



20211027 MIS Xu Ly Ngon Ngu Tu Nhien CK NHOM 1 V1

Management Information Systems (Trường Đại học Kinh tế Thành phố Hồ Chí Minh)



Scan to open on Studocu

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC KINH TẾ TP. HỒ CHÍ MINH**



NHÓM 1

**ỨNG DỤNG XỬ LÝ NGÔN NGỮ TỰ NHIÊN
TRONG PHÂN TÍCH SỰ ƯA THÍCH CỦA KHÁCH HÀNG ĐỐI VỚI
QUẢNG CÁO BỘT GIẶT ABBA**

**BÀI TẬP CUỐI KHÓA
MÔN XỬ LÝ NGÔN NGỮ TỰ NHIÊN**

TP. HỒ CHÍ MINH - NĂM 2021

**BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC KINH TẾ TP. HỒ CHÍ MINH**

NHÓM 1

**ỨNG DỤNG XỬ LÝ NGÔN NGỮ TỰ NHIÊN
TRONG PHÂN TÍCH SỰ ƯA THÍCH CỦA KHÁCH HÀNG ĐỐI VỚI
QUẢNG CÁO BỘT GIẶT ABBA**

**BÀI TẬP CUỐI KHÓA
MÔN XỬ LÝ NGÔN NGỮ TỰ NHIÊN**

Người hướng dẫn khoa học: Tiến sĩ Đỗ Trọng Hợp

1. *Trần Sơn Nam - 202118009*
2. *Phạm Văn Long - 202118008*
3. *Nguyễn Văn Đức - 202118003*
4. *Ngô Thị Diệu Quỳnh - 202118014*
5. *Nguyễn Ngọc Châu Uyên - 202118021*

TP. HỒ CHÍ MINH - NĂM 2021

LỜI CẢM ƠN



Để hoàn thành bài tập nhóm này, các thành viên nhóm chân thành gửi lời cảm ơn đến thầy Đỗ Trọng Hợp đã giúp đỡ và hỗ trợ cho Nhóm được tiếp cận các kiến thức mới trong lĩnh vực Xử lý ngôn ngữ tự nhiên. Đây là một lĩnh vực hợp với xu hướng phát triển ngày nay với rất nhiều thông tin dữ liệu liên tục cập nhật trên các nền tảng mạng và máy tính cần phải hiểu để có khả năng phân tích, xử lý và hiểu được ngôn ngữ của con người. Với công nghệ xử lý ngôn ngữ tự nhiên sẽ giúp cho Doanh nghiệp tăng tính cạnh tranh thông qua hỗ trợ trong điều hành doanh nghiệp và tăng sự trải nghiệm đối với khách hàng. Với kiến thức của môn học này mà nhóm lĩnh hội được từ Thầy đã giúp cho nhóm có thêm kiến thức và tự tin hơn khi vận dụng kiến thức này vào thực tế trong công việc.

Kính chúc Thầy cùng Quý Nhà trường luôn mạnh khỏe, và Thành công trong công tác giảng dạy.

Xin chân thành cảm ơn!

MỤC LỤC

□□

LỜI NÓI ĐẦU.....	1
CHƯƠNG 1.....	2
LÝ THUYẾT XỬ LÝ NGÔN NGỮ TỰ NHIÊN.....	2
1.1 XỬ LÝ NGÔN NGỮ TỰ NHIÊN.....	2
1.1.1 ĐỊNH NGHĨA.....	2
1.1.2 QUY TRÌNH XỬ LÝ NGÔN NGỮ TỰ NHIÊN.....	2
1.1.3 ỨNG DỤNG CỦA XỬ LÝ NGÔN NGỮ TỰ NHIÊN.....	3
CHƯƠNG 2.....	6
ỨNG DỤNG XỬ LÝ NGÔN NGỮ TỰ NHIÊN TRONG MÁY HỌC.....	6
2.1 ĐỊNH NGHĨA CHUNG.....	6
2.2 XÂY DỰNG TỪ ĐIỂN TIẾNG VIỆT TRONG MÁY TÍNH.....	7
2.3.1 Thu thập ngữ liệu.....	7
2.3.2 Chuẩn hoá ngữ liệu.....	8
2.3.3 Gán nhãn ngữ liệu.....	8
2.3.4 Xây dựng kho tư liệu.....	10
2.3 CÁC THUẬT TOÁN HÀM ỨNG DỤNG.....	11
2.3.1 Thuật toán support vector machine.....	11
2.3.2 Thuật toán Bayes.....	14
2.3.2.1 Định lý Bayes.....	15
2.3.2.2 Bài toán phân loại Bắc hay Nam và thuật toán Bayes.....	16
CHƯƠNG 3.....	18
PHÂN TÍCH BÌNH LUẬN QUẢNG CÁO BỘT GIẶT ABBA TRÊN KÊNH YOUTUBE.....	18
2.1 GIỚI THIỆU VỀ BỘT GIẶT ABBA.....	18
2.2 MÔ TẢ PHƯƠNG THỨC LẤY DỮ LIỆU CỦA VIDEO QUẢNG CÁO BỘT GIẶT ABBA TRÊN YOUTUBE.....	19

2.3.1	Phân Tích Cảm Xúc.....	19
2.2.2.1	Phân tích cảm xúc tiếp cận theo xử lý ngôn ngữ tự nhiên.....	19
2.2.2.2	Phân tích cảm xúc tiếp cận theo phương pháp Học máy.....	20
2.2.2.3	Thuật toán Hồi quy Logistic.....	21
2.3.2	Bài toán thực nghiệm.....	23
2.2.2.1	Tổng quan.....	23
2.2.2.2	Thu thập dữ liệu.....	24
2.2.2.3	Tiền xử lý dữ liệu.....	25
2.2.2.4	Tiền Xử Lý Và Làm Sạch Dữ Liệu.....	26
2.2.2.5	Xóa URL.....	26
2.2.2.6	Xóa Email.....	26
2.2.2.7	Xóa SĐT.....	26
2.2.2.8	Xóa \n.....	26
2.2.2.9	Xóa Timestamp.....	27
2.2.2.10	Xóa Ký tự đặc biệt.....	27
2.2.2.11	Xóa Khoảng trắng thừa.....	27
2.2.2.12	Xóa row trống.....	27
2.2.2.13	Lowercase.....	27
2.2.2.14	Chuẩn hóa Unicode.....	27
2.2.2.15	Lưu file.....	30
2.3.3	Gán nhãn dữ liệu.....	30
2.3.4	Phương pháp biểu diễn văn bản.....	31
2.3.5	Chạy Thuật Toán Và Mô Hình.....	33
2.3.6	Tách từ.....	33
2.3.7	Xóa stopwords.....	33
2.3.8	WordCloud.....	34
2.3.9	Phân loại cảm xúc.....	34
2.3.10	Kết quả thực nghiệm :.....	35

2.2.2.1	Đánh giá mô hình.....	35
2.4	Trực quan hóa và đưa qua kết luận.....	36
2.4.1	Các từ phổ biến.....	37
2.4.2	WordCloud.....	37
2.4.3	Phân loại cảm xúc.....	37
KẾT LUẬN.....		38
DANH MỤC TÀI LIỆU.....		39

...

LỜI NÓI ĐẦU

Xử lý ngôn ngữ tự nhiên, một nhánh nghiên cứu của trí tuệ nhân tạo, được phát triển nhằm xây dựng các chương trình máy tính có khả năng phân tích, xử lý, và hiểu ngôn ngữ con người. Công nghệ này đã và đang mang lại những ứng dụng hỗ trợ thiết thực trong các hoạt động vận hành doanh nghiệp cũng như nâng cao trải nghiệm khách hàng.

Một trong những mong muốn mãnh liệt, xuất hiện từ rất sớm của các nhà khoa học máy tính (computer science) nói chung và trí tuệ nhân tạo (artificial intelligence) nói riêng là xây dựng thành công các hệ thống, chương trình máy tính có khả năng giao tiếp với con người thông qua ngôn ngữ tự nhiên (natural language), tức thứ ngôn ngữ con người sử dụng hàng ngày thay vì các ngôn ngữ lập trình (programming language) hay ngôn ngữ máy (computer language) bậc thấp. Xử lý ngôn ngữ tự nhiên (natural language processing), một nhánh nghiên cứu của trí tuệ nhân tạo, trong đó phát triển các thuật toán, xây dựng các chương trình máy tính có khả năng phân tích, xử lý, và hiểu ngôn ngữ của con người, chính là lĩnh vực nhằm hiện thực hóa mục tiêu này. Do đó ngay từ khi trí tuệ nhân tạo mới ra đời (năm 1956), các nhà nghiên cứu đã đặt xử lý ngôn ngữ tự nhiên là một trong hai nhiệm vụ trọng tâm của trí tuệ nhân tạo, bên cạnh việc phát triển các chương trình máy tính có khả năng chiến thắng con người trong các trò chơi trí tuệ đối kháng. Vì vậy, nhóm em xin chọn đề tài **“ỨNG DỤNG XỬ LÝ NGÔN NGỮ TỰ NHIÊN TRONG PHÂN TÍCH SỰ ƯA THÍCH CỦA KHÁCH HÀNG ĐỐI VỚI QUẢNG CÁO BỘT GIẶT ABBA”** để giới thiệu về lĩnh vực xử lý ngôn ngữ tự nhiên, các bước cơ bản trong xử lý ngôn ngữ tự nhiên, một số ứng dụng của xử lý ngôn ngữ tự nhiên, và cách thức công nghệ này giúp máy tính giao tiếp với con người.

CHƯƠNG 1

LÝ THUYẾT XỬ LÝ NGÔN NGỮ TỰ NHIÊN

1.1 XỬ LÝ NGÔN NGỮ TỰ NHIÊN

1.1.1 ĐỊNH NGHĨA

Xử lý ngôn ngữ tự nhiên (*Natural Language processing - NLP*) là một nhánh của Trí tuệ nhân tạo, tập trung vào việc nghiên cứu sự tương tác giữa máy tính và ngôn ngữ tự nhiên của con người, dưới dạng tiếng nói (*speech*) hoặc văn bản (*text*). Mục tiêu của lĩnh vực này là giúp máy tính hiểu và thực hiện hiệu quả những nhiệm vụ liên quan đến ngôn ngữ của con người như: tương tác giữa người và máy, cải thiện hiệu quả giao tiếp giữa con người với con người, hoặc đơn giản là nâng cao hiệu quả xử lý văn bản và lời nói.

1.1.2 QUY TRÌNH XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Gồm 4 bước chính sau:

Phân tích hình vị: là sự nhận biết, phân tích, và miêu tả cấu trúc của hình vị trong một ngôn ngữ cho trước và các đơn vị ngôn ngữ khác, như từ gốc, biên từ, phụ tố, từ loại, v.v. Trong xử lý tiếng Việt, hai bài toán điển hình trong phần này là tách từ (*word segmentation*) và gán nhãn từ loại (*part-of-speech tagging*).

Phân tích cú pháp: là quy trình phân tích một chuỗi các biểu tượng, ở dạng ngôn ngữ tự nhiên hoặc ngôn ngữ máy tính, tuân theo văn phạm hình thức. Văn phạm hình thức thường dùng trong phân tích cú pháp của ngôn ngữ tự nhiên bao gồm Văn phạm phi ngữ cảnh (*Context-free grammar – CFG*), Văn phạm danh mục kết nối (*Combinatory categorial grammar – CCG*), và Văn phạm phụ thuộc (*Dependency grammar – DG*). Đầu vào của quá trình phân tích là một câu gồm một chuỗi từ và nhãn từ loại của chúng, và đầu ra là một cây phân tích thể hiện cấu trúc cú pháp của câu đó.

Phân tích ngữ nghĩa: là quá trình liên hệ cấu trúc ngữ nghĩa, từ cấp độ cụm từ, mệnh đề, câu và đoạn đến cấp độ toàn bài viết, với ý nghĩa độc lập của chúng. Nói cách khác, việc này nhằm tìm ra ngữ nghĩa của đầu vào ngôn từ. Phân tích ngữ nghĩa bao gồm hai mức độ: Ngữ nghĩa từ vựng biểu hiện các ý nghĩa của những từ thành phần, và phân biệt nghĩa của từ; Ngữ nghĩa thành phần liên quan đến cách thức các từ liên kết để hình thành những nghĩa rộng hơn.

Phân tích diễn ngôn: là phân tích văn bản có xét tới mối quan hệ giữa ngôn ngữ và ngữ cảnh sử dụng (*context-of-use*). Phân tích diễn ngôn, do đó, được thực hiện ở mức độ đoạn văn hoặc toàn bộ văn bản thay vì chỉ phân tích riêng ở mức câu.

1.1.3 ỨNG DỤNG CỦA XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Xử lý ngôn ngữ tự nhiên ngày càng được ứng dụng nhiều. Một số ứng dụng có thể kể đến như:

Nhận dạng tiếng nói (*Automatic Speech Recognition – ASR*, hoặc *Speech To Text – STT*) chuyển đổi ngôn ngữ từ dạng tiếng nói sang dạng văn bản, thường được ứng dụng trong các chương trình điều khiển qua giọng nói.

Tổng hợp tiếng nói (*Speech synthesis hoặc Text to Speech – TTS*) chuyển đổi ngôn ngữ từ dạng văn bản sang tiếng nói, thường được dùng trong đọc văn bản tự động.

Truy xuất thông tin (*Information Retrieval – IR*) có nhiệm vụ tìm các tài liệu dưới dạng không có cấu trúc (*thường là văn bản*) đáp ứng nhu cầu về thông tin từ những nguồn tổng hợp lớn. Những hệ thống truy xuất thông tin phổ biến nhất bao gồm các công cụ tìm kiếm như Google, Yahoo, hoặc Bing search. Những công cụ này cho phép tiếp nhận một câu truy vấn dưới dạng ngôn ngữ tự nhiên làm đầu vào và cho ra một danh sách các tài liệu được sắp xếp theo mức độ phù hợp.

Trích chọn thông tin (*Information Extraction – IE*) nhận diện một số loại thực thể được xác định trước, mối quan hệ giữa các thực thể và các sự kiện trong văn bản ngôn ngữ tự nhiên. Khác với truy xuất thông tin trả về một danh sách các văn bản hợp lệ thì trích chọn thông tin trả về chính xác thông tin mà người dùng cần. Những thông tin này có thể là về con người, địa điểm, tổ chức, ngày tháng, hoặc thậm chí tên công ty, mẫu sản phẩm hay giá cả.

Trả lời câu hỏi (*Question Answering – QA*) có khả năng tự động trả lời câu hỏi của con người ở dạng ngôn ngữ tự nhiên bằng cách truy xuất thông tin từ một tập hợp tài liệu. Một hệ thống QA đặc trưng thường bao gồm ba mô đun: Mô đun xử lý truy vấn (*Query Processing Module*) – tiến hành phân loại câu hỏi và mở rộng truy vấn; Mô đun xử lý tài liệu (*Document Processing Module*) – tiến hành truy xuất thông tin để tìm ra tài liệu thích hợp; và Mô hình xử lý câu trả lời (*Answer Processing Module*) – trích chọn câu trả lời từ tài liệu đã được truy xuất.

Tóm tắt văn bản tự động (*Automatic Text Summarization*) là bài toán thu gọn văn bản đầu vào để cho ra một bản tóm tắt ngắn gọn với những nội dung quan trọng nhất của văn bản gốc. Có hai phương pháp chính trong tóm tắt, là phương pháp trích xuất (*extractive*) và phương pháp tóm lược ý (*abstractive*). Những bản tóm tắt trích xuất được hình thành bằng cách ghép một số câu được lấy y nguyên từ văn bản cần thu gọn. Những bản tóm lược ý thường truyền đạt những thông tin chính của đầu vào và có thể sử dụng lại những cụm từ hay mệnh đề trong đó, nhưng nhìn chung được thể hiện ở ngôn ngữ của người tóm tắt.

Chatbot là việc chương trình máy tính có khả năng trò chuyện (*chat*), hỏi đáp với con người qua hình thức hội thoại dưới dạng văn bản (*text*). Chatbot thường được sử dụng trong ứng dụng hỗ trợ khách hàng, giúp người dùng tìm kiếm thông tin sản phẩm, hoặc giải đáp thắc mắc.

Dịch máy (*Machine Translation – MT*) là việc sử dụng máy tính để tự động hóa một phần hoặc toàn bộ quá trình dịch từ ngôn ngữ này sang ngôn ngữ khác. Các phương pháp dịch máy phổ biến bao gồm dịch máy dựa trên ví dụ (*example-based machine translation – EBMT*), dịch máy dựa trên luật (*rule-based machine translation – RBMT*), dịch máy thống kê (*statistical machine translation – SMT*), và dịch máy sử dụng mạng nơ-ron (*neural machine translation*).

Kiểm lỗi chính tả tự động là việc sử dụng máy tính để tự động phát hiện các lỗi chính tả trong văn bản (*lỗi từ vựng, lỗi ngữ pháp, lỗi ngữ nghĩa*) và đưa ra gợi ý cách chỉnh sửa lỗi.

CHƯƠNG 2

ỨNG DỤNG XỬ LÝ NGÔN NGỮ TỰ NHIÊN TRONG MÁY HỌC

2.1 ĐỊNH NGHĨA CHUNG

Xử lý ngôn ngữ tự nhiên là một nhánh của Trí tuệ nhân tạo, tập trung vào việc nghiên cứu sự tương tác giữa máy tính và ngôn ngữ tự nhiên của con người, dưới dạng tiếng nói (*speech*) hoặc văn bản (*text*). Mục tiêu của lĩnh vực này là giúp máy tính hiểu và thực hiện hiệu quả những nhiệm vụ liên quan đến ngôn ngữ của con người như: tương tác giữa người và máy, cải thiện hiệu quả giao tiếp giữa con người với con người, hoặc đơn giản là nâng cao hiệu quả xử lý văn bản và lời nói.

Xử lý ngôn ngữ tự nhiên ra đời từ những năm 40 của thế kỷ 20, trải qua các giai đoạn phát triển với nhiều phương pháp và mô hình xử lý khác nhau. Có thể kể tới các phương pháp sử dụng ô-tô-mát và mô hình xác suất (*những năm 50*), các phương pháp dựa trên ký hiệu, các phương pháp ngẫu nhiên (*những năm 70*), các phương pháp sử dụng học máy truyền thống (*những năm đầu thế kỷ 21*), và đặc biệt là sự bùng nổ của học sâu trong thập kỷ vừa qua.

Xử lý ngôn ngữ tự nhiên có thể được chia ra thành hai nhánh lớn, không hoàn toàn độc lập, bao gồm xử lý tiếng nói (*speech processing*) và xử lý văn bản (*text processing*). Xử lý tiếng nói tập trung nghiên cứu, phát triển các thuật toán, chương trình máy tính xử lý ngôn ngữ của con người ở dạng tiếng nói (*dữ liệu âm thanh*). Các ứng dụng quan trọng của xử lý tiếng nói bao gồm nhận dạng tiếng nói và tổng hợp tiếng nói. Nếu như nhận dạng tiếng nói là chuyển ngôn ngữ từ dạng tiếng nói sang dạng văn bản thì ngược lại, tổng hợp tiếng nói chuyển ngôn ngữ từ dạng văn bản thành tiếng nói. Xử lý văn bản tập trung vào phân tích dữ liệu văn bản. Các ứng dụng quan trọng của xử lý văn bản bao gồm tìm kiếm và truy xuất thông tin, dịch máy, tóm tắt văn

bản tự động, hay kiểm lỗi chính tả tự động. Xử lý văn bản đôi khi được chia tiếp thành hai nhánh nhỏ hơn bao gồm hiểu văn bản và sinh văn bản. Nếu như hiểu liên quan tới các bài toán phân tích văn bản thì sinh liên quan tới nhiệm vụ tạo ra văn bản mới như trong các ứng dụng về dịch máy hoặc tóm tắt văn bản tự động.

2.2 XÂY DỰNG TỪ ĐIỂN TIẾNG VIỆT TRONG MÁY TÍNH

Thuật ngữ “ngữ liệu” ở đây được tạm dịch từ thuật ngữ tiếng Anh “corpus” (*danh từ số nhiều là “corpora”*) và trong tiếng Hi Lạp có nghĩa là “thân thể” (*body*). Nghĩa của từ “corpus” được hiểu là “*phần thân của văn bản*” và là tập hợp của nhiều văn bản. Chính vì vậy, từ “corpus” được dịch là “kho dữ liệu, kho sưu tập tài liệu,...”. Ngữ liệu là những “*dữ liệu, cứ liệu của ngôn ngữ*”, tức là những chứng cứ thực tế sử dụng ngôn ngữ. Những chứng cứ sử dụng ngôn ngữ này có thể là của ngôn ngữ nói (*spoken language*) hoặc ngôn ngữ viết (*written language*). Ngữ liệu của ngôn ngữ viết thường được hiểu là tập hợp các văn bản. Ngữ liệu này có thể tồn tại dưới nhiều dạng khác nhau: như dạng giấy (*hardcopy*), dạng điện tử (*softcopy*) và hiện nay, các nhà ngôn ngữ học – ngữ liệu chủ yếu xét đến ngữ liệu dạng điện tử. Ngữ liệu có thể là ngữ liệu đơn ngữ (*monolingual corpus*) hoặc ngữ liệu đa ngữ (*multilingual corpus*) và nói chung có 2 dạng: dạng chỉ gồm các ngữ liệu thô thu được, không có chú thích (*unannotated corpus*) và dạng có chú thích (*annotated corpus*) thêm thông tin về ngôn ngữ cho các đơn vị ngôn ngữ trong ngữ liệu thô đó. Dạng thứ nhất thì dễ xây dựng (*vì thường có sẵn*) còn dạng thứ hai thì chúng ta phải tốn nhiều công sức và thời gian để gán thêm thông tin về ngôn ngữ cho ngữ liệu thô để sau này khai thác thông tin được hiệu quả hơn.

2.3.1 Thu thập ngữ liệu

Hiện nay, với sự ra đời của máy tính điện tử, thì việc thu thập ngữ liệu đã được tự động hoá rất nhiều. Hầu như người ta không còn cần phải gõ lại các ngữ liệu, vì phần lớn các ngữ liệu đó đã tồn tại dưới dạng điện tử (như trên Internet), người ta chỉ việc tổ chức, sắp xếp lại theo mục đích nghiên cứu. Hiện nay các nhà ngôn ngữ học máy tính chủ yếu dựa vào một số ngữ liệu chuẩn đã được chọn lọc kỹ lưỡng từ các văn bản chuẩn, như bên tiếng Anh thì dùng Wall Street Journal trong PTB [Mitchell 1993], Brown, BNC, ANC, OEC... Trong tiếng Việt cũng đã có rất nhiều trang mạng để có thể trích xuất ngữ liệu. Trong quá trình thu thập, các nhà ngôn ngữ học – máy tính phải tuân theo các tiêu chí nhất quán trong việc lấy mẫu ngữ liệu, như: chủng loại (*ngôn ngữ nói/ngôn ngữ viết, báo chí, sách vở, tạp chí, phim ảnh, chat, blog, văn bản pháp quy,...*) của ngữ liệu, lĩnh vực của ngữ liệu, tỉ lệ giữa các lĩnh vực, phương ngữ của ngữ liệu, độ lớn của ngữ liệu và độ dài từng văn bản trong kho ngữ liệu, thời gian lưu hành của ngữ liệu, đặc điểm (*tuổi tác, nghề nghiệp, giới tính,...*) về tác giả của ngữ liệu,... Chẳng hạn kho ngữ liệu OEC (*Oxford English Corpus*) có chứa 190 triệu từ từ các bản tin tức.

Một điều mà chúng ta cũng cần chú ý khi thu thập ngữ liệu là tính hợp pháp của việc thu thập đó (*bản quyền, giấy phép, tính riêng tư,...*).

2.3.2 Chuẩn hoá ngữ liệu

Do ngữ liệu được thu thập từ nhiều nguồn khác nhau, nên chúng ta nhất thiết phải có công đoạn chuẩn hoá ngữ liệu để đưa về một dạng thống nhất cho dễ xử lý tự động. Việc chuẩn hoá ngữ liệu gồm các nhiệm vụ chính như: đưa về đúng dạng điện tử, định dạng tập tin (*TXT, SGML, XML*), mã/font (*unicode, utf-8*), chuẩn chính tả (*bao gồm cả việc ghi dấu phụ, biến thể,...*). Việc kiểm tra tính chuẩn của ngữ liệu được thực hiện tự động bởi chương trình máy tính, còn việc kiểm lỗi chính tả cũng được người thực hiện với sự trợ giúp của chương trình máy tính.

2.3.3 Gán nhãn ngữ liệu

Mục tiêu của việc xây dựng kho ngữ liệu là để từ đó có thể khai thác phục vụ cho các mục đích nghiên cứu khác nhau. Để có thể khai thác hiệu quả, nhất thiết kho ngữ liệu đó phải được gán các thông tin về ngôn ngữ (*như: hình thái, ngữ pháp, ngữ nghĩa, ...*) mà các nhà ngôn ngữ học - ngữ liệu gọi là nhãn ngôn ngữ. Hệ thống nhãn ngôn ngữ bao gồm các nhãn về hình thái, ngữ pháp (*từ pháp*) và ngữ nghĩa của từ, ngữ và câu. Nhãn hình thái ở đây bao gồm các nhãn về ranh giới của từ, ranh giới ngữ và ranh giới câu. Nhãn hình thái từ cũng bao gồm các trường hợp viết tắt, tỉnh lược. Đối với các tiếng đơn lập như tiếng Việt, việc xác định ranh giới từ không phải là chuyện đơn giản. Nhãn ngữ pháp ở đây bao gồm các nhãn phân loại căn cứ theo mặt ngữ pháp của từ (*hay còn gọi là từ pháp*), ngữ pháp của ngữ và ngữ pháp của câu (*cú pháp*). Cụ thể bao gồm hai phạm trù ngữ pháp của từ như sau: phạm trù phân loại từ và phạm trù ngữ pháp biến đổi từ. Phạm trù phân loại từ là một phạm trù ngữ pháp chung, bao gồm việc phân từ thành các từ loại (*như: danh từ, động từ, tính từ, ...*) và các tiểu từ loại (*như: danh từ chung/riêng, động từ nội động/ngoại động, ...*). Phạm trù biến đổi từ là phạm trù ngữ pháp bộ phận bao gồm việc phân chia từ ứng với các nhãn của các phạm trù ngữ pháp như: cách (*mood*), giống (*gender*), số (*number*), thì (*tense*), lối (*voice*),... Để ngắn gọn và chính xác, từ đây trở đi, chúng ta có thể gọi chung cho các loại nhãn trên là nhãn từ pháp.

Về nhãn ngữ nghĩa. Qua khảo sát ý nghĩa từ vựng của mỗi từ thực, ta thấy nói chung mỗi từ có thể mang nhiều nghĩa khác nhau, nhưng trong một ngữ cảnh cụ thể, thì chúng sẽ mang một nghĩa nhất định nào đó. Chẳng hạn, danh từ “bank” trong tiếng Anh có thể là “ngân hàng”, hoặc “bờ sông” hoặc “dây”; danh từ “đường” trong tiếng Việt có thể có nghĩa là “đường ăn” (*sugar*) hay “đường đi” (*line*),... Để dễ phân biệt các nghĩa từ vựng khác

nhau, các nhà ngữ nghĩa học từ vựng học và tâm lí học – ngôn ngữ đã phân chia toàn bộ các ý nghĩa từ vựng có thể có thành hệ thống các ý niệm (*cây ý niệm*) và mỗi ý niệm như vậy được coi như là một nhãn ngữ nghĩa của từ. Chẳng hạn, với danh từ “bank” nói trên, các nghĩa tương ứng của chúng sẽ là: “ngân hàng” thuộc về ý niệm “công trình xây dựng nhân tạo” (*nhãn HOU*); “bờ sông” sẽ thuộc về ý niệm “công trình thiên tạo” (*nhãn NAT*); “dây” sẽ thuộc về ý niệm “sự sắp xếp tổ chức” (*nhãn GRP*). Tương tự cho danh từ “đường” trong tiếng Việt, nghĩa “đường ăn” sẽ được xếp vào ý niệm “hoá chất” (*nhãn CHM*); còn nghĩa “đường đi” sẽ được xếp vào ý niệm “đường nét, dấu vết” (*nhãn LIN*);...

Về phương pháp gán nhãn ngôn ngữ. Một từ (*hay một đơn vị ngôn ngữ nào đó*) trong một phương diện nào đó (*hình thái, ngữ pháp, ngữ nghĩa,...*) thường mang nhiều hơn một nhãn ngôn ngữ, vì vậy, vấn đề khó khăn nhất trong việc gán nhãn ngôn ngữ cho ngữ liệu chính là việc làm thế nào để chọn được nhãn đúng trong số các nhãn khả dĩ của một đơn vị ngôn ngữ? Đây chính là bài toán khử tính nhập nhằng (*disambiguate*) cho ngôn ngữ tự nhiên ở hầu hết các cấp độ (*từ, ngữ, câu*) và các khía cạnh (*hình thái, ngữ pháp, ngữ nghĩa, ngữ dụng*). Đây cũng là công việc khó khăn, tốn kém thời gian và công sức nhất. Để giải quyết bài toán này, người ta đã tìm cách xây dựng các chương trình sử dụng nhiều mô hình xử lí (*thống kê, suy luận,...*) phức tạp trong lĩnh vực trí tuệ nhân tạo, tính toán thông minh để giải quyết tự động bài toán nói trên. Đến nay, đối với tiếng Anh, các bài toán về gán nhãn hình thái và ngữ pháp đã đạt kết quả khả quan (*trên 90%, có bài toán đạt 98% như bài toán gán nhãn từ loại*). Đối với tiếng Việt, thì kết quả này tuy chưa bằng nước ngoài nhưng ngày càng được cải thiện do có sự đầu tư xây dựng các kho ngữ liệu lớn đã được gán nhãn ngôn ngữ để dùng làm ngữ liệu huấn luyện cho máy tính cũng như áp dụng các thuật giải, các mô hình xử lí ngày càng chính xác hơn.

2.3.4 Xây dựng kho tư liệu

Bên cạnh kho ngữ liệu (*chủ yếu là dạng văn bản, text*), chúng ta còn có các kho thông tin khác về từ, như: ngữ âm của mục từ (*tập tin âm thanh chứa giọng phát âm của người bản xứ chuẩn cho mục từ đó*), hình ảnh (*tĩnh và động*) minh hoạ cho mục từ đó, các tri thức ngôn ngữ (*hình thái, ngữ pháp, ngữ nghĩa, ngữ dụng, từ nguyên*) của mục từ; tri thức bách khoa có liên quan đến mục từ đó.

Toàn bộ các thông tin nói trên, chúng ta cần lưu dưới dạng chuẩn và được gán nhãn theo quy tắc nhất quán bằng các thẻ (*tag*) trong tập tin XML để sau này chương trình máy tính có thể tham chiếu hai chiều (*double-link*) dễ dàng và chính xác. Các dạng chuẩn nên chọn dạng gốc và có thể hiệu chỉnh (*edit*) được, như:

- ✓ Hình ảnh: cùng dạng bitmap và cùng độ phân giải (*chẳng hạn: 320x240 pixel*)
- ✓ Âm thanh: cùng dạng wav, stereo, 2 channels, 44100 Hz, PCM
- ✓ Video: AVI
- ✓ Text: utf-8

Để cho máy tính có cơ sở tri thức trong việc suy diễn và xử lý tự động, chúng ta phải xây dựng các danh sách đặc biệt, như: danh sách các mục từ theo tần suất xuất hiện, danh sách các từ gốc, danh sách các từ cơ bản, danh sách tên riêng, danh sách viết tắt, danh sách các thẻ, các nhãn; danh sách các ngoại lệ; các quy tắc suy diễn (*quy tắc bỏ dấu thanh tiếng Việt, viết hoa, chấm câu*) để máy tính mới có thể trợ giúp chúng ta trong việc kiểm lỗi chính tả, chuẩn hoá văn bản, tập tin mô tả cấu trúc (DTD: *Document Type Definition*) của tập tin dữ liệu (XML),...

2.3 CÁC THUẬT TOÁN HÀM ỨNG DỤNG

2.3.1 Thuật toán support vector machine

Mô hình toán học

Support Vector Machine không đưa ra khả năng output bằng 1 như Logistic Regression, thay vào đó nó chỉ đơn thuần dự đoán output bằng 0 hay bằng 1.

$$\hat{y} = \begin{cases} 1 & \text{khi } x^T w \geq 0 \\ 0 & \text{khi } x^T w < 0 \end{cases}$$

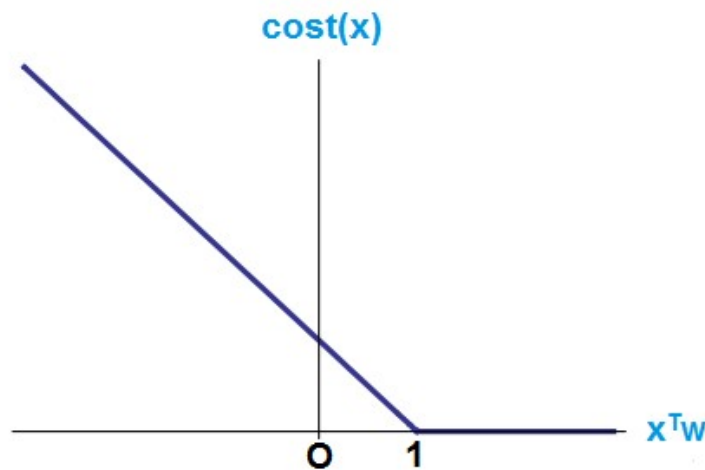
Độ chính xác của phương trình giả thuyết

Trong Support Vector Machine, phần mất mát mỗi input đóng góp có dạng hàm hinge loss

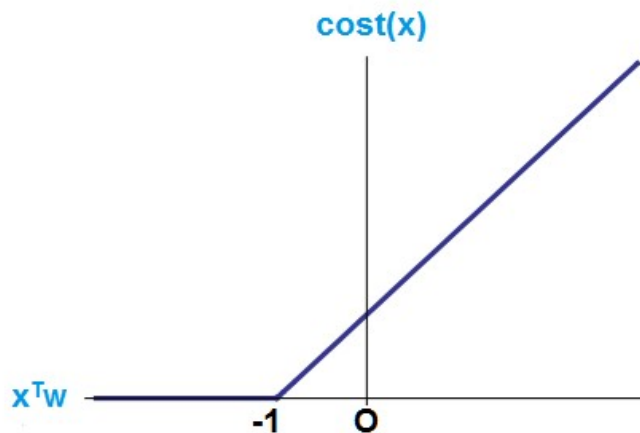
$$\text{cost}(x) = \begin{cases} \max(0, k(1 - x^T w)) & \text{khi } y = 1 \\ \max(0, k(1 + x^T w)) & \text{khi } y = 0 \end{cases}$$

với k là số dương bất kỳ.

Khi $y = 1$, $\text{cost}(x) = 0$ nếu $x^T w \geq 1$ và $\text{cost}(x)$ tăng dần nếu $x^T w < 1$ và tiến tới âm vô cực.



Khi $y = 0$, $\text{cost}(x) = 0$ nếu $x^T w \leq -1$ và $\text{cost}(x)$ tăng dần nếu $x^T w > -1$ và tiến tới dương vô cực.



Hàm mất mát của Support Vector Machine

$$J(w) = C \sum_{i=1}^m [y^{(i)} \max(0, k(1 - x^{(i)T} w)) + (1 - y^{(i)}) \max(0, k(1 + x^{(i)T} w))] + \frac{1}{2} \sum_{j=1}^n w_j^2$$

Ở đây hằng số C đóng vai trò như $1/\lambda$ là độ chính quy hóa của hàm mất mát giúp kiểm soát sai lầm của phương trình giả thuyết. Khi xảy ra underfitting, ta cần tăng C . Khi xảy ra overfitting, ta cần giảm C .

Nghiệm của thuật toán support vector machine

Ta có thể tìm điểm cực tiểu của hàm mất mát bằng thuật toán Gradient Descent với các biến đổi

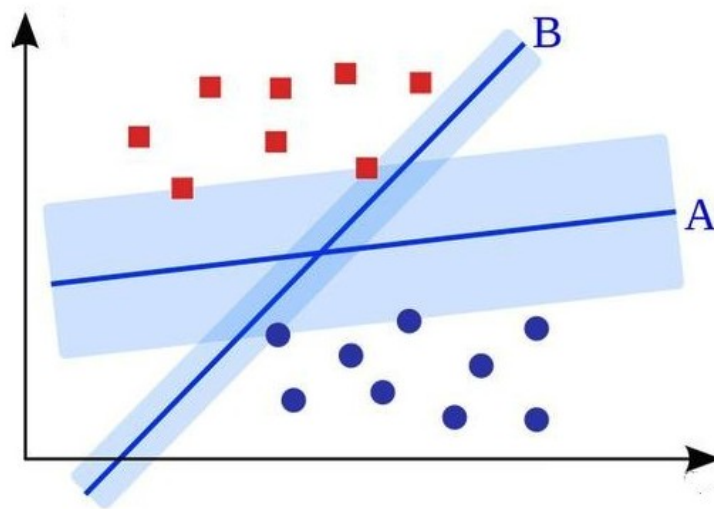
$$w_0 := w_0 - \alpha C \sum_{i=1}^m [y^{(i)} (x^{(i)T} w \geq 1 ? 0 : -k) + (1 - y^{(i)}) (x^{(i)T} w \geq -1 ? k : 0)]$$

$$w_1 := w_1(1 - \alpha) - \alpha C \sum_{i=1}^m [y^{(i)} (x^{(i)T} w \geq 1 ? 0 : -kx_1^{(i)}) + (1 - y^{(i)}) (x^{(i)T} w \geq -1 ? kx_1^{(i)} : 0)]$$

...

$$w_n := w_n(1 - \alpha) - \alpha C \sum_{i=1}^m [y^{(i)} (x^{(i)T} w \geq 1 ? 0 : -kx_n^{(i)}) + (1 - y^{(i)}) (x^{(i)T} w \geq -1 ? kx_n^{(i)} : 0)]$$

Một đặc điểm của Support Vector Machine là nó luôn cố gắng tìm nghiệm sao cho Decision Boundary cách xa các điểm dữ liệu nhất cho thể. Trong hình dưới đây, thuật toán có xu hướng chọn phương án A thay vì phương án B vì nó cách xa các điểm dữ liệu hơn. Điều này có thể dẫn tới overfitting và ta có thể làm giảm xu hướng này bằng cách giảm C .



Việc tìm nghiệm của thuật toán Support Vector Machine tương đối phức tạp nếu cài đặt thủ công. Có rất nhiều thư viện đã được cài đặt sẵn Support Vector Machine và ta nên dùng chúng vì chẳng những giúp tiết kiệm thời gian mà các thư viện đó còn được áp dụng nhiều kỹ thuật tối ưu hóa để thuật toán chạy nhanh hơn.

Bài toán phân chia hai classes và lập trình tìm nghiệm cho SVM

Bài toán: Giả sử rằng có hai class khác nhau được mô tả bởi các điểm trong không gian nhiều chiều, hai classes này linearly separable, tức tồn tại một siêu phẳng phân chia chính xác hai classes đó. Hãy tìm một siêu mặt phẳng phân chia hai classes đó, tức tất cả các điểm thuộc một class nằm về cùng một phía của siêu mặt phẳng đó và ngược phía với toàn bộ các điểm thuộc class còn lại.

Sử dụng hàm `sklearn.svm.SVC` ở đây. Các bài toán thực tế thường sử dụng thư viện `libsvm` được viết trên ngôn ngữ C, có API cho Python và Matlab.

```

from sklearn.svm import SVC

y1 = y.reshape((2*N,))
X1 = X.T # each sample is one row
clf = SVC(kernel = 'linear', C = 1e5) # just a big number

clf.fit(X1, y1)

w = clf.coef_
b = clf.intercept_
print('w = ', w)
print('b = ', b)

w = [[-2.00971102  0.64194082]]
b = [ 4.66595309]

```

2.3.2 Thuật toán Bayes

Bộ phân lớp Bayes là một giải thuật thuộc lớp giải thuật thống kê, nó có thể dự đoán xác suất của một phần tử dữ liệu thuộc vào một lớp là bao nhiêu. Phân lớp Bayes được dựa trên định lý Bayes (*định lý được đặt theo tên tác giả của nó là Thomas Bayes*).

2.3.2.1 Định lý Bayes

- Gọi A, B là hai biến cố

Với $P(B) > 0$:

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

Suy ra:

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

Công thức Bayes:

$$\begin{aligned}
 P(B|A) &= \frac{P(AB)}{P(A)} = \frac{P(A|B)P(B)}{P(A)} = \frac{P(A|B)P(B)}{P(AB) + P(\overline{A}B)} \\
 &= \frac{P(A|B)P(B)}{P(A|B)P(B) + P(A|\overline{B})P(\overline{B})}
 \end{aligned}$$

- Công thức Bayes tổng quát

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:

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

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 . Giả sử tập dữ liệu khách hàng của chúng ta được mô tả bởi các thuộc tính tuổi và thu nhập, và một khách hàng X có tuổi là 25 và thu nhập là 2000\$. Giả sử H là giả thiết khách hàng đó sẽ mua máy tính, thì $P(H|X)$ phản ánh xác suất người dùng X sẽ mua máy tính với điều kiện ta biết tuổi và thu nhập của người đó. Ngược lại $P(H)$ là xác suất tiên nghiệm (*prior probability hay priori probability*) của H . Trong ví dụ trên, nó là xác suất một khách hàng sẽ mua máy tính mà không cần biết các thông tin về tuổi hay thu nhập của họ. Hay nói cách khác, xác suất này không phụ thuộc vào yếu tố X . Tương tự, $P(X|H)$ là xác suất của X với điều kiện H (*likelihood*), nó là một xác suất hậu nghiệm. Ví dụ, nó là xác suất người dùng X (*có tuổi là 25 và thu nhập là \$200*) sẽ mua

máy tính với điều kiện ta đã biết người đó sẽ mua máy tính. Cuối cùng $P(X)$ là xác suất tiên nghiệm của X . Trong ví dụ trên, nó sẽ là xác suất một người trong tập dữ liệu sẽ có tuổi 25 và thu nhập \$2000.

2.3.2.2 Bài toán phân loại Bắc hay Nam và thuật toán Bayes

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

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

Sử dụng thư viện Sklearn để giải quyết bài toán như sau:

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

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

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

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

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

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

Kết quả

```
Predicting class of d5: B  
Probability of d6 in each class: [[ 0.29175335  0.70824665]]
```

CHƯƠNG 3

PHÂN TÍCH BÌNH LUẬN QUẢNG CÁO BỘT GIẶT ABBA TRÊN KÊNH YOUTUBE

2.1 GIỚI THIỆU VỀ BỘT GIẶT ABBA

Aba là bột giặt thuộc công ty Đại Việt Hương- công ty chuyên sản xuất các sản phẩm chăm sóc cơ thể và gia đình nổi tiếng tại Việt Nam. Ra đời từ năm 2005 và sau 15 năm hoạt động, Đại Việt Hương đã cho ra đời nhiều dòng sản phẩm nổi tiếng như Biona, sữa rửa mặt E100 và đặc biệt là bột giặt Aba. Có thể nói, Aba là một trong những sản phẩm thành công nhất của Đại Việt Hương bởi nó giải quyết được hầu hết nhu cầu và mong muốn của chị em phụ nữ về việc lựa chọn một loại bột giặt làm sạch quần áo.

Ai cũng chê quảng cáo bột giặt Aba vừa dài vừa dở tới mức "nhảm nhí", nhưng sự thật đằng sau lại khiến nhiều người bất ngờ. Thế giới từ lâu đã luôn "ngập tràn" quảng cáo, chỉ cần mở mắt ra, hàng loạt nội dung sẽ lần lượt xuất hiện nhằm chi phối quyết định mua sắm của con người. Về cơ bản, quảng cáo luôn đề cao điểm mạnh của sản phẩm/ dịch vụ với mục tiêu nâng cao mức độ nhận diện thương hiệu và doanh thu.

theo thống kê gần nhất của VIETNAM-TAM, bình quân mỗi tối, một kênh truyền hình sẽ làm "quá tải" người xem với 50 đoạn quảng cáo, mỗi đoạn dài từ 2 đến 4 phút. Trong đó nổi bật nhất là ngành hàng tiêu dùng nhanh với sự thống trị của hàng loạt "đại gia" quốc tế, từ dầu gội, sữa tắm, bánh kẹo và đặc biệt là bột giặt. Tưởng chừng như các thương hiệu Việt chỉ còn cạnh tranh được ở "vùng quê", nhưng khi chất lượng quảng cáo đang dần trở nên quan trọng hơn số lượng quảng cáo, các doanh nghiệp nội địa ngày càng tự tin cạnh tranh ngay tại thị trường thành thị.

Và đại diện cho xu hướng đó chính là Đại Việt Hương, một thương hiệu Việt "mạo hiểm" đưa sản phẩm bột giặt Aba vào phân khúc cao cấp với

giá thành chỉ thấp hơn 10% so với "ông hoàng" Omo của Unilever, trong khi các thương hiệu ít tên tuổi khác chủ động bán giá thấp hơn 30% đến 40% so với Omo. Nhưng dù ra sức chê bai và chỉ trích, người dùng dần cảm thấy đỡ quá... cũng hóa thú vị, vô số người tiêu dùng trở nên "nghiện" quảng cáo Aba, trông ngóng mẫu quảng cáo mới từng ngày để tiếp tục đem ra bàn tán. Với xu hướng đó, có thể thấy bột giặt Aba đã thành công vang dội vì ngay lập tức gia tăng mức độ nhận biết.

2.2 MÔ TẢ PHƯƠNG THỨC LẤY DỮ LIỆU CỦA VIDEO QUẢNG CÁO BỘT GIẶT ABBA TRÊN YOUTUBE

2.3.1 Phân Tích Cảm Xúc

2.2.2.1 Phân tích cảm xúc tiếp cận theo xử lý ngôn ngữ tự nhiên

Các ý kiến, bình luận của khách hàng là dạng ngôn ngữ tự nhiên được viết ra (Eisenstein, 2019; Popescu & Etzioni, 2007). Trong một số nghiên cứu của Buche, Chandak, và Zadgaonkar (2013), Sun, Luo, và Chen (2017), Thanh và Phuc (2015) đã đưa ra một số phương pháp và kỹ thuật xử lý ngôn ngữ tự nhiên trong việc phân tích ý kiến và cảm xúc khách hàng thông qua bình luận trực tuyến. Như vậy, việc chuẩn bị tập dữ liệu để phân tích, ở đây là dữ liệu văn bản là các nội dung bình luận của khách hàng để lại sau khi trải nghiệm những sản phẩm và dịch của các cửa hàng, có thể trên website, youtube, trên các trang mạng xã hội. Tiếp theo là tiền xử lý, ta tiến hành làm sạch dữ liệu, loại bỏ các kí tự đặc biệt, các dữ liệu rác, các dữ liệu không chuẩn hóa, chuẩn hóa dữ liệu về ngữ pháp ngữ nghĩa. Khảo sát phân tích dữ liệu, xem dữ liệu đã đầy đủ chưa, phân bổ độ dài của nội dung. Giai đoạn này nghiên cứu sẽ phát họa khái quát tính chất, nội dung, số lượng của tập dữ liệu mình thu được. Lựa chọn các yếu tố đầu vào để phân tích, và dữ liệu ban đầu sẽ có rất nhiều chiều. Lựa chọn chiều nào thích hợp nhất để phân tích là việc rất quan trọng. Các chiều đầu vào càng chính xác thì kết quả phân tích sẽ có

độ chính xác càng cao. Bước cuối cùng là đánh giá kết quả và triển khai dự án.

2.2.2.2 Phân tích cảm xúc tiếp cận theo phương pháp Học máy

Phân tích cảm xúc đã được định nghĩa là tính toán nghiên cứu ý kiến, tình cảm và cảm xúc thể hiện trong văn bản (Liu, 2012). Nói cách khác, khai thác ý kiến là một phương pháp trích xuất ý kiến của người đã tạo ra một tài liệu cụ thể gần đây đã trở thành mối quan tâm nghiên cứu lớn nhất trong mạng xã hội (Pang & Lee, 2008). Tầm quan trọng ngày càng tăng của phân tích tình cảm tăng dần cùng với sự phát triển của phương tiện truyền thông xã hội như đánh giá, thảo luận diễn đàn, và mạng xã hội. Đặc biệt, trong thời đại phát triển kỹ thuật số, chúng ta hiện có một khối lượng dữ liệu lớn được ghi lại dưới dạng văn bản để phân tích. Học máy là một ứng dụng của Trí tuệ nhân tạo, là lĩnh vực giúp hệ thống tự động hiểu dữ liệu từ dữ liệu được đào tạo mà không cần lập trình cụ thể. Học máy tập trung vào vấn đề cung cấp hệ thống tự động hiểu dữ liệu và thực hiện các phép dự đoán. Học máy chia làm 4 phần (Das, Dey, Pal, & Roy, 2015): học có giám sát, học bán giám sát, học không giám sát và học củng cố.

Máy học có giám sát là thuật toán dự đoán dữ liệu đầu ra dựa vào các tập dữ liệu (*dữ liệu đầu vào, kết quả đầu ra*) đã biết từ trước. Có hai loại máy học có giám sát đó là phân loại và hồi quy. Phân loại thì dự đoán kết quả phân chia thành các nhóm dữ liệu có cùng tính chất, hồi quy thì cho ra kết quả dự đoán là một số thực cụ thể thay vì chỉ phân nhóm như học máy phân loại.

Máy học không giám sát là thuật toán dự đoán dữ liệu đầu ra dựa vào duy nhất tập dữ liệu đầu vào, dữ liệu đầu vào sẽ không được dán nhãn hoặc kết quả đầu ra. Thuật toán sẽ dựa vào cấu trúc dữ liệu để thực hiện lưu trữ và tính toán. Máy học không giám sát bao gồm phân nhóm và tích hợp. Thuật toán phân nhóm dựa sẽ phân nhóm toàn bộ dữ liệu thành các nhóm nhỏ dựa

trên dự liên quan của các dữ liệu trong nhóm. Thuật toán tích hợp sẽ khai phá một số quy luật dựa trên nhiều dữ liệu cho trước.

Học bán giám sát là thuật toán kết hợp cả hai thuật toán có giám sát và không giám sát. Áp dụng với một phần tập dữ liệu đã được dán nhãn, phần còn lại thì không được dán nhãn.

Học củng cố là thuật toán giúp hệ thống tự động xác định các hành vi để đạt hiệu quả tối ưu nhất.

Trong bài viết của nhóm, chúng tôi chọn phương pháp học có giám sát để áp dụng cho bài toán phân loại cảm xúc khách hàng dựa trên bình luận về các video clip trên youtube của sản phẩm bột giặt Abba.

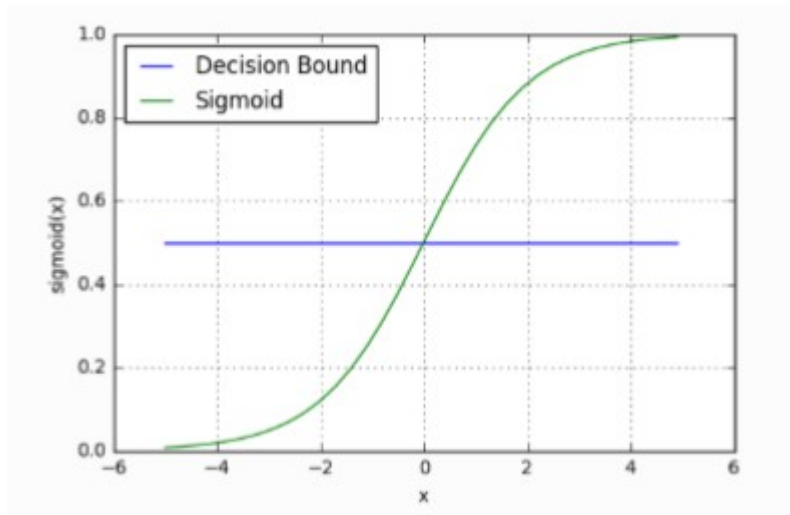
2.2.2.3 Thuật toán Hồi quy Logistic

Thuật toán Hồi quy Logistic thuộc học máy có giám sát để phân loại dữ liệu. Mô hình hồi quy Logistic áp dụng cho biến phụ thuộc là biến định tính hoặc định lượng chỉ có hai giá trị (có hoặc không) hay nhị phân là 0 hoặc 1. Điều này phù hợp với bài toán phân loại bình luận người dùng. Đầu ra của bài toán đó là xác định bình luận đó là tích cực hay tiêu cực. Phương trình tổng quát (hàm Sigmoid) hoặc hàm Logistic:

$$y = f(s) = \frac{1}{1 + e^{-s}} \quad (1)$$

Trong đó, $f(s)$ là xác suất xảy ra giá trị $y = 1$ hoặc $y = 0$, s là phương trình tuyến tính phụ thuộc vào các biến đầu vào. Phương trình mô hình đơn biến: $s = \alpha_0 + \alpha_1 x_1$, phương trình tuyến tính phụ thuộc vào duy nhất biến x_1 . Phương trình mô hình đa biến: $s = \alpha_0 + \alpha_1 x_1 + \dots + \alpha_n$, phương trình tuyến tính phụ thuộc vào các biến x . Dạng ma trận khi $\alpha_0 = 0$ là:

$$[\alpha_1 \alpha_2 \dots \alpha_n] \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$



Hình 1. Đồ thị hàm Sigmoid (Hieu, 2018)

Đồ thị hàm số thể hiện:

$$s > 0 \Leftrightarrow e^{-s} < 1 \Leftrightarrow f(s) > 0.5$$

Chia làm hai lớp:

$y = 0$ nếu $s < 0$

$y = 1$ nếu $s \geq 0$

Các tính chất hàm Logistic:

- ✓ Miền xác định: Tất cả các số thực;
- ✓ Miền giá trị: (0,1);
- ✓ Hàm liên tục;
- ✓ Hàm tăng trên miền xác định;
- ✓ Hàm đối xứng qua điểm $(0, \frac{1}{2})$, không phải hàm chẵn cũng không phải hàm lẻ;
- ✓ Bị giới hạn trên và dưới;

- ✓ Không có cực trị địa phương;
- ✓ Tiệm cận ngang: $y = 0$ và $y = 1$;
- ✓ Không có tiệm cận đứng;
- ✓ Mượt (smooth) nên có đạo hàm mọi nơi, có thể được lợi trong việc tối ưu hàm Sigmoid.

Giải thích:  Giới hạn

$$\lim_{s \rightarrow -\infty} f(s) = \lim_{s \rightarrow -\infty} \left(\frac{1}{1 + e^{-s}} \right) = 0 \quad (2)$$

$$\lim_{s \rightarrow +\infty} f(s) = \lim_{s \rightarrow +\infty} \left(\frac{1}{1 + e^{-s}} \right) = 1 \quad (3)$$

Hàm mất mát (Jurafsky & Martin, 2008): hàm mất mát là hàm số xác định sự chênh lệch giữa đầu ra y dự đoán so với kết quả đầu ra y đã đúng (y dùng trong huấn luyện). Việc tối ưu hàm mất mát sẽ cho ra kết quả bài toán chính xác hơn

$$j(\theta) = \frac{1}{m} \sum_{i=1}^m (\text{cost}(h_{\theta}(x^{(i)}), y^{(i)})) = \frac{1}{m} \sum_{i=1}^m [-y^{(i)} \log(h_{\theta}(x^{(i)})) - (1 - y^{(i)}) \log(1 - h_{\theta}(x^{(i)}))] \quad (4)$$

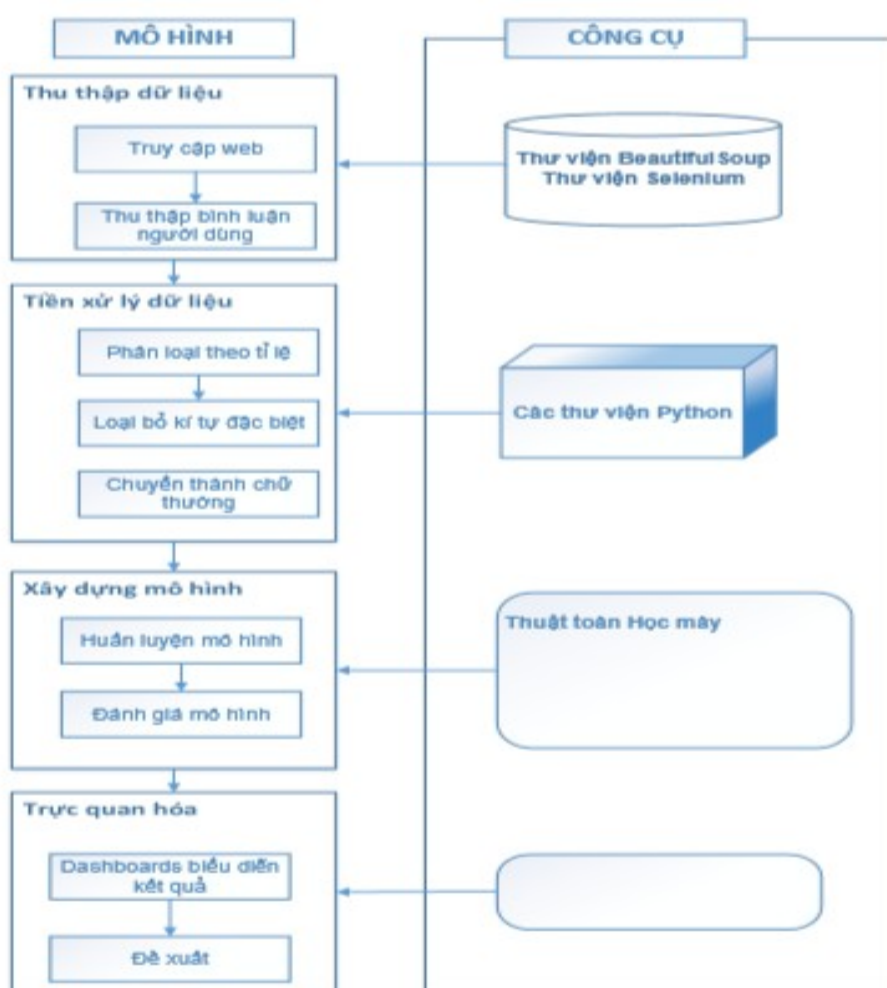
$$\text{cost}(h_{\theta}(x), y) = \begin{cases} -\log(h_{\theta}(x)) & \text{if } y=1 \\ -\log(1 - h_{\theta}(x)) & \text{if } y=0 \end{cases}$$

2.3.2 Bài toán thực nghiệm

2.2.2.1 Tổng quan

Trong bài viết này, trước tiên chúng tôi tiến hành thu thập dữ liệu thô từ các bình luận trên Youtube của các video clip quảng cáo của bột giặt Abba. Sau đó dữ liệu thô được tiền xử lý và lấy mẫu, và gán nhãn trước khi tiến hành học máy. Dữ liệu lấy mẫu được chia thành ba nhóm: tập dữ liệu huấn luyện (training data), tập dữ liệu xác nhận (validation data) và tập dữ liệu

kiểm tra (test data). Tập dữ liệu huấn luyện được sử dụng để thiết lập các mô hình học máy, bộ dữ liệu xác nhận được sử dụng để lặp lại và tinh chỉnh các mô hình được chọn, chúng tôi dựa trên kết quả phân loại chính xác trên dữ liệu tập kiểm tra để tìm ra mô hình học máy phù hợp nhất. Các bộ dữ liệu kiểm tra chỉ được sử dụng một lần là bước cuối cùng để báo cáo tỷ lệ lỗi ước tính cho dự đoán trong tương lai.



2.2.2.2 Thu thập dữ liệu

Các thư viện BeautifulSoup và Selenium trên ngôn ngữ Python được dùng để thu thập dữ liệu trên Youtube. Việc thu thập dữ liệu bình luận trên các Video Clip quảng cáo của Abba được thực hiện thông qua API của Youtube.. Tập dữ liệu thu thập được có 8,558 dòng chứa nội dung bình luận của khách hàng về clip .

```
[ ] !pip install --upgrade google-api-python-client

[ ] from googleapiclient.discovery import build
youtubeApiKey='AIzaSyDw0P-U01m0FTLio5N0a00a7m2zIMC_hwc'
youtube=build('youtube','v3',developerKey=youtubeApiKey)

[ ] import json
from csv import writer

[ ] response=youtube.commentThreads().list(part='snippet',maxResults=100,textFormat='plainText',order='time',videoId='_x50ox0_0Hw').execute()

[ ] comments = []

[ ] while response: # this loop will continue to run until you max out your quota
    for item in response['items']:
        #5 index item for desired data features
        comment = item['snippet']['topLevelComment']['snippet']['textDisplay']

        #6 append to lists
        comments.append(comment)

    #8 check for nextPageToken, and if it exists, set response equal to the JSON response
    if 'nextPageToken' in response:
        response = youtube.commentThreads().list(
            part='snippet',
            maxResults=100,
            textFormat='plainText',
            order='time',
            videoId='_x50ox0_0Hw',
            pageToken=response['nextPageToken']
        ).execute()
    else:
        break

[ ] def append_list_as_row(file_name, list_of_elem):
    # Open file in append mode
    with open(file_name, 'a') as write_obj:
        csv_writer = writer(write_obj)
        for ad in list_of_elem:
            csv_writer.writerow([ad])

[ ] ucomments=set(comments)

[ ] append_list_as_row('Comment_Youtube.csv', ucomments)
```

2.2.2.3 Tiền xử lý dữ liệu

Dữ liệu thu thập về sẽ có dạng thô, do chưa qua xử lý nên có thể dữ liệu bị rỗng, dữ liệu sai chính tả, dữ liệu quá ngắn, quá dài hoặc chứa các biểu tượng icon. Điều này sẽ gây ảnh hưởng đến kết quả của việc phân tích, vì vậy ta cần làm sạch dữ liệu.

Xóa các icon, kí tự đặc biệt: các kí tự đặc biệt không mang ý nghĩa phân loại, mặc khác sẽ gây nhiễu trong quá trình phân tích. Chuyển tất cả về chữ thường: mỗi số, ký tự đặc biệt, ký tự là đại diện cho một dãy nhị phân trong bộ nhớ máy tính. Chữ in hoa sẽ có mã Unicode khác chữ in thường, về mặt ngữ nghĩa là giống nhau tuy nhiên máy tính sẽ không thể phân biệt dữ liệu đầu vào, dẫn đến có thể kết quả dự đoán bị ảnh hưởng. Vì vậy việc chuyển toàn bộ chữ về chữ thường là hợp lý cho hệ thống phân tích và dự đoán.

Chuyển dạng từ rõ nghĩa: việc chuyển dạng từ rõ nghĩa là cần thiết cho bước tiền xử lý dữ liệu. Các bình luận trên Youtube do người dùng bình luận tiếng Việt nên việc viết tắt hoặc sai chính tả. Chẳng hạn từ ko hay (không hay), vs (với), 15k (15000) ... hay dữ liệu không đồng bộ, không chuẩn hóa. Việc này sẽ ảnh hưởng gây nhiều kết quả phân tích. Trong quá trình huấn luyện của học máy, dữ liệu đưa vào là “không hay”, nhưng khi dự đoán dữ liệu đầu ra, cụm từ “ko hay” không xuất hiện trong quá trình huấn luyện, vì vậy sẽ khó thể nhận diện cảm xúc và dự đoán kết quả được.

Xóa dòng dữ liệu: tập dữ liệu thu về sẽ có nhiều dữ liệu bị trống, dữ liệu trống không có ý nghĩa trong quá trình phân tích, gây tốn bộ nhớ lưu trữ.

2.2.2.4 Tiền Xử Lý Và Làm Sạch Dữ Liệu

```
[ ] from csv import reader
import re

[ ] # read csv file as a list of lists
with open('Comment_Youtube.csv', 'r') as read_obj:
    # pass the file object to reader() to get the reader object
    csv_reader = reader(read_obj)
    # Pass reader object to list() to get a list of lists
    list_of_rowsx = list(csv_reader)
    list_of_rows = [val for sublist in list_of_rowsx for val in sublist]
    print(list_of_rows)
```

2.2.2.5 Xóa URL

```
[ ] for item in list_of_rows:
    sear = re.search("www|https|\.com|\.vn|\.asia|\.be", item)
    if sear:
        reitem = re.sub(r'http\S+|www\S+|\.com|\.vn', '', item)
        list_of_rows.remove(item)
        list_of_rows.append(reitem)
```

2.2.2.6 Xóa Email

```
[ ] for item in list_of_rows:
    sear = re.search("\w@\w+", item)
    if sear:
        reitem = re.sub(r'\w@\w+', '', item)
        list_of_rows.remove(item)
        list_of_rows.append(reitem)
```

2.2.2.7 Xóa SĐT

```
[ ] for item in list_of_rows:
    sear = re.search("^0\d{9}", item)
    if sear:
        reitem = re.sub(r'^0\d{9}', '', item)
        list_of_rows.remove(item)
        list_of_rows.append(reitem)
```

2.2.2.8 Xóa \n

```
[ ] for item in list_of_rows:
    sear = re.search("\n", item)
    if sear:
        reitem = re.sub(r'\n', ' ', item)
        list_of_rows.remove(item)
        list_of_rows.append(reitem)
```

2.2.2.9 Xóa Timestamp

```
[ ] for item in list_of_rows:
    sear = re.search("\d{1}:\d{2}:\d{2}:\d{2}", item)
    if sear:
        reitem = re.sub(r'\d{1}:\d{2}:\d{2}:\d{2}', '', item)
        list_of_rows.remove(item)
        list_of_rows.append(reitem)
```

2.2.2.10 Xóa Ký tự đặc biệt

```
[ ] for item1 in list_of_rows:
    reitem1 = re.sub(r'^\W+', ' ', item1)
    list_of_rows.remove(item1)
    list_of_rows.append(reitem1)
```

2.2.2.11 Xóa Khoảng trắng thừa

```
[ ] for item2 in list_of_rows1:
    reitem2 = re.sub(r"\s+", " ", item2)
    list_of_rows1.remove(item2)
    list_of_rows1.append(reitem2)
```

2.2.2.12 Xóa row trống

```
[ ] for item3 in list_of_rows1:
    if item3 == "" or item3 == " ":
        list_of_rows1.remove(item3)
```

2.2.2.13 Lowercase

```
[ ] for item in list_of_rows1:
    item1 = item.lower()
    list_of_rows1.remove(item)
    list_of_rows1.append(item1)
```

2.2.2.14 Chuẩn hóa Unicode

```
[ ] """
    Copyright @ nguyenvanhieu.vn
    """
    import re
    import os
    import sys
    # from Logger import LogEventSourcing
    from datetime import datetime
    import dateutil.parser
    import traceback
    import time
    import requests

    def remove_html(txt):
        return re.sub(r'<[^>*>', '', txt)
```



```

def vn_sentence_to_telex_type(sentence):
    """
    Chuyển câu tiếng việt có dấu về kiểu gõ telex.
    :param sentence:
    :return:
    """
    words = sentence.split()
    for index, word in enumerate(words):
        words[index] = vn_word_to_telex_type(word)
    return ' '.join(words)

"""
End section: Chuyển câu văn về kiểu gõ telex khi không bật Unikey
"""

"""
Start section: Chuyển câu văn về cách gõ dấu kiểu cũ: dùng ôa úy thay ôa ưý
Xem tại đây: https://vi.wikipedia.org/wiki/Quy\_t%E1%BA%AFC\_%C4%91%E1%BA%B7t\_d%E1%BA%A5u\_thanh\_trong\_ch%E1%BB%AF\_qu%E1%BB%91c\_ng%E1%BB%AF
"""

def chuan_hoa_dau_tu_tiang_viet(word):
    if not is_valid_vietnam_word(word):
        return word

    chars = list(word)
    dau_cau = 0
    nguyen_am_index = []
    qu_or_gi = False
    for index, char in enumerate(chars):
        x, y = nguyen_am_to_ids.get(char, (-1, -1))
        if x == -1:
            continue
        elif x == 9: # check qu
            if index != 0 and chars[index - 1] == 'q':
                chars[index] = 'u'
                qu_or_gi = True
        elif x == 5: # check gi
            if index != 0 and chars[index - 1] == 'g':
                chars[index] = 'i'
                qu_or_gi = True
        if y != 0:
            dau_cau = y
            chars[index] = bang_nguyen_am[x][0]
            if not qu_or_gi or index != 1:
                nguyen_am_index.append(index)
        if len(nguyen_am_index) < 2:
            if qu_or_gi:
                if len(chars) == 2:
                    x, y = nguyen_am_to_ids.get(chars[1])
                    chars[1] = bang_nguyen_am[x][dau_cau]
                else:
                    x, y = nguyen_am_to_ids.get(chars[2], (-1, -1))
                    if x != -1:
                        chars[2] = bang_nguyen_am[x][dau_cau]
                    else:
                        chars[1] = bang_nguyen_am[5][dau_cau] if chars[1] == 'i' else bang_nguyen_am[9][dau_cau]
            return ''.join(chars)
        return word

    for index in nguyen_am_index:
        x, y = nguyen_am_to_ids[chars[index]]
        if x == 4 or x == 8: # ê, ô
            chars[index] = bang_nguyen_am[x][dau_cau]
            # for index2 in nguyen_am_index:
            #     if index2 != index:
            #         x, y = nguyen_am_to_ids[chars[index2]]
            #         chars[index2] = bang_nguyen_am[x][0]
            return ''.join(chars)

    if len(nguyen_am_index) == 2:
        if nguyen_am_index[-1] == len(chars) - 1:
            x, y = nguyen_am_to_ids[chars[nguyen_am_index[0]]]
            chars[nguyen_am_index[0]] = bang_nguyen_am[x][dau_cau]
            # x, y = nguyen_am_to_ids[chars[nguyen_am_index[1]]]
            # chars[nguyen_am_index[1]] = bang_nguyen_am[x][0]
        else:
            # x, y = nguyen_am_to_ids[chars[nguyen_am_index[0]]]
            # chars[nguyen_am_index[0]] = bang_nguyen_am[x][0]
            x, y = nguyen_am_to_ids[chars[nguyen_am_index[1]]]
            chars[nguyen_am_index[1]] = bang_nguyen_am[x][dau_cau]
    else:
        # x, y = nguyen_am_to_ids[chars[nguyen_am_index[0]]]
        # chars[nguyen_am_index[0]] = bang_nguyen_am[x][0]
        x, y = nguyen_am_to_ids[chars[nguyen_am_index[1]]]
        chars[nguyen_am_index[1]] = bang_nguyen_am[x][dau_cau]
        # x, y = nguyen_am_to_ids[chars[nguyen_am_index[2]]]
        # chars[nguyen_am_index[2]] = bang_nguyen_am[x][0]
    return ''.join(chars)

```

```

def is_valid_vietnam_word(word):
    chars = list(word)
    nguyen_am_index = -1
    for index, char in enumerate(chars):
        x, y = nguyen_am_to_ids.get(char, (-1, -1))
        if x != -1:
            if nguyen_am_index == -1:
                nguyen_am_index = index
            else:
                if index - nguyen_am_index != 1:
                    return False
                nguyen_am_index = index
    return True

def chuan_hoa_dau_cau_tiang_viet(sentence):
    """
    Chuyển câu tiếng việt về chuẩn gõ dấu kiểu cũ.
    :param sentence:
    :return:
    """
    sentence = sentence.lower()
    words = sentence.split()
    for index, word in enumerate(words):
        words[index] = chuan_hoa_dau_tu_tiang_viet(word)
    return ' '.join(words)

"""
End section: Chuyển câu văn về cách gõ dấu kiểu cũ: dùng ôa úy thay òa ỳ
Xem tại đây: https://vi.wikipedia.org/wiki/Quy\_t%E1%BA%AFC\_%C4%91%E1%BA%B7t\_d%E1%BA%A5u\_thanh\_trong\_ch%E1%BB%AF\_qu%E1%BB%91c\_ng%E1%BB%AF
"""

if __name__ == '__main__':
    # with open('C:/Users/htv/Desktop/testunicode.txt') as f:
    #     content = f.read()
    #     output = decodetounicode(content)
    #     writefile('C:/Users/htv/Desktop/unicode.txt', output)
    txt = 'nếu ngày mai trời nắng'
    # print(is_valid_vietnam_word(txt))
    txt = chuan_hoa_dau_cau_tiang_viet(txt)
    print(txt)

[ ] list_of_rows2=[]

[ ] for item in list_of_rows1:
    item1=convertwindown1525toutf8(item)
    list_of_rows2.append(item1)

```

2.2.2.15 Lưu file

```

[ ] import json
    from csv import writer

[ ] def append_list_as_row(file_name, list_of_elem):
    # Open file in append mode
    with open(file_name, 'a') as write_obj:
        csv_writer = writer(write_obj)
        for ad in list_of_elem:
            csv_writer.writerow([ad])

[ ] append_list_as_row('Comment_Youtube_Clear.csv', list_of_rows2)

```

2.3.3 Gán nhãn dữ liệu

Để thực hiện quá trình gán nhãn dữ liệu trước khi đưa vào huấn luyện, nghiên cứu áp dụng phương pháp phân loại cảm xúc theo điểm số đánh giá (Rating) của khách hàng (Liu, 2017) để phân chia tập dữ liệu đã thu thập được thành 2 bộ dữ liệu được gán nhãn theo quy tắc sau:

Rate ≤ 5 : bình luận nào đánh giá dưới 5 sao sẽ được dán nhãn là tiêu cực (negative).

Rate > 5 : bình luận nào đánh giá trên 5 sao sẽ được dán nhãn là tích cực (positive).

Kết quả gán nhãn cho thấy, chiếm đa số dữ liệu là các bình luận tích cực 78% so với tổng bình luận, bình luận tiêu cực chiếm 22% tổng bình luận.

2.3.4 Phương pháp biểu diễn văn bản

Trong học máy, máy tính không thể hiểu trực tiếp ngôn ngữ tự nhiên mà chỉ hiểu được ngôn ngữ khi chúng được biểu diễn dưới dạng không gian vector. Các chiều thuộc tính đầu vào sẽ được biểu diễn dưới dạng ma trận vector, có nhiều phương pháp để biểu diễn văn bản sang dạng ma trận vector chẳng hạn: cách truyền thống như mô hình Bag of N-grams, mô hình TF-IDF, mô hình chủ đề hay các cách cải tiến như các mô hình Word2Vec, GloVe, FastText (Sarkar, 2019). Trong nghiên cứu này, chúng tôi áp dụng hai phương pháp là Bag of N-grams và TF-IDF để thử nghiệm mô hình và biểu diễn dữ liệu.

Phương pháp Bag of word (BoW): mô hình BoW chỉ tập hợp tất cả các từ dạng một từ duy nhất, không chứa các cụm từ gồm nhiều từ ghép lại. Mô hình Bag of N-Grams sẽ giải quyết vấn đề này. Bag of N-grams sẽ thành lập một tập hợp các cụm từ gồm n-từ ghép lại với nhau tùy thuộc vào nhu cầu.

Phương pháp TF-IDF: mô hình Bag of word n-grams gặp một vài vấn đề đối với tập dữ liệu lớn, đó là các từ có tần suất xuất hiện nhiều ở đa số các đoạn văn bản, nhưng không có ý nghĩa phân loại, ví dụ như các từ “này”, “đó”, “rất”, “clip”, ... Khi đó chỉ số TF-IDF sẽ được dùng để tính toán và phát hiện các từ có trọng số cao và thấp.

Bước 1: Tính TF theo công thức

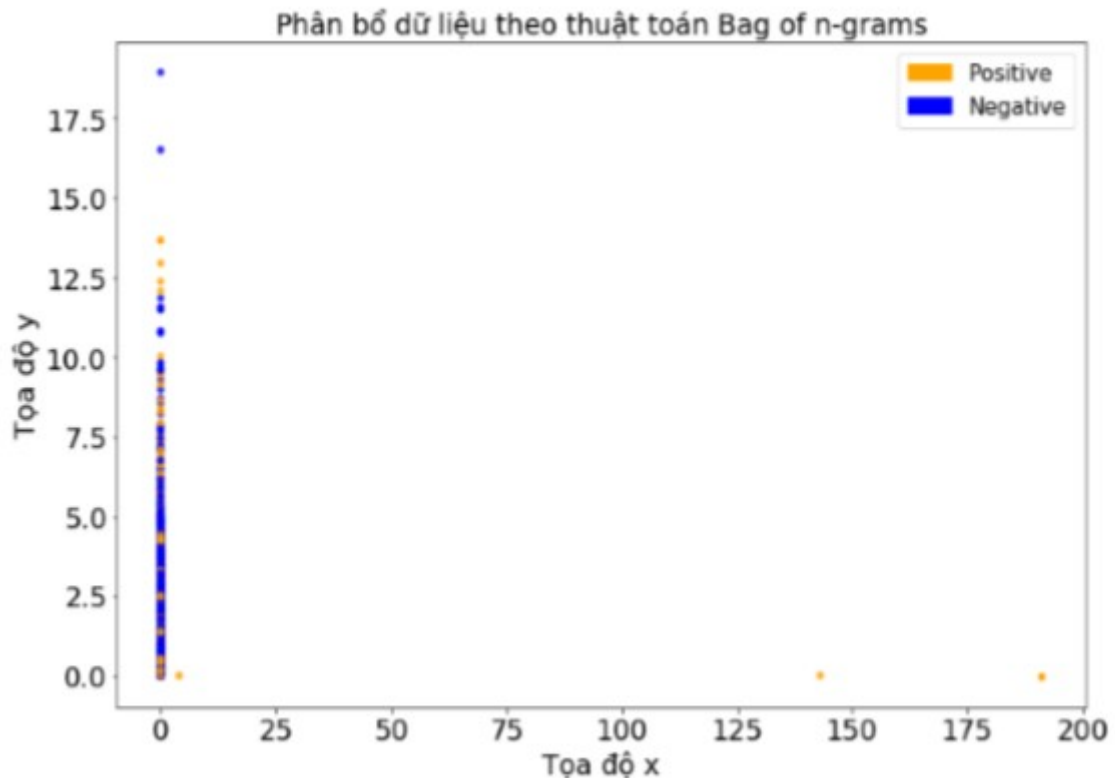
$$TF(t, d) = \frac{df(d, t)}{|D|}$$

Bước 2: Tính IDF theo công thức

$$TF-IDF(t, d, D) = tf(t, d) \frac{|D|}{df(d, t)}$$

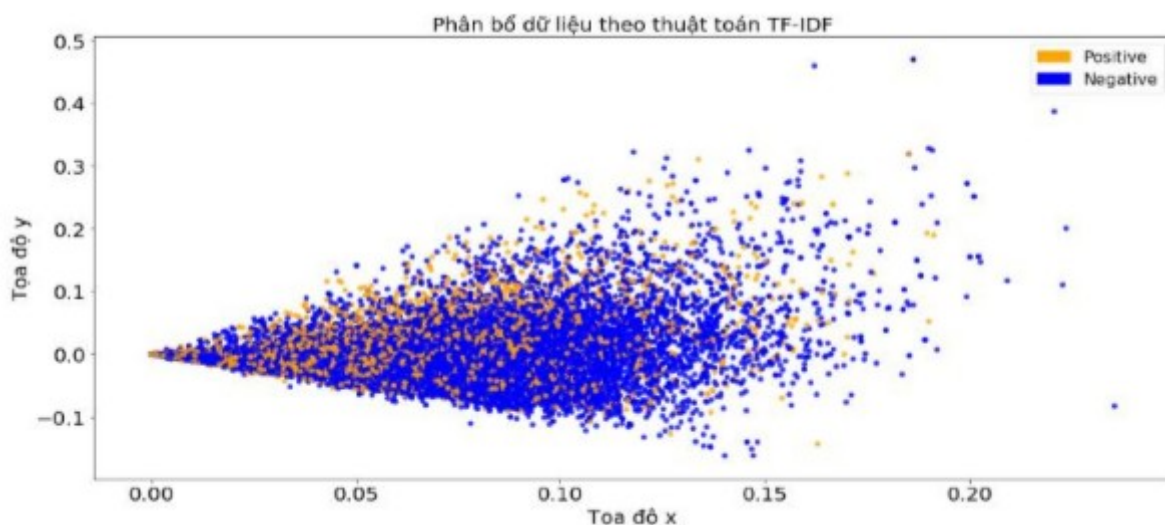
Ở đây:

- ✓ $|D|$ là số lượng các văn bản trong tập ngữ liệu;
- ✓ $df(d,t)$ là số lượng các văn bản mà từ t xuất hiện;
- ✓ $tf(t,d)$ là tần suất các từ xuất hiện trong một văn bản.
- ✓ Phân bố dữ liệu theo thuật toán BoW



Trước tiên, thuật toán BoW biểu diễn giá trị của các thuộc tính bằng giá trị 1 và 0. Từ không gian nhiều chiều ta chuyển đổi thành không gian 2 chiều thì các giá trị sẽ phân bố dọc theo trục y, giá trị 0 của trục x cố định và giá trị của trục y sẽ biến thiên. Dữ liệu phân bố theo một đường thẳng sẽ khó cho việc phân loại dữ liệu vì vậy nghiên cứu tiếp tục thực hiện phân bố dữ liệu theo TF-IDF.

Phân bố dữ liệu theo Phương pháp TF-IDF



Thuật toán TF-IDF không biểu diễn giá trị của các thuộc tính bằng giá trị 0 và 1 mà sẽ biểu diễn với giá trị trọng số TF-IDF đã tính. Chính vì vậy khi biểu diễn trên đồ thị giảm từ nhiều chiều sang 2 chiều, các giá trị của dữ liệu phân bố phụ thuộc cả hai chiều, khi trục x tăng thay đổi thì cũng kéo theo giá trị trục y thay đổi. Do vậy dữ liệu phân bố rời rạc và tách biệt hơn, việc này giúp quá trình phân loại sẽ dễ dàng hơn.

2.3.5 Chạy Thuật Toán Và Mô Hình

```
[ ] from csv import reader
import re

[ ] # read csv file as a list of lists
with open('Comment_Youtube_Clear.csv', 'r') as read_obj:
    # pass the file object to reader() to get the reader object
    csv_reader = reader(read_obj)
    # Pass reader object to list() to get a list of lists
    list_of_rowsx = list(csv_reader)
    list_of_rows = [val for sublist in list_of_rowsx for val in sublist]
```

2.3.6 Tách từ

```
[ ] import nltk
from nltk import word_tokenize
nltk.download('punkt')

[ ] list_tokenize = []

[ ] for item in list_of_rows:
    a1 = word_tokenize(item)
    list_tokenize.append(a1)
```

2.3.7 Xóa stopwords

```
[ ] # read csv file as a list of lists
with open('stopwords.txt', 'r') as read_obj:
    # pass the file object to reader() to get the reader object
    csv_reader = reader(read_obj)
    # Pass reader object to list() to get a list of lists
    stopword = list(csv_reader)
    stopwords = [val for sublist in stopword for val in sublist]
```

```
[ ] list_restopwords = []
```

```
[ ] for item1 in list_tokenize:
    a1 = []
    for item2 in item1:
        if item2 not in stopword:
            a1.append(item2)
    list_restopwords.append(a1)
```

2.3.8 WordCloud

```
[ ] from wordcloud import WordCloud, STOPWORDS
import matplotlib.pyplot as plt
%matplotlib inline
from nltk import word_tokenize
from nltk.stem import WordNetLemmatizer
from collections import Counter
import re
```

```
[ ] # Python program to convert a list to string
```

```
# Function to convert
def listToString(s):

    # initialize an empty string
    str1 = ""

    # traverse in the string
    for ele in s:
        str1 = str1 + ele

    # return string
    return str1
```

```
[ ] list_restopwords1 = []
```

```
[ ] for item in list_restopwords:
    a2 = listToString(item)
    list_restopwords1.append(a2)
```

```
[ ] list_restopwords2 = listToString(list_restopwords1)
```

```
[ ] a3 = word_tokenize(list_restopwords2)
```

```
[ ] Counter(a3).most_common(10)
```

```
❶ wordcloud = WordCloud(width = 800, height = 800,
                        background_color = 'white',
                        max_words=50,
                        min_font_size = 10).generate(list_restopwords2)
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

2.3.9 Phân loại cảm xúc

```
[76] !pip install underthesea
      from underthesea import sentiment
```

```
[77] Negative = 0
      Positive = 0
      Unknow = 0
      for item in list_restopwords1:
          b1 = sentiment(item)
          if b1 == 'negative':
              Negative += 1
          elif b1 == 'positive':
              Positive += 1
          else:
              Unknow += 1
```

```
[80] print("Negative: " + str(Negative))
      print("Positive: " + str(Positive))
      print("Unknow: " + str(Unknow))
```

2.3.10 Kết quả thực nghiệm :

2.2.2.1 Đánh giá mô hình

Tập dữ liệu đã được huấn luyện bằng mô hình học máy, sử dụng các thuật toán gồm: Decision Tree, Naïve Bayes, hồi quy Logistic. Với kết quả đánh giá mô hình, xác suất dự đoán như Bảng 1, nhận xét cụ thể như sau:

$$F_score = \frac{2 * Precision * Recall}{Precision + Recall}$$

True Positive (TP): tổng số lượng tích cực dự đoán ĐÚNG với số lượng tích cực thực tế;

False Positive (FP): tổng số lượng tích cực dự đoán SAI với số lượng tích cực thực tế;

True Negative (FN): tổng số lượng tiêu cực dự đoán ĐÚNG với số lượng tiêu cực thực tế;

False Negative (FN): tổng số lượng tiêu cực dự đoán SAI với số lượng tiêu cực thực tế.

Accuracy: độ chính xác trung bình các thuật toán, là tỷ lệ giữa kết quả dự đoán với dữ liệu thực tế. Cây quyết định và Hồi quy Logistic dự đoán 90%, nghĩa là trong 100 dữ liệu dự đoán thì hai mô hình này dự đoán đúng 90 dữ liệu so với kết quả thực tế.

Precision: được định nghĩa là số lượng dự đoán được thực hiện chính xác hoặc có liên quan trong số tất cả các dự đoán dựa trên lớp tích cực. Thuật toán Cây quyết định có độ chính xác là 90,075 % đối với dự đoán tích cực, có nghĩa là trong 100 dữ liệu tích cực thực tế thì mô hình dự đoán đúng 90,075 dữ liệu tích cực.

Recall: chỉ số thể hiện trong tất cả các trường hợp Positive, bao nhiêu trường hợp đã được dự đoán chính xác. Recall của Cây quyết định là 94.996% nghĩa là trong 100 dự đoán tích cực thì có khoảng 94.996 dự đoán là đúng.

F_score: có một số trường hợp chúng tôi muốn tối ưu hóa cân bằng cả độ chính xác và thu hồi. Điểm F1 là giá trị trung bình hài hòa của độ chính xác và thu hồi và giúp chúng tôi tối ưu hóa một bộ phân loại cho độ chính xác cân bằng và hiệu suất thu hồi. Thời gian huấn luyện và dự đoán lâu nhất là thuật toán cây quyết định (huấn luyện 48.3s và dự đoán 328 ms), thời gian dự đoán nhanh nhất là hồi quy Logistic, thời gian huấn luyện nhanh nhất là Naïve Bayes bởi vì thuật toán này chạy dựa trên lý thuyết các biến dữ liệu độc lập với nhau. Độ chính xác cao nhất là 90% của thuật toán hồi quy Logistic, thấp nhất là Naïve Bayes với 78%. Như vậy có thể thấy Hồi quy Logistic là thuật toán tốt hơn so với các thuật toán còn lại khi xét tổng thể về tốc độ thực thi và độ chính xác

Kết quả đánh giá mô hình

Thuật toán	Decision Tree		Naïve Bayes		Hồi quy Logistic	
	Positive	Negative	Positive	Negative	Positive	Negative
Precision	90.075	84.312	77.125	98.868	88.635	92.587
Recall	94.996	71.982	99.911	20.681	98.085	67.392
F_score	92.471	77.661	87.502	34.207	93.085	78.006
Accuracy	89%		78%		90%	
Thời gian huấn luyện	48.3 s		96.3 ms		1.79 s	
Thời gian dự đoán	328 ms		24.4 ms		11 ms	

2.4 Trực quan hóa và đưa qua kết luận

2.4.1 Các từ phổ biến

```
[63] Counter(a3).most_common(10)
```

```
[('cáo', 2570),  
 ('quảng', 2389),  
 ('aba', 1950),  
 ('xăm', 1054),  
 ('xem', 867),  
 ('quan', 733),  
 ('giặt', 725),  
 ('bột', 699),  
 ('liên', 664),  
 ('làm', 529)]
```

2.4.2 WordCloud

```
[75] wordcloud = WordCloud(width = 1200, height = 1200,  
                           background_color = 'white',  
                           max_words=100,  
                           min_font_size = 2).generate(list_restopwords2)  
plt.imshow(wordcloud, interpolation='bilinear')  
plt.axis("off")  
plt.show()
```



2.4.3 Phân loại cảm xúc

```
[80] print("Negative: " + str(Negative))  
      print("Positive: " + str(Positive))  
      print("Unknow: " + str(Unknow))
```

```
Negative: 4360  
Positive: 1231  
Unknow: 2967
```

KẾT LUẬN

Nhóm chúng em đã giới thiệu cơ bản về Xử lý ngôn ngữ tự nhiên với ứng dụng trong môi trường thực tế. Tuy nhiên, Xử lý ngôn ngữ tự nhiên là rất rộng và phần tìm hiểu giới thiệu vừa rồi của chúng em vừa rồi chỉ là một mảng nhỏ.

Cụ thể, Xử lý ngôn ngữ tự nhiên có thể đưa ra rất nhiều ứng dụng khác của Xử lý ngôn ngữ tự nhiên như dịch ngôn ngữ, chatbot hay phân tích cụ thể và chuyên sâu trên các tài liệu văn bản. Ngày nay, để xử lý ngôn ngữ tự nhiên, các kỹ thuật được áp dụng chủ yếu là các thuật toán học sâu (Deep Learning) như mạng Nơ-ron (RNNs) hay mạng Long-Short Term Memory (LSTMs).

Trên đây là bài giới thiệu tổng quan về xử lý ngôn ngữ tự nhiên ứng dụng trong môi trường thực tế của quảng cáo bột giặt Abba. Với những sinh viên mới tìm hiểu như nhóm chúng em cũng muốn thêm nhiều các ví dụ về cách xử lý văn bản để phân tích cú pháp và trích xuất thông tin từ văn bản. Và bắt đầu với Deep Learning để thực hiện những thứ tốt hơn.

DANH MỤC TÀI LIỆU

1.