
model.vector_size))
            count += 1.
    if count != 0:
        vec /= count
    return vec
```

```

train_x = np.concatenate([get_vector(sent, w2v_model) for sent in
word_train])
val_x = np.concatenate([get_vector(sent, w2v_model) for sent in
word_valid])
test_x = np.concatenate([get_vector(sent, w2v_model) for sent in
word_test])

#xác định các tập để huấn luyện mô hình
train_y = to_categorical(train['Topic'])
val_y = to_categorical(val['Topic'])

train_x = train_x.reshape(-1, 1, embedding_dim)
val_x = val_x.reshape(-1, 1, embedding_dim)

start_time_lstm = time.time()
# Huấn luyện mô hình
early_stopping = EarlyStopping(monitor='val_loss', patience=3,
restore_best_weights=True)
model_LSTM_topic.fit(train_x, train_y, epochs=50, batch_size=32,
validation_data=(val_x, val_y), callbacks=[early_stopping])

end_time_lstm = time.time()
elapsed_time_lstm = end_time_lstm - start_time_lstm

```

Sau khi xây dựng các cấu trúc của mô hình LSTM, nhóm sẽ tiến hành huấn luyện mô hình qua các bước như sau:

Đầu tiên định nghĩa hàm `get_vector()` để chuyển đổi mỗi câu thành một vector bằng cách lấy trung bình của các vector từ của các từ trong câu và sử dụng để biến đổi các câu trong tập huấn luyện, tập validation và tập kiểm tra thành các vector. Kế đến biến đổi dữ liệu thành dạng phù hợp với mô hình LSTM bằng cách thêm một chiều mới (1) cho phù hợp với đầu vào của mô hình LSTM và chuyển đổi nhãn thành dạng one-hot encoding bằng hàm `to_categorical()` và cuối cùng là sử dụng phương thức `fit()` để huấn luyện mô hình trên dữ liệu huấn luyện và validation với các điều kiện sau:

- `EarlyStopping` callback để dừng quá trình huấn luyện sớm nếu không có cải thiện đáng kể trong việc giảm validation loss sau một số lượng epochs quy định.
- `restore_best_weights = True` để lưu lại trọng số của mô hình khi validation loss đạt giá trị thấp nhất.

```

# Lưu model thành file pickle
model_LSTM_topic.save('lstm_model.h5')

```

Lưu mô hình vào file 'lstm_model.h5'.

5.3.4 Đánh giá chung

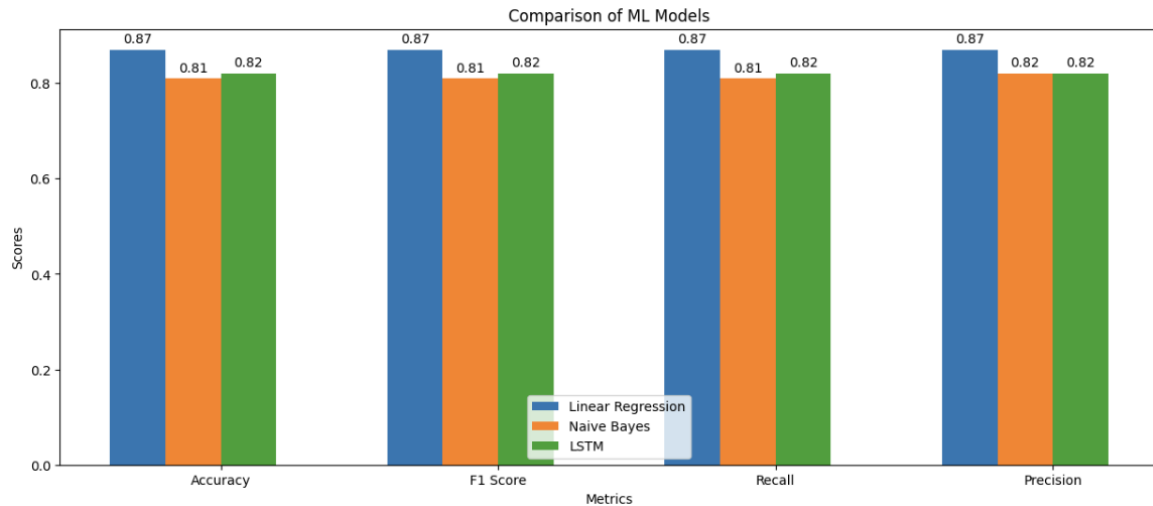
Classification Report:				
	precision	recall	f1-score	support
0	0.82	0.82	0.82	890
1	0.82	0.84	0.83	1067
2	0.84	0.83	0.84	944
3	0.89	0.83	0.86	914
4	0.84	0.87	0.86	900
5	0.84	0.77	0.80	913
6	0.83	0.83	0.83	875
7	0.85	0.90	0.87	1419
8	0.86	0.79	0.82	1062
9	0.78	0.78	0.78	920
10	0.69	0.83	0.75	896
11	0.81	0.75	0.78	1137
accuracy			0.82	11937
macro avg	0.82	0.82	0.82	11937
weighted avg	0.82	0.82	0.82	11937

Hình 24. Điểm số đánh giá trên các lớp của mô hình LSTM

Điểm Accuracy cho thấy mô hình LSTM hoạt động tương đối ổn trên tập test khi đạt trên 82% các dòng dữ liệu được gán nhãn chính xác. Các chỉ số điểm còn lại cũng cho thấy sự đồng đều trên các nhãn khác nhau tuy nhiên chỉ có nhãn 10 (Sports) lại có điểm precision khá thấp (0.69). Điều này cũng diễn ra tương tự đối với mô hình Logistic Regression. Tuy nhiên tổng thể mô hình vẫn không đạt được hiệu quả tốt như mô hình Logistic Regression.

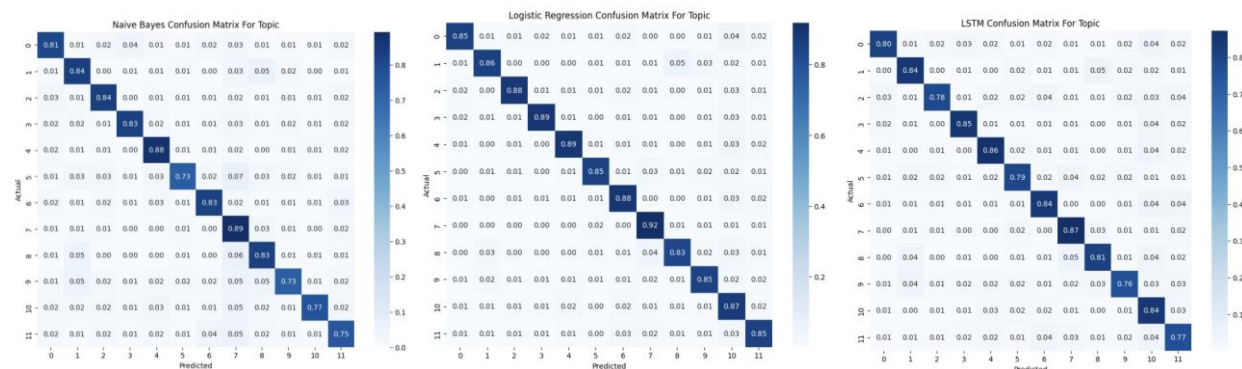
5.4 So sánh kết quả

Qua kết quả từ đánh sơ bộ 3 mô hình có thể thấy được các mô hình đã hoạt động tương đối hiệu quả. Điểm số đánh giá theo từng Topic là không chênh lệch quá nhiều, tuy nhiên có một vài nhãn giá trị có số điểm thấp hơn tương đối so với các nhãn còn lại. Để so sánh trực quan hơn về tính hiệu quả của cả ba mô hình nhóm sẽ tiến hành trực quan hóa các chỉ số đánh giá trên.



Hình 25. Biểu đồ cột so sánh 4 điểm số đánh giá của 3 mô hình

Có thể thấy thông qua hình ảnh trực quan biểu đồ cột, mô hình Logistic Regression thể hiện vượt trội hơn hẳn 2 mô hình còn lại với đồng loạt đạt 0,87 điểm ở cả 4 chỉ số. Trong khi đó 2 mô hình LSTM và Naïve Bayes có điểm số đánh giá tương đối gần nhau tuy nhiên mô hình LSTM lại thể hiện ổn hơn ở 2 chỉ số F1 score và Recall.



Hình 26. Ma trận nhầm lẫn của 3 mô hình

Hình ảnh trực quan từ ma trận nhầm lẫn cũng cho thấy kết quả giống với biểu đồ cột trên. Đường chéo chính của ma trận mô hình Logistic Regression đậm hơn hẳn so với 2 mô hình còn lại.

Ngoài ra còn có thể thấy mô hình LSTM và Naïve Bayes đều gặp vấn đề với nhãn số 9 (Science & Technology) và các giá trị bị nhầm lẫn thường bị đánh nhầm sang nhãn số 2 (Education). Điều này có thể lý giải do sự mù mờ khá cao giữa 2 chủ đề này.

Trong khi đó, mô hình Logistic Regression lại gặp khó khăn đối với nhãn số 8 là (NonProfit & Activism) khi đây là nhãn mà tỷ lệ đánh đúng của mô hình đạt điểm thấp nhất.

Thời gian chạy của mô hình Logistic Regression: 40.52636790275574
Thời gian chạy của mô hình Naive Bayes: 0.11581850051879883
Thời gian chạy của mô hình LSTM: 340.54167914390564

Hình 27. Thời gian huấn luyện của 3 mô hình

Có thể thấy khi xét về tốc độ huấn luyện của 3 mô hình thì LSTM tốn rất nhiều thời gian để mô hình có thể xây dựng được mô hình tối ưu, điều này có thể được giải thích do mô hình LSTM có số lượng tham số lớn hơn so với mô hình Logistic Regression và Naive Bayes. Điều này đồng nghĩa với việc cần nhiều thời gian để tính toán và cập nhật các trọng số trong quá trình huấn luyện. Ngoài ra, mô hình LSTM thực hiện nhiều phép tính phức tạp hơn trong quá trình lan truyền ngược so với các mô hình đơn giản hơn. Điều này là do cơ chế của LSTM cần xử lý các trạng thái ẩn và các cổng đầu vào/ra đặc biệt.

Tóm lại, xét cả về hiệu quả và tốc độ huấn luyện, mô hình tối ưu nhất chính là Logistic Regression khi mô hình này không những đạt điểm số đánh giá vượt trội so với 2 mô hình còn lại mà thời gian thực hiện huấn luyện tuy không phải tối ưu nhất nhưng vẫn có thể chấp nhận được.

CHƯƠNG 6: XÂY DỰNG HỆ THỐNG GỢI Ý VÀ DEMO GIAO DIỆN NGƯỜI DÙNG (RECOMMENDATION SYSTEM & UI DEMO)

6.1 Hệ thống gợi ý video

6.1.1 Dữ liệu gợi ý

Để xây dựng được hệ thống gợi ý video cho người xem, trước hết nhóm cần phải chuẩn bị dữ liệu gợi ý có sẵn.

```
label_counts = data["Topic"].value_counts(normalize=True)
sample_size_per_label = 1000 // len(label_counts)

# Initialize an empty DataFrame
recommend_data = pd.DataFrame()

# Assume label_counts contains the proportion of each label
# sample_size_per_label is the desired sample size per label

# Loop through each label to sample data
for label, proportion in label_counts.items():
    # Sample rows for each label based on the sample size per label
    sampled_rows = data[data['Topic'] == label].sample(n=sample_size_per_label, replace=False)
    # Concatenate sampled rows to the recommend_data DataFrame
    recommend_data = pd.concat([recommend_data, sampled_rows])

# If the total sampled rows are less than 1000, sample random rows from
the entire dataset to ensure 1000 rows

# Sample random rows if necessary
if len(recommend_data) < 1000:
    remaining_rows = 1000 - len(recommend_data)
    remaining_sample = data.sample(n=remaining_rows, replace=False)
    recommend_data = pd.concat([recommend_data, remaining_sample])

# Reset the index of the new DataFrame
recommend_data.reset_index(drop=True, inplace=True)

recommend_data.to_csv('Recommend videos sample.csv', index=False)

video = recommend_data['clean_title']
topic = recommend_data['Topic']
tokens = recommend_data['tokens']
re_title = recommend_data['Title']
```

```
dataset_videos = video.tolist()
```

Nhóm sẽ tạo ra một dataframe mới “recommend_data” để lưu trữ dữ liệu các video gợi ý cho người xem với tổng số lượng là 1000 video với tỷ lệ các nhãn là tương tự như đối với bộ dữ liệu gốc ban đầu. Sau cùng sẽ là lưu trữ dữ liệu dưới dạng file csv để dễ dàng thực hiện demo ở bước sau.

6.1.2 Xây dựng hệ thống gợi ý

Sau bước huấn luyện mô hình, đánh giá và so sánh kết quả, nhóm sẽ chọn ra mô hình hoạt động tối ưu nhất là Logistic Regression để xây dựng hệ thống gợi ý video.

6.1.2.1 Quy tắc hoạt động

Các bước hoạt động của hệ thống gợi ý mà nhóm xây dựng như sau:

- Bước 1: Hệ thống sẽ nhận thông tin từ người dùng về title của một video bất kỳ, sử dụng mô hình phân loại topic, hệ thống sẽ xác định loại video đó thuộc nhóm topic nào.
- Bước 2: Hệ thống sẽ tính toán độ tương đồng của video người dùng đăng nhập và các video trong bộ dữ liệu gợi ý trong cùng một topic được xác định từ bước 1.
- Bước 3: Hệ thống sắp xếp các video gợi ý theo độ tương đồng.
- Bước 4: Hệ thống đưa ra 5 video có độ tương đồng cao nhất cho người dùng.

Sau khi kết thúc 4 bước thực hiện, hệ thống sẽ trả về cho người dùng những video có độ tương thích cao nhất đối với video mà người dùng đã nhập từ trước. Hệ thống hoạt động trên nguyên tắc giả định rằng video mà người dùng nhập vào hệ thống thuộc sở thích xem phim của họ.

6.1.2.2 Xây dựng mô hình vector hóa và hàm tính toán độ tương đồng

Đầu tiên, nhóm sẽ tiến hành xây dựng mô hình vector hóa dữ liệu title được nhập từ người dùng để đưa vào mô hình phân loại. Do mô hình mà nhóm sử dụng là Logistic Regression sử dụng phương pháp TF-IDF nên mô hình này cũng sẽ được huấn luyện theo như vậy.

```
# Tạo và cấu hình vectorizer
vectorizer = TfidfVectorizer(max_features=1000, stop_words='english')

# Huấn luyện vectorizer
vectorizer.fit(dataset_videos)
```


Tạo đối tượng TfidfVectorizer với các tham số được chỉ định. Trong trường hợp này, max_features=1000 xác định rằng chỉ có tối đa 1000 đặc trưng (từ) sẽ được sử dụng để tạo vector TF-IDF. Điều này giúp giảm chiều của không gian đặc trưng, làm giảm độ phức tạp của mô hình và tăng tốc độ huấn luyện. Tham số stop_words='english' chỉ định rằng các stop words trong tiếng Anh sẽ được loại bỏ khỏi văn bản trước khi chuyển đổi thành vector. Stop words là những từ phổ biến như "a", "an", "the",... không mang ý nghĩa quan trọng và thường không cần thiết trong việc phân loại hoặc xử lý văn bản.

```
def classify_video_title(title, pipeline):
    # Sử dụng mô hình đã huấn luyện để phân loại tựa đề video
    predicted_label = pipeline.predict([title])[0]
    return predicted_label

def compute_similarity(title, videos, labels, predicted_label,
vectorizer, recommended_title):
    # Lọc chỉ các video cùng thể loại với tựa đề video nhập
    same_label_indices = [i for i, label in enumerate(labels) if label
== predicted_label]
    same_label_videos = videos.iloc[same_label_indices]
    same_label_parameters = recommended_title.iloc[same_label_indices]

    # Biến đổi tựa đề video nhập thành vector đặc trưng
    title_vector = vectorizer.transform([title])

    # Biến đổi tựa đề của các video cùng thể loại thành vector đặc
    trung
    same_label_vectors = vectorizer.transform(same_label_videos)

    # Tính độ tương đồng giữa tựa đề video nhập và các video cùng thể
    loại
    similarities = cosine_similarity(title_vector, same_label_vectors)

    # Sắp xếp các video theo độ tương đồng và lấy 5 video có độ tương
    đồng cao nhất
    top_indices = np.argsort(similarities.flatten())[-5:]
    top_videos = same_label_videos.iloc[top_indices]
    top_parameters = same_label_parameters.iloc[top_indices]

    return top_parameters
```

Kế đến nhóm sẽ tiến hành xây dựng 2 hàm tính toán cho hệ thống gợi ý bao gồm hai hàm chính để phân loại và tính toán độ tương đồng giữa tựa đề video nhập và các video cùng thể loại:

Hàm classify_video_title(title, pipeline):

- Đầu vào: tựa đề video title cần phân loại và pipeline pipeline chứa mô hình đã được huấn luyện để phân loại tựa đề video.
- Đầu ra: nhãn dự đoán cho tựa đề video.
- Hàm này sử dụng mô hình đã được huấn luyện để dự đoán nhãn cho tựa đề video nhập và trả về nhãn dự đoán.

Hàm `compute_similarity` (title, videos, labels, predicted_label, vectorizer, recommended_title):

- Đầu vào:
 - title: tựa đề video nhập.
 - videos: dataframe chứa thông tin về các video (bao gồm tựa đề và các đặc trưng khác).
 - labels: danh sách các nhãn tương ứng với các video.
 - predicted_label: nhãn dự đoán cho tựa đề video nhập.
 - vectorizer: mô hình để biến đổi tựa đề video thành vector đặc trưng.
 - recommended_title: dataframe chứa các thông tin khác về các video cùng thể loại với tựa đề video nhập.
- Đầu ra: tựa đề của 5 video có độ tương đồng cao nhất với tựa đề video nhập.

Sau khi xây dựng thành công hệ thống, nhóm tiến hành thực hiện thử với các dữ liệu video ngẫu nhiên

```

Nhập tựa đề video: Mint Lemonade - Mike Beating, créature sonore Jazz Club 🍷 Relaxing lofi jazz ~ Chill Smooth Background Jazz For Work, Study, Focus
Video thuộc topic: Music
Gợi ý 5 video có độ tương đồng cao nhất:
1. Bach Study Music Playlist 🎹 Instrumental Classical Music Mix for Studying, Concentration, Relaxation
2. Deep Focus Music To Improve Concentration - 12 Hours of Ambient Study Music to Concentrate #676
3. San Antonio Spurs vs Utah Jazz - Full Game Highlights | February 25, 2024 NBA Season
4. Work & Jazz 🎷 Relaxing Piano Jazz Instrumental Music & Upbeat Bossa Nova for Begin the week
5. Smooth Jazz Chillout Lounge • Smooth Jazz Saxophone Instrumental Music for Relaxing, Dinner, Study
  
```

Hình 28. Kết quả gợi ý của hệ thống với một video bất kỳ

Kết quả cho thấy hệ thống đã tính toán và đưa ra các video gợi ý khá ổn khi các video gợi ý thuộc các chủ đề rất giống với video ngẫu nhiên được nhập vào mô hình.

6.2 Xây dựng demo giao diện người dùng

6.2.1 Đoạn mã thực hiện

Ở phần này, nhóm thực hiện triển khai mô hình thu được để xây dựng ứng dụng thực nghiệm. Với kinh nghiệm đã từng trải nghiệm qua framework Flask, nhóm quyết định sử dụng thư viện này để xây dựng một hệ thống hoàn chỉnh kết hợp giữa Python và các ngôn ngữ lập trình frontend như HTML, CSS.

Ý tưởng sẽ là nhóm sử dụng cả ba mô hình máy học và học sâu để tìm chủ đề từ đoạn mô tả mà người dùng nhập vào, sau đó sử dụng thuật toán theo các quy tắc trong tệp tin

ipy notebook mà nhóm đã thực hiện. Sau khi dự đoán, nhóm sẽ sử dụng quy tắc đề xuất các video liên quan cho người dùng theo công thức độ tương đồng cosin (cosine similarity) từ chủ đề mà số đông các mô hình chọn được.

```
1  from flask import Flask, render_template, request
2  import string, re, pandas as pd, numpy as np, html
3  import pickle, joblib
4  from gensim.models import Word2Vec
5  import matplotlib.pyplot as plt, seaborn as sns
6  from keras.models import load_model
7  from sklearn.metrics.pairwise import cosine_similarity
8
9  # App & Models
10 app = Flask(__name__)
11 vectorizer = joblib.load('models/count_vectorizer.pkl')
12 lr_vectorizer = pickle.load(open('models/tfidf_vectorizer.pkl', 'rb'))
13 w2v_model = Word2Vec.load('models/word2vec_model.bin')
14 model = joblib.load('models/naive_bayes_model.pkl')
15 lr_model = pickle.load(open('models/logistic_regression_model.pkl', 'rb'))
16 lstm_model = load_model('models/lstm_model.h5')
17
18 # Labels & Dataset for Recommendation System
19 ...
20
21 > def is_english(word): ...
26
27 > def remove_non_english(text): ...
40
41 > def processing(text): ...
54
55 > def get_vector(word_list, model): ...
65
66 > def predict(text): ...
86
87 > def viz(models, accuracy_scores, labels): ...
115
116 > def check_prediction(label_nb, label_lr, label_lstm): ...
125
126 > def compute_similarity(title, videos, labels, predicted_label, vectorizer, recommended_title): ...
145
146 @app.route('/')
147 > def home(): ...
149
150 @app.route('/predict', methods=['POST'])
151 > def backend(): ...
167
168 if __name__ == '__main__':
169     app.run(debug=True)
```

Hình 29. Preview tệp tin “app.py” thực hiện xây dựng backend cho trang web thực nghiệm

Các hàm mà nhóm sử dụng trong tệp tin này đa số sẽ tương đồng với các hàm nhóm đã thực hiện tại bước xây dựng mô hình phân lớp chủ đề ở chương V, trong đó, các tệp tin vectorizer và model đã được thực hiện lưu riêng từ giai đoạn fit mô hình và sử dụng nhằm tiết kiệm thời gian huấn luyện lại. Nhóm thực hiện đặt tên cho tệp tin này là “app.py”.

Tương tự với backend, để trang web có thể sử dụng cho nhiều mục đích và hướng tới nhiều đối tượng hơn, nhóm cũng thực hiện xây dựng phần giao diện frontend cho trang web bằng ngôn ngữ HTML/CSS.

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Youtube Topic Classification</title>
7   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet"
8     integrity="sha384-QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNiyT2brjXh0JMHjY6hW+ALEwIH" crossorigin="anonymous">
9   <style>
10 > .bg-image { ...
22 > }
23 > #page-container { ...
31 > }
32 > h1 { ...
35 > }
36 > .logo { ...
42 > }
43 > .footer-text { ...
50 > }
51 </style>
52 </head>
53 <body>
54 <div class="bg-image"></div>
55 
56 <div class="footer-text">© Project made by Personality (Group 5). UEH Big Data Application 2024.</div>
57 <div class="d-flex flex-column gap-3 px-4 py-3 mx-auto mt-5" id="page-container">
58 <h1>Youtube Topic Classification</h1>
59 <form action="/predict" id="youtube-form" method="post" onsubmit="return checkForm()">...
65 </form>
66 <input type="submit" style="width: fit-content;" form="youtube-form" class="btn btn-success px-3 align-self-center" value="Cheers">
67 </div>
68 <script>
69 > function checkForm() { ...
77 >
78 > /* // Nghịch tính năng add-up...
85
86 let textarea = document.querySelector("textarea");
87 let container = document.getElementById("page-container");
88 textarea.addEventListener("keyup", () => {
89   container.style.setProperty('--textarea-height', calcHeight(textarea.value));
90 });
91 </script>
92 </body>
93 </html>

```

Hình 30. Preview tệp tin “index.html” thực hiện xây dựng giao diện mặc định cho trang web thực nghiệm

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Prediction Result</title>
7   <style>
8 > .background { ...
20 > }
21 > body { ...
26 > }
27 > .container { ...
34 > }
35 > h1 { ...
38 > }
39 > h2 { ...
41 > }
42 > .button { ...
50 > }
51 > .button:hover { ...
53 > }
54 > .footer-text { ...
61 > }
62 </style>
63 </head>
64 <body>
65 <div class="background"></div>
66 <div class="footer-text">© Personality (Group 5). UEH Big Data Application 2024.</div>
67 <div class="container">
68 <h1>PREDICTION RESULT</h1>
69 <h2>Naive Bayes predicted topic: {{ prediction_nb }} ({{ accuracy_nb }}%)</h2>
70 <h2>Logistic Regression predicted topic: {{ prediction_lr }} ({{ accuracy_lr }}%)</h2>
71 <h2>LSTM predicted topic: {{ prediction_lstm }} ({{ accuracy_lstm }}%)</h2>
72 
73 <form action="/recommend" id="recommend-form" method="post">...
75 </form>
76 <div class="recommendations" style="margin-top: 20px;">...
83 </div>
84 <a href="/" class="button">Back to Home</a>
85 </div>
86 </body>
87 </html>

```

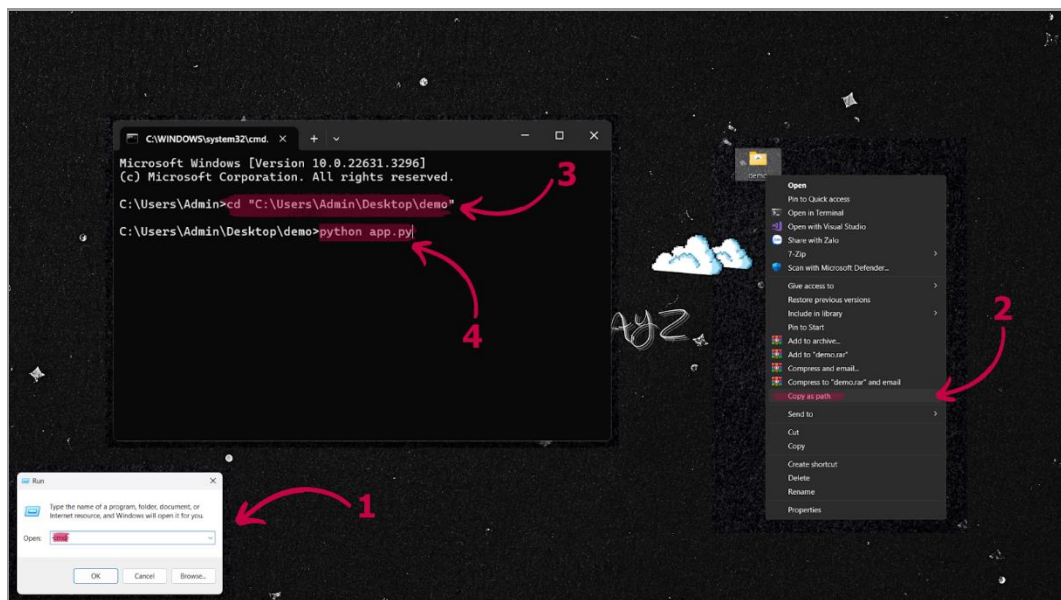
Hình 31. Preview tệp tin “result.html” thực hiện xây dựng giao diện kết quả cho trang web thực nghiệm

6.2.2 Hướng dẫn khởi chạy chương trình

Flask là một framework phát triển ứng dụng web bằng Python với cộng đồng người sử dụng đang phát triển tương đối nhanh. Một trong những điểm mạnh của Flask là khả năng phát triển ứng dụng web nhanh chóng và dễ dàng. Tuy nhiên, mặc dù có thể triển khai Flask trên môi trường local dễ dàng để thử nghiệm và phát triển, nhưng việc chỉ chạy ứng dụng trên local cũng có hạn chế: Ứng dụng được tạo ra này chỉ có thể truy cập từ máy tính cục bộ và không thể được truy cập từ bên ngoài mạng, điều này có thể làm giảm tính ứng dụng của ứng dụng nếu cần phải chia sẻ hoặc truy cập từ xa.

Ngoài ra, để có thể khởi chạy ứng dụng tối ưu nhất, các máy tính cục bộ này cần cài đặt một số tính năng bắt buộc mà tệp tin yêu cầu. Vậy nên nếu trong quá trình thực hiện các bước khởi chạy, người dùng gặp các vấn đề về tệp tin có thể tham khảo trong mục 6.2.4 dưới đây để có thể thực hiện các bước khắc phục.

Để thực hiện khởi chạy ứng dụng xây dựng được, người dùng cần thực hiện truy cập vào terminal của máy tính cục bộ bằng Command Prompt, sau đó, người dùng thực hiện truy cập vào thư mục lưu trữ tệp tin backend “app.py” và khởi chạy nó.



Hình 32. Giới thiệu các bước thực hiện khởi chạy ứng dụng thực nghiệm

Chi tiết hơn, đầu tiên để truy cập vào thư mục lưu trữ tệp tin, người dùng sẽ sử dụng cmd và nhập vào:

cd <đường dẫn tới thư mục>

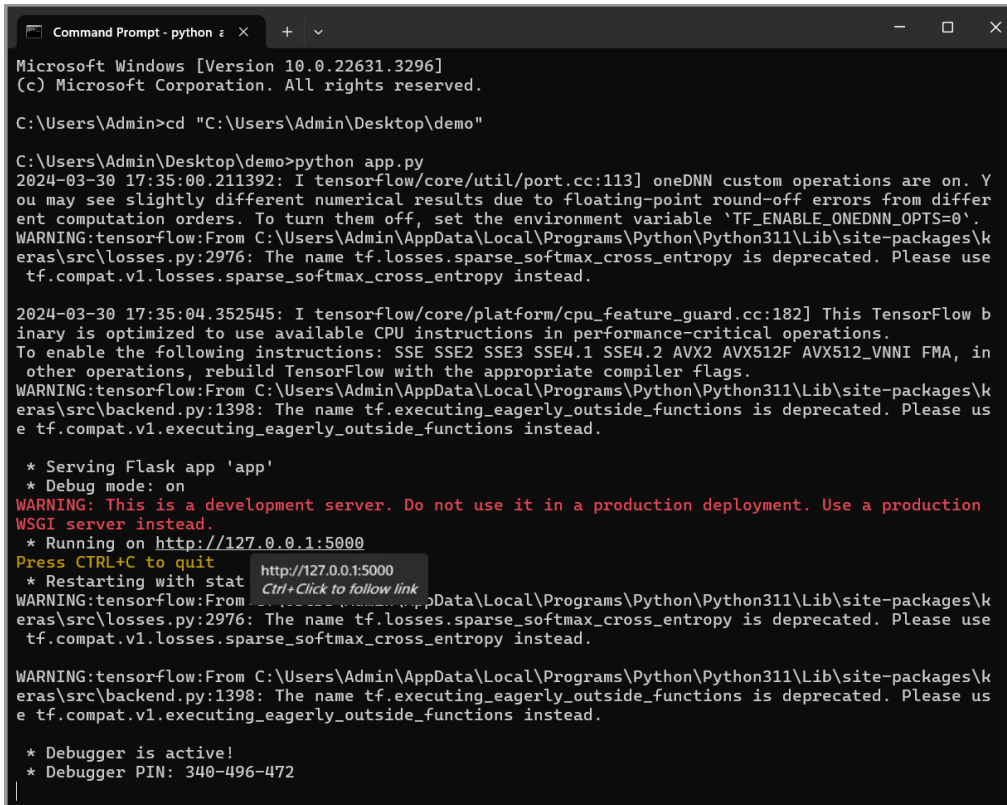
Sau khi máy tính cục bộ đã chuyển thư mục làm việc hiện tại sang thư mục chứa tệp tin backend, người dùng tiếp tục thực hiện khởi chạy tệp tin Python này bằng cách nhập vào:

python app.py

Như vậy, công đoạn khởi chạy ứng dụng thực nghiệm đã hoàn thành. Sau khi chờ đợi Flask thực hiện request để yêu cầu máy chủ xây dựng development server, người dùng sẽ nhận thấy một dòng chứa đường dẫn liên kết có cấu trúc như sau:

★Running on <liên kết http>

Sau đó, người dùng thực hiện nhấn tổ hợp phím **Ctrl+Click** để truy cập vào đường dẫn này, một trang web sẽ hiện lên và người dùng có thể tương tác trực tiếp tại trang web này để chạy ứng dụng thực nghiệm của đề tài.



```
Microsoft Windows [Version 10.0.22631.3296]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd "C:\Users\Admin\Desktop\demo"

C:\Users\Admin\Desktop\demo>python app.py
2024-03-30 17:35:00.211392: I tensorflow/core/util/port.cc:113] oneDNN custom operations are on. You may see slightly different numerical results due to floating-point round-off errors from different computation orders. To turn them off, set the environment variable 'TF_ENABLE_ONEDNN_OPTS=0'.
WARNING:tensorflow:From C:\Users\Admin\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\losses.py:2976: The name tf.losses.sparse_softmax_cross_entropy is deprecated. Please use tf.compat.v1.losses.sparse_softmax_cross_entropy instead.

2024-03-30 17:35:04.352545: I tensorflow/core/platform/cpu_feature_guard.cc:182] This TensorFlow binary is optimized to use available CPU instructions in performance-critical operations. To enable the following instructions: SSE SSE2 SSE3 SSE4.1 SSE4.2 AVX2 AVX512F AVX512_VNNI FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
WARNING:tensorflow:From C:\Users\Admin\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\backend.py:1398: The name tf.executing_eagerly_outside_functions is deprecated. Please use tf.compat.v1.executing_eagerly_outside_functions instead.

* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat http://127.0.0.1:5000
WARNING:tensorflow:From C:\Users\Admin\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\losses.py:2976: The name tf.losses.sparse_softmax_cross_entropy is deprecated. Please use tf.compat.v1.losses.sparse_softmax_cross_entropy instead.

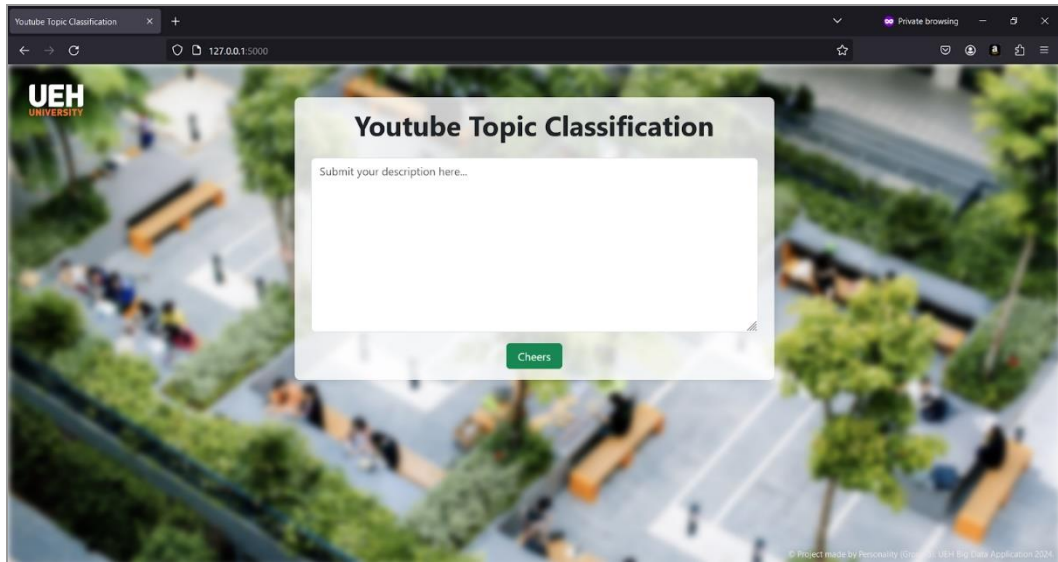
WARNING:tensorflow:From C:\Users\Admin\AppData\Local\Programs\Python\Python311\Lib\site-packages\keras\src\backend.py:1398: The name tf.executing_eagerly_outside_functions is deprecated. Please use tf.compat.v1.executing_eagerly_outside_functions instead.

* Debugger is active!
* Debugger PIN: 340-496-472
```

Hình 33. Thực hiện khởi chạy development server đã xây dựng bằng Flask

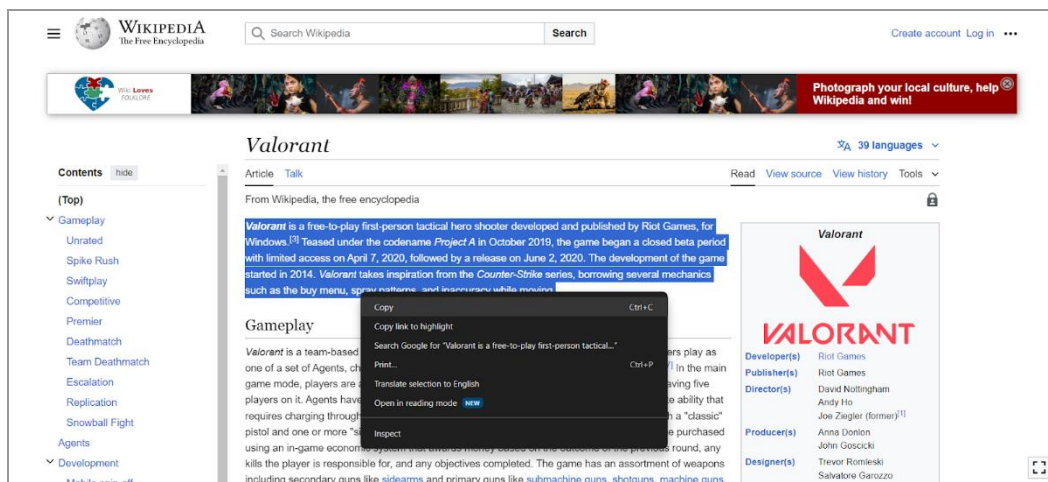
6.2.3 Minh họa kết quả thực hiện

Vậy sau đây, nhóm sẽ thực hiện minh họa sử dụng Demo bằng một câu ngẫu nhiên để kiểm tra xem hiệu suất xử lý của giải thuật này.



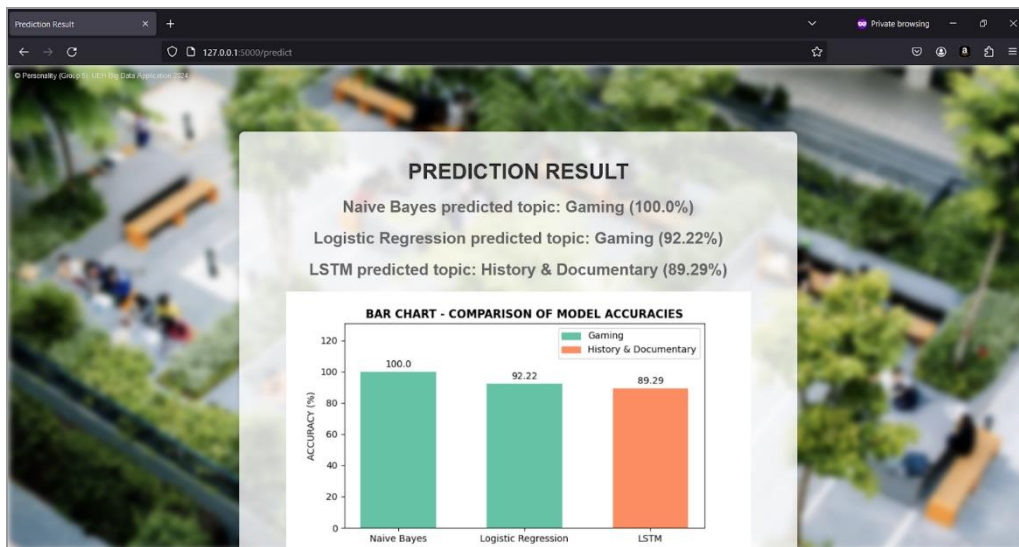
Hình 34. Trang web thực nghiệm ban đầu được xây dựng bằng HTML và CSS

Nhóm sẽ sử dụng Wikipedia cho một chủ đề bất kỳ để input vào ô trống. Nhóm chọn một chủ đề không quá xa lạ trong khoảng thời gian gần đây để mô hình dữ liệu được cào có thể ứng dụng tốt, đó là **Valorant** - một trò chơi bắn súng chiến thuật đối kháng.



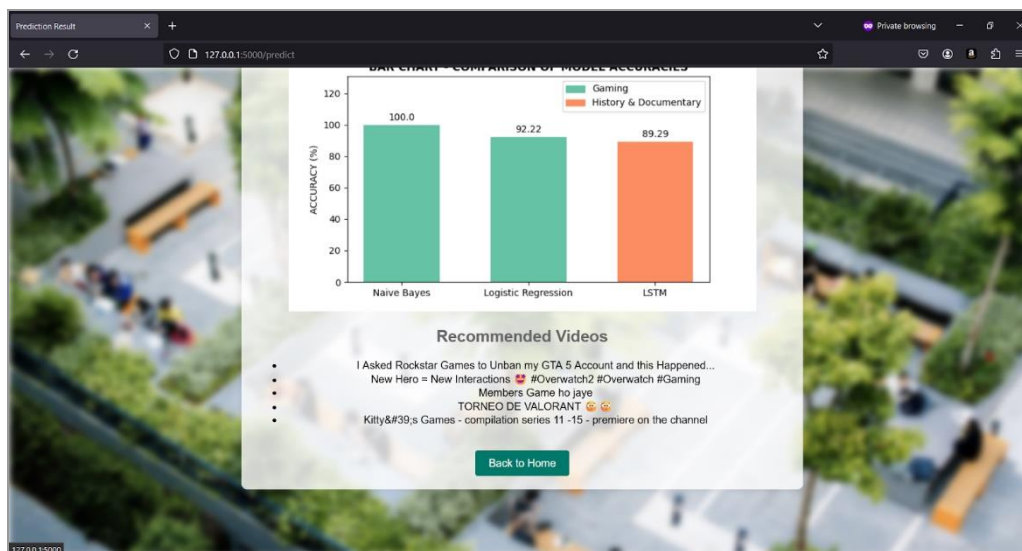
Hình 35. Trang web Wikipedia nội dung đoạn văn nhóm sử dụng để thực nghiệm mô hình

Sau khi input đoạn văn cần phân loại, người dùng sẽ **nhấn nút “Cheers”** để mô hình thực hiện phần backend, lúc này, người dùng sẽ được chuyển qua một trang web mới để hiển thị kết quả của quá trình xử lý phân lớp chủ đề.



Hình 36. Kết quả phân lớp của trang web

Giao diện kết quả này sẽ được nhóm chia làm hai phần chính: Kết quả phân lớp và Hệ thống gợi ý video. Trong phần Recommendation System, nhóm thực hiện sử dụng độ tương đồng Cosin để tính toán các giá trị về sự giống nhau của các video trong tập dữ liệu đề xuất với đoạn mô tả người dùng nhập vào; từ đó, hệ thống này sẽ **so sánh và hiển thị các Tiêu đề Youtube Video khác cùng chủ đề và tương tự với đoạn mô tả được nhập vào**. Dưới đây là năm video Youtube khác trong tập dữ liệu được đề xuất cho người dùng dựa trên quy tắc được đề cập.



Hình 37. Phần nội dung về Hệ thống gợi ý sử dụng Độ tương đồng Cosin

6.2.4 Các lỗi thường gặp và cách khắc phục (Poka-yoke)

6.2.4.1 Lỗi đường dẫn

Vấn đề lỗi đường dẫn là một trường hợp phổ biến khi sử dụng Flask, vì framework này hoạt động dựa trên máy chủ cục bộ. Trong quá trình khởi tạo đường dẫn, có thể xảy ra những vấn đề chủ quan do sự không nhất quán trong cấu trúc thư mục của dự án giữa nhà phát triển ứng dụng (dev) và người dùng cuối (tester/end user).

Nhận biết: Cách để nhận biết lỗi này khi khởi chạy ứng dụng là những dòng báo lỗi không thể tìm thấy thư mục trong tệp tin. Điều này có thể bởi vì các thư mục cha chứa thư mục ứng dụng có những ký tự mà hệ thống không hiểu được.

```
Microsoft Windows [Version 10.0.22631.3296]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin>cd "U:\UEH_lastsem\BDA\endofsem_proj\demo"

C:\Users\Admin>python app.py
python: can't open file 'C:\\Users\\Admin\\app.py': [Errno 2] No such file or directory
```

Hình 38. Ví dụ minh họa về lỗi đường dẫn có thể gặp trong quá trình khởi chạy tệp tin

Để khắc phục lỗi này, người dùng nên thực hiện chuyển toàn bộ thư mục chứa tệp tin ứng dụng ra thư mục Desktop của máy tính cục bộ và thực hiện khởi chạy lại bằng Command Prompt.

6.2.4.2 Lỗi thiếu thư viện

Vấn đề về lỗi thiếu cái thư viện cũng là một trong các trường hợp phổ biến khi khởi chạy các demo thực nghiệm. Danh sách các thư viện sử dụng trong ứng dụng bao gồm:

```
from flask import Flask, render_template, request
from gensim.models import Word2Vec
from keras.models import load_model
from sklearn.metrics.pairwise import cosine_similarity
import pickle, joblib
import matplotlib.pyplot as plt, seaborn as sns
import string, re, pandas as pd, numpy as np, html
```

Nhận biết: Cách để nhận biết lỗi này khi khởi chạy ứng dụng là các dòng lỗi không thể tìm thấy thư viện Python trong hệ thống máy cục bộ.

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19042.1348]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Administrator>cd "C:\Users\Administrator\Desktop\demo\demo"

C:\Users\Administrator\Desktop\demo\demo>python app.py
Traceback (most recent call last):
  File "C:\Users\Administrator\Desktop\demo\demo\app.py", line 1, in <module>
    from flask import Flask, render_template, request
ModuleNotFoundError: No module named 'flask'
```

Hình 39. Ví dụ minh họa về lỗi thiếu thư viện có thể gặp trong quá trình khởi chạy tệp tin

Vậy để khắc phục lỗi này, người dùng chỉ cần chắc chắn đã cài đặt tất cả các thư viện cần thiết để khởi chạy ứng dụng. Nếu chưa, người dùng có thể thực hiện nhập vào Command Prompt:

```
python -m pip install <thư viện>
```

CHƯƠNG 7: KẾT LUẬN VÀ ĐÁNH GIÁ

7.1 Kết quả đạt được

Trong khuôn khổ của đề tài 'Phân tích, dự đoán chủ đề của các video trên nền tảng Youtube, xây dựng hệ thống gợi ý video', nhóm đã đạt được những kết quả đáng kể sau:

- Triển khai phương pháp khai thác dữ liệu lớn của từ nền tảng Youtube một cách hiệu quả. Thu thập được nhiều thông tin về các video trên Youtube về nhiều chủ đề đa dạng với kích thước dữ liệu phù hợp để huấn luyện các mô hình.
- Tiền xử lý cụ thể để chuẩn bị cho các hướng phát triển sau đó như phân tích dự đoán chủ đề, xây dựng hệ thống gợi ý nội dung. Trích xuất được cơ bản những thông tin, insights có giá trị từ bộ dữ liệu để có cái nhìn rõ hơn về bộ dữ liệu.
- Phát triển một thuật toán phân tích dữ liệu, xử lý ngôn ngữ tự nhiên có khả năng xử lý dữ liệu văn bản từ Tiêu đề và Mô tả của các video Youtube, đảm bảo đầu vào phù hợp phục vụ cho mô hình dự đoán chủ đề video.
- Xây dựng một mô hình dự đoán chủ đề video dựa trên các thuật toán máy học, mô hình học sâu, cho phép dự đoán chủ đề của video với độ chính xác tương đối tốt.
- Tạo ra một hệ thống gợi ý video động, có khả năng tự học hỏi và thích ứng với sở thích và hành vi xem của người dùng.
- Phát triển một demo giao diện người dùng thân thiện, dễ tương tác với mục đích minh họa trực quan hiệu suất của các mô hình vừa xây dựng.

Những kết quả này không chỉ là nỗ lực của nhóm trong việc khai thác dữ liệu lớn và ứng dụng của chúng, mà còn mở ra hướng phát triển mới cho các nghiên cứu tiếp theo trong lĩnh vực phân tích dữ liệu và hệ thống gợi ý. Đề án này giúp chúng tôi có được những trải nghiệm về những ưu, nhược điểm của các phương án thu thập dữ liệu lớn trên nền tảng Youtube, đồng thời đạt được một số hiểu biết nhất định về những đặc trưng của dữ liệu từ Youtube. Bên cạnh đó đề tài cũng là cơ hội vận dụng tổng hợp những kiến thức và kỹ thuật đã học để khai phá những ứng dụng tiềm năng của dữ liệu lớn trong lĩnh vực xử lý ngôn ngữ tự nhiên và cải thiện trải nghiệm liền mạch cho người dùng của một nền tảng phương tiện truyền thông xã hội.

7.2 Hạn chế

Khi sử dụng API của YouTube để cào dữ liệu, nhóm nhận thấy hạn chế về số lượng dữ liệu thu thập được, cụ thể là giới hạn trong việc truy cập thông tin về video.

Các bước tiền xử lý dữ liệu có thể không đủ để loại bỏ hoặc xử lý triệt để các dữ liệu nhiễu, dẫn đến ảnh hưởng chất lượng của mô hình. Đồng thời, khám phá dữ liệu chỉ tập trung vào một số thông tin cụ thể và có thể bỏ qua những thông tin quan trọng khác.

Mặc dù mô hình dự đoán chủ đề video đạt được độ chính xác tương đối tốt, nhưng vẫn có thể gặp phải sai số và không đồng nhất trong việc dự đoán chủ đề cho một số video.

Các thuật toán máy học và học sâu đôi khi không đủ linh hoạt để xử lý đầy đủ các biến thể và đặc điểm của dữ liệu từ YouTube, dẫn đến sự hạn chế trong hiệu suất của mô hình.

Tuy hệ thống gợi ý video được mô tả có khả năng tự học hỏi và thích ứng với sở thích và hành vi xem của người dùng, nhưng vẫn có thể gặp phải sự hạn chế trong việc hiểu và dự đoán đúng theo sở thích và hành vi của người dùng.

7.3 Các phương pháp cải tiến đề xuất

Thông qua quá trình thực hiện phân tích trên bộ dữ liệu cũng như tham khảo qua các trang điện tử, nhóm chúng em tiến hành đưa ra một số phương pháp cải tiến:

- Đa dạng hóa nghiên cứu và áp dụng: Có thể xem xét mở rộng phạm vi nghiên cứu để áp dụng hệ thống gợi ý video vào các nền tảng khác ngoài YouTube. Nghiên cứu về tính ứng dụng của hệ thống gợi ý video trong các lĩnh vực cụ thể như giáo dục trực tuyến, marketing số, ...
- Nghiên cứu các phương pháp và kỹ thuật để xử lý dữ liệu độc lập và đảm bảo tính riêng tư của người dùng trong quá trình xây dựng hệ thống. Sử dụng kỹ thuật gom cụm (clustering) và giảm chiều dữ liệu (dimensionality reduction) để trích xuất thông tin ẩn tiềm năng từ dữ liệu video một cách hiệu quả hơn.
- Xây dựng một mô hình đa tác vụ có khả năng dự đoán nhiều thuộc tính của video đồng thời như chủ đề, ngôn ngữ, hoặc độ tuổi đối tượng, từ đó cải thiện hiệu suất và tính đa dạng của mô hình.
- Tích hợp các phương pháp và kỹ thuật tương tác người-máy (HCI) để cải thiện trải nghiệm người dùng khi sử dụng hệ thống gợi ý video. Xây dựng hệ thống có khả năng đề xuất video dựa trên nhận dạng và đánh giá cảm xúc của người dùng, từ đó cải thiện trải nghiệm xem video của người dùng.
- Nghiên cứu về ứng dụng công nghệ trí tuệ nhân tạo và kỹ thuật xử lý ngôn ngữ tự nhiên tiên tiến để cải thiện hiệu suất và chất lượng của hệ thống gợi ý video.

Nếu có cơ hội tiếp tục thực hiện đề tài này trong tương lai, nhóm hy vọng sẽ có thêm nguồn lực và thời gian để triển khai các hướng đề xuất, từ việc áp dụng mô hình học sâu tiên tiến đến việc tăng cường dữ liệu huấn luyện và tích hợp kiến thức bên ngoài. Điều này sẽ giúp tạo ra một hệ thống gợi ý video linh hoạt và đáng tin cậy hơn, mang lại trải nghiệm tốt nhất cho người dùng trên nền tảng truyền thông xã hội.

PHỤ LỤC

BẢNG PHÂN CÔNG

Họ và tên	Phân công	Đánh giá
Trần Phạm Hải Nam	Cào dữ liệu; Mô tả tổng quan bộ dữ liệu và phương pháp cào (Chương 3); Xây dựng ứng dụng Demo Web bằng HTML/CSS.	100%
Lý Minh Nguyên	Cào dữ liệu; Tổng quan lý thuyết (Chương 2); Xây dựng mô hình máy học Naive Bayes và Logistic Regression; Xây dựng Hệ thống gợi ý; Tổng hợp mã nguồn.	100%
Trần Duy Tuấn	Cào dữ liệu; Thực hiện Khám phá Phân tích dữ liệu EDA (Chương 4); Xây dựng kết luận về hạn chế và phương pháp cải tiến; Thiết kế slide thuyết trình.	100%
Nguyễn Nhật Thảo Vy	Cào dữ liệu; Thực hiện Tổng quan đề tài nghiên cứu (Chương 1); Xây dựng mô hình học sâu LSTM; Xây dựng kết luận về các kết quả đạt được từ nghiên cứu.	100%

TÀI LIỆU THAM KHẢO

- [1] Bài 6: Logistic Regression (Hồi quy Logistic). (2023, October 13). Trí tuệ nhân tạo. Retrieved March 20, 2024, from <https://trituenhantao.io/machine-learning-co-ban/bai-6-logistic-regression-hoi-quy-logistic/>
- [2] Chung Pham Van. (2021, January 13). [Recommender System] Giới thiệu hệ thống gợi ý. Viblo. Retrieved March 26, 2024, from <https://viblo.asia/p/recommender-system-gioi-thieu-he-thong-goi-y-gAm5yJPqKdb>
- [3] Đinh Hoàng. (2021, May 20). Tổng quan về Recommender System [Recommender System cơ bản - Phần 1]. Viblo. Retrieved March 26, 2024, from <https://viblo.asia/p/tong-quan-ve-recommender-system-recommender-system-co-ban-phan-1-924IJGBb5PM>
- [4] Hamdaoui, Y. (2024, February 8). TF-IDF: An Introduction. Built In. Retrieved March 20, 2024, from <https://builtin.com/articles/tf-idf>
- [5] Kiên Nguyễn. (n.d.). Flask python là gì? - Những điều cần biết. TopDev. Retrieved March 26, 2024, from <https://topdev.vn/blog/flask-python-la-gi-nhung-dieu-can-biet/>
- [6] Liên Nhựt. (2024, February 9). Flask là gì? Cập nhật những kiến thức cơ bản về Web Framework của Python mà bạn cần biết. FPT Shop. Retrieved March 26, 2024, from <https://fptshop.com.vn/tin-tuc/danh-gia/flask-la-gi-175382>
- [7] Nguyen, H. T. (2019). Thuật toán phân lớp Naive Bayes. Viblo. Retrieved March 20, 2024, from <https://viblo.asia/p/thuat-toan-phan-lop-naive-bayes-924IJWPm5PM>
- [8] Saxena, S. (2021, March 17). What is LSTM? Introduction to Long Short-Term Memory. Analytics Vidhya. Retrieved March 20, 2024, from <https://www.analyticsvidhya.com/blog/2021/03/introduction-to-long-short-term-memory-lstm/>
- [9] Topper, N. (2023, August 2). Bag of Words Model in NLP Explained. Built In. Retrieved March 20, 2024, from <https://builtin.com/machine-learning/bag-of-words>
- [10] YouTube Data API Overview. (2012, June 20). Google for Developers. Retrieved March 10, 2024, from <https://developers.google.com/youtube/v3/getting-started>