

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN  
ĐẠI HỌC QUỐC GIA TP.HCM



TÊN ĐỒ ÁN

Môn học: Python cho Khoa học Dữ liệu

Giảng viên hướng dẫn: ThS. Hà Minh Tuấn

Nhóm sinh viên thực hiện:

Nguyễn Thị Ngọc Anh	23280037
Trương Thị Quỳnh Giang	23280052
Trần Trung Kiên	23280066

TP.HCM, 2025

# NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

1. Mục tiêu và nội dung của đồ án

2. Kết quả đạt được

3. Ý thức làm việc của sinh viên:

TP. Hồ Chí Minh, ngày ... tháng ... năm 2025

Giảng viên hướng dẫn

ThS. Hà Minh Tuấn

# LỜI CẢM ƠN

Nội dung

# Mục lục

Bảng ký hiệu và chữ viết tắt	5
Danh sách bảng	6
Danh sách hình	7
Tóm tắt	8
Giới thiệu	9
<b>Chương 1: CƠ SỞ LÝ THUYẾT</b>	<b>12</b>
1.1 Các phương pháp vectorize dữ liệu . . . . .	12
1.1.1 TF IDF . . . . .	12
1.2 Một số phương pháp khác . . . . .	13
1.3 Các mô hình học máy . . . . .	14
1.3.1 Logistic Regression . . . . .	14
1.3.2 Support Vector Machine . . . . .	16
1.3.3 Naive Bayes . . . . .	19
<b>Chương 2: DỮ LIỆU VÀ CÔNG CỤ</b>	<b>22</b>
2.1 Giới thiệu bộ dữ liệu . . . . .	22
2.2 Khám phá bộ dữ liệu . . . . .	24
2.2.1 Kiểm tra dữ liệu trống . . . . .	24
2.2.2 Kiểm tra phân phối nhãn . . . . .	24
2.2.3 Kiểm tra độ dài chuỗi . . . . .	25
2.2.4 Kiểm tra emoji . . . . .	25
<b>Chương 3: XÂY DỰNG MÔ HÌNH PHÂN TÍCH</b>	<b>29</b>
3.1 Module Loader . . . . .	29
3.1.1 Các thư viện chính . . . . .	29
3.2 Module Preprocessor . . . . .	30
3.3 Module feature . . . . .	30
3.4 Tối ưu tham số . . . . .	30
3.4.1 Logistic Regression . . . . .	30
3.4.2 Support Vector Machine . . . . .	30
3.4.3 Naive Bayes . . . . .	30
3.5 Huấn luyện mô hình . . . . .	30
3.5.1 Logistic Regression . . . . .	30

3.5.2	Support Vector Machine . . . . .	30
3.5.3	Naive Bayes . . . . .	30
<b>Chương 4: KẾT QUẢ THỰC NGHIỆM</b>		<b>31</b>
4.1	Dánh giá mô hình . . . . .	31
4.2	So sánh mô hình . . . . .	31
4.3	Ứng dụng thực tế . . . . .	31

# Bảng ký hiệu và chữ viết tắt

# Danh sách bảng

# **Danh sách hình**

# Tóm tắt

Nhận diện và phân loại phát ngôn là một trong những bài toán quan trọng của xử lý ngôn ngữ tự nhiên, nơi hệ thống cần xác định đặc trưng và sắc thái biểu đạt của các phát ngôn được tạo ra trong môi trường trực tuyến. Các phát ngôn có thể mang nhiều dạng nội dung khác nhau như tích cực, trung tính hoặc tiêu cực, và thường đa dạng và phức tạp. Điều này khiến bài toán phân loại trở nên khó xử lý, đòi hỏi mô hình phải nắm bắt được ý nghĩa, mục đích và đặc điểm ngôn ngữ ẩn sau từng câu.

Trong lĩnh vực xử lý ngôn ngữ tự nhiên, các phương pháp học máy truyền thống giữ vai trò quan trọng nhờ tính ổn định và khả năng khai quát hóa tốt trong nhiều bài toán phân loại văn bản. Các mô hình này dựa trên việc học các đặc trưng ngôn ngữ được trích xuất từ dữ liệu, sau đó sử dụng những đặc trưng đó để suy luận và phân tách các nhóm phát ngôn khác nhau. Cách tiếp cận này cho phép mô hình nắm bắt được những tín hiệu ngữ nghĩa cốt lõi trong văn bản, hỗ trợ việc nhận diện nội dung và sắc thái biểu đạt một cách nhất quán.

Trong đồ án này, ba thuật toán phân loại phổ biến gồm Support Vector Machine (SVM), Naive Bayes và Logistic Regression được triển khai nhằm khảo sát khả năng nhận biết các kiểu tính chất thường gặp trong văn bản tiếng Việt trên không gian số. Các mô hình này đại diện cho những hướng tiếp cận phổ biến trong bài toán phân loại văn bản, nhờ vào khả năng học các đặc trưng ngôn ngữ quan trọng và đưa ra dự đoán ổn định trong quá trình phân tích.

# Giới thiệu

Phân loại phát ngôn đóng vai trò quan trọng trong việc phân tích nội dung trao đổi của người dùng trên mạng xã hội, nơi ngôn ngữ thường được thể hiện qua nhiều sắc thái và biến thể khác nhau. Một nhiệm vụ trọng yếu trong phân loại phát ngôn là nhận diện loại nội dung mà một phát ngôn thể hiện ở các mức độ khác nhau, từ toàn văn bản đến từng câu hay từng kiểu biểu đạt - nhằm xác định liệu thông điệp đó mang sắc thái tích cực, tiêu cực, trung tính hay thuộc các hình thức diễn đạt đặc thù như mỉa mai hay công kích. Nhờ đó, hệ thống có thể hỗ trợ doanh nghiệp nắm bắt phản hồi của khách hàng, theo dõi xu hướng đánh giá về sản phẩm và nhận diện sớm các chủ đề tiêu cực liên quan đến thương hiệu. Đối với các nhà nghiên cứu, việc phân loại phát ngôn còn giúp phân tích thái độ xã hội và hành vi giao tiếp trong các bối cảnh trực tuyến.

Kết quả thu được từ đồ án tạo nền tảng cho việc phát triển các hệ thống phân tích chuyên sâu hơn trong tương lai. Trên cơ sở đó, các ứng dụng như phát hiện và sàng lọc nội dung độc hại, nhận diện khuynh hướng cảm xúc hay phân tích xu hướng thảo luận trên mạng xã hội có thể được mở rộng và hoàn thiện.

## Đặt vấn đề

Sự phát triển mạnh mẽ của Internet và các nền tảng mạng xã hội đã làm gia tăng đáng kể khối lượng phát ngôn được tạo ra mỗi ngày, tạo nên nhu cầu lớn đối với việc thu thập và phân tích nội dung người dùng. Các mạng xã hội, diễn đàn thảo luận hay hệ thống bình luận trực tuyến liên tục xuất hiện những ý kiến thể hiện quan điểm, thái độ và cảm xúc của người dùng về nhiều chủ đề khác nhau.

Vai trò then chốt của việc phân loại phát ngôn:

- Đối với người dùng: Phân loại phát ngôn giúp họ nhanh chóng nhận diện xu hướng thảo luận, tiếp cận các ý kiến trước đó và đưa ra đánh giá chính xác hơn về một vấn đề, sự kiện hay sản phẩm. Người dùng có thể xem tổng quan ý kiến tích cực – tiêu cực của cộng đồng để hỗ trợ việc ra quyết định.

- Đối với tổ chức, doanh nghiệp: Các phát ngôn trên mạng xã hội là nguồn dữ liệu quan trọng giúp doanh nghiệp lắng nghe phản hồi, phát hiện các chủ đề nhạy cảm, theo dõi danh tiếng thương hiệu và điều chỉnh chiến lược truyền thông. Nhờ nắm bắt kịp thời thái độ người dùng, các tổ chức có thể nâng cao chất lượng dịch vụ, tối ưu hoạt động quản lý cộng đồng và hạn chế các xu hướng tiêu cực lan rộng.

Tuy nhiên, việc xử lý và phân loại phát ngôn trực tuyến gặp phải nhiều thách thức:

- Khối lượng dữ liệu lớn: Số lượng phát ngôn phát sinh liên tục từ nhiều nền tảng khiến việc phân tích thủ công trở nên tốn kém và thiếu hiệu quả.

- Ngôn ngữ phức tạp và đa dạng: Phát ngôn trên mạng xã hội thường mang tính chủ quan, đa dạng về cách diễn đạt và sử dụng ngôn ngữ, gây khó khăn cho việc phân tích tự động

Do đó, việc xây dựng các hệ thống phân loại phát ngôn tự động trở nên cần thiết nhằm hỗ trợ nhận diện nhanh chóng nội dung và xu hướng trao đổi trên môi trường trực tuyến. Trước khi người dùng tiếp cận một chủ đề hay doanh nghiệp tiến hành phân tích phản hồi, những hệ thống này có thể cung cấp góc nhìn tổng quát về thái độ và mức độ quan tâm của cộng đồng. Bằng cách xác định đặc trưng của từng nhóm phát ngôn, các mô hình phân loại giúp khai thác hiệu quả thông tin ngôn ngữ do người dùng tạo ra, từ đó phục vụ tốt hơn cho các hoạt động nghiên cứu và ứng dụng trong lĩnh vực phân tích dữ liệu xã hội.

## Lý do chọn đề tài

Ngày nay, cùng với sự phát triển mạnh mẽ của Internet và các nền tảng mạng xã hội, lượng thông tin và phát ngôn do người dùng tạo ra ngày càng trở nên phong phú và đa dạng. Trên các trang mạng xã hội, diễn đàn trực tuyến hay các nền tảng thảo luận công cộng, người dùng liên tục bày tỏ quan điểm, cảm xúc và thái độ của mình về nhiều vấn đề trong đời sống. Những phát ngôn này không chỉ phản ánh nhận thức và trải nghiệm cá nhân mà còn góp phần hình thành các xu hướng, ảnh hưởng đến hành vi của cộng đồng cũng như tác động trực tiếp đến doanh nghiệp và tổ chức.

Tuy nhiên, sự gia tăng nhanh chóng của khối lượng dữ liệu cùng đặc điểm ngôn ngữ thiếu chuẩn hóa trên mạng xã hội - như teencode, emoji, ký tự viết tắt hoặc các lối diễn đạt mang tính cảm xúc - khiến việc phân tích thủ công trở nên không khả thi. Trong nhiều trường hợp, các phát ngôn tiêu cực hoặc nội dung gây tranh cãi có thể lan truyền nhanh chóng, gây ảnh hưởng đến danh tiếng thương hiệu, tạo áp lực cho công tác quản lý cộng đồng và đặt ra yêu cầu cấp thiết cho các hệ thống xử lý tự động. Do đó, nhu cầu xây dựng các mô hình phân loại phát ngôn nhằm hỗ trợ nhận diện nhanh chóng nội dung, theo dõi xu hướng thảo luận và phát hiện những biểu hiện bất thường trở nên vô cùng quan trọng.

Xuất phát từ thực tiễn đó, việc nghiên cứu và xây dựng một hệ thống phân loại phát ngôn trên mạng xã hội tiếng Việt không chỉ có ý nghĩa về mặt khoa học mà còn mang tính ứng dụng cao. Hệ thống này giúp doanh nghiệp nắm bắt phản hồi khách hàng, hỗ trợ nhà quản lý giám sát tương tác trực tuyến, đồng thời là công cụ hữu ích cho các nhà nghiên cứu trong việc phân tích hành vi ngôn ngữ của cộng đồng. Với mong muốn góp phần giải quyết những thách thức nêu trên, nhóm chúng em quyết định lựa chọn đề tài “Nhận diện và phân loại phát ngôn trên mạng xã hội” làm hướng nghiên cứu và triển khai trong đồ án này.

# Phát biểu bài toán

Trong nghiên cứu này, mục tiêu chính là xây dựng và đánh giá mô hình có khả năng nhận diện và phân loại các phát ngôn được tạo ra trên mạng xã hội. Bài toán hướng đến việc xác định nhóm nội dung hoặc sắc thái biểu đạt của từng phát ngôn, với các lớp thường được quan tâm như tích cực, tiêu cực và trung tính.

Dữ liệu đầu vào của bài toán gồm gần 7.000 phát ngôn tiếng Việt được thu thập từ mạng xã hội, đã được gán nhãn cảm xúc và lưu trữ dưới dạng văn bản thô sau khi loại bỏ các yếu tố nhiễu cơ bản.

Trong đồ án này, chúng em sử dụng ba mô hình học máy truyền thông gồm Support Vector Machine (SVM), Naive Bayes và Logistic Regression để giải quyết bài toán.

## Cấu trúc đồ án

### **PHẦN 1: CƠ SỞ LÝ THUYẾT**

Trình bày các khái niệm liên quan đến bài toán xử lý ngôn ngữ tự nhiên, những đặc điểm của ngôn ngữ trên mạng xã hội, cùng các phương pháp phân loại văn bản cơ bản thường được sử dụng trong NLP.

### **PHẦN 2: THU THẬP, XỬ LÝ VÀ KHÁM PHÁ DỮ LIỆU**

Mô tả quy trình thu thập dữ liệu phát ngôn từ mạng xã hội, các bước tiền xử lý văn bản như làm sạch, chuẩn hóa, xử lý emoji, teencode và stopwords, đồng thời thực hiện phân tích khám phá dữ liệu (EDA) nhằm hiểu rõ các đặc trưng của tập dữ liệu.

### **PHẦN 3: XÂY DỰNG MÔ HÌNH PHÂN TÍCH**

Trình bày các mô hình học máy được sử dụng trong nghiên cứu gồm Support Vector Machine (SVM), Naive Bayes và Logistic Regression; mô tả cách trích xuất đặc trưng, thiết lập mô hình và quy trình huấn luyện.

### **PHẦN 4: KẾT QUẢ THỰC NGHIỆM**

Tiến hành thử nghiệm, trình bày và so sánh kết quả mô hình dựa trên các thước đo đánh giá, trực quan hóa kết quả và đưa ra nhận xét về hiệu quả phân loại phát ngôn.

# Chương 1 CƠ SỞ LÝ THUYẾT

## 1.1 Các phương pháp vectorize dữ liệu

Trong các bài toán xử lý ngôn ngữ tự nhiên (NLP), đầu vào thường là dữ liệu tồn tại dưới dạng văn bản thô (raw text). Tuy nhiên, các mô hình học máy và học sâu không thể xử lý trực tiếp dữ liệu dạng ký tự hoặc chuỗi từ, mà yêu cầu dữ liệu phải được biểu diễn dưới dạng vector số trong không gian đặc trưng. Do đó, một bước quan trọng trong quy trình xử lý NLP là vector hóa dữ liệu (data vectorization) - chuyển đổi các văn bản thành các biểu diễn số học có ý nghĩa, phản ánh được thông tin ngữ nghĩa và cú pháp của văn bản.

### 1.1.1 TF IDF

TF-IDF là một phương pháp thống kê được sử dụng rộng rãi để đánh giá mức độ quan trọng của một từ đối với một tài liệu so với một tập tài liệu lớn hơn. Phương pháp kết hợp hai thành phần:

- Tần suất thuật ngữ (TF) Đo lường tần suất xuất hiện của một từ trong tài liệu. Tần suất càng cao thì mức độ quan trọng càng lớn. Nếu một thuật ngữ xuất hiện thường xuyên trong tài liệu, thì có thể nó liên quan đến nội dung của tài liệu.

$$TF(t, d) = \frac{\text{Number of times term } t \text{ appears in document } d}{\text{Total number of terms in document } d}$$

Hình 1.1: Công thức tính TF

- Tần suất tài liệu nghịch đảo (IDF) Giảm trọng số của các từ phổ biến trong nhiều tài liệu đồng thời tăng trọng số của các từ hiếm. Nếu thuật ngữ xuất hiện trong ít tài liệu hơn, nó có nhiều khả năng có ý nghĩa và cụ thể hơn.

Sự cân bằng này cho phép TF-IDF làm nổi bật các thuật ngữ thường xuyên xuất hiện trong một tài liệu cụ thể và đặc biệt trong toàn bộ tài liệu văn bản, biến nó trở thành một công cụ hữu ích cho các tác vụ như xếp hàng tìm kiếm, phân loại văn bản và trích xuất từ khóa.

$$\text{IDF}(t, D) = \log \frac{\text{Total number of documents in corpus } D}{\text{Number of documents containing term } t}$$

Hình 1.2: Công thức tính IDF

Ưu điểm: Đơn giản và hiệu quả, dễ dàng triển khai trong các hệ thống xử lý văn bản

Hạn chế: Phương pháp không phân biệt được các từ đồng nghĩa dẫn đến bỏ sót thông tin quan trọng (ví dụ: “đẹp” và “xinh” là hoàn toàn khác nhau). Ngoài ra, độ dài của tài liệu có thể ảnh hưởng đến kết quả tính toán, đặc biệt khi các văn bản có độ dài không đồng đều.

## 1.2 Một số phương pháp khác

Bên cạnh TF-IDF, nhiều phương pháp biểu diễn văn bản khác cũng được ứng dụng trong xử lý ngôn ngữ tự nhiên bao gồm:

### a) One-hot Encoding

One-hot Encoding là phương pháp chuyển đổi các biến phân loại thành định dạng nhị phân. Nó tạo ra các cột mới cho mỗi danh mục, trong đó 1 có nghĩa là danh mục đó xuất hiện và 0 có nghĩa là không xuất hiện. Số chiều của vector này đúng bằng số từ trong từ điển. Tuy nhiên, dữ liệu này chỉ đáp ứng được khả năng huấn luyện mà chưa phản ánh được mối liên hệ về mặt ngữ nghĩa của các từ.

### b) Word2Vec

Word2Vec là mô hình học biểu diễn từ dựa trên mạng nơron nồng, nó học các vector liên tục phản ánh được mối quan hệ ngữ nghĩa và cú pháp giữa các từ trong không gian vector. Phương pháp này khắc phục được hạn chế của One-hot Encoding. Word2Vec gồm 2 kiến trúc chính: Skip-grams và CBOW (Continuous Bag of Words) - Skip-grams: đầu vào là một từ trung tâm, mô hình dự đoán các từ ngữ cảnh. - CBOW: ngược lại, đầu vào là các từ ngữ cảnh, mô hình dự đoán từ trung tâm Hạn chế của Word2Vec: - Không xử lý được đa dạng ngữ cảnh - không thay đổi theo câu. Ví dụ: “con chuột” có ngữ nghĩa khác nhau ở các ngữ cảnh khác nhau như “con chuột máy tính này thật đẹp” và “con chuột này to thật”. Word2Vec tìm ra 1 vector đại diện cho mỗi từ dựa trên tập ngữ liệu lớn nên không thể hiện được sự đa dạng của ngữ cảnh. - Không xử lý được những từ không tồn tại trong từ điển từ vựng (OOV - out of vocabulary), tức là nếu xuất hiện một từ mới, mô hình sẽ không tạo được vector.

### c) PhoBert

PhoBert là mô hình dựa trên BERT (Bidirectional Encoder Representations from Transformer) được giới thiệu vào năm 2020 và được xem là bước tiến

quan trọng cho nhiều bài toán như Question Answering, Sentiment Analysis,... Đây là một mô hình huấn luyện cho đơn ngôn ngữ (monolingual language), tức là toàn bộ quá trình huấn luyện được thực hiện trên kho dữ liệu tiếng Việt quy mô lớn, giúp mô hình nắm bắt tốt các đặc trưng ngữ pháp và ngữ nghĩa của tiếng Việt. Nó đã cải thiện được hạn chế của Word2Vec trong việc cung cấp biểu diễn theo ngữ cảnh (contextualized embeddings) - cho phép một từ có vector khác nhau tùy thuộc vào ngữ cảnh xuất hiện.

#### d) FastText

FastText là một mô hình mở rộng của Word2Vec, cung cấp các embedding cho các từ nằm ngoài từ điển. Bạn đọc có thể tham khảo thêm ở đây:

<https://www.geeksforgeeks.org/machine-learning/fasttext-working-and-implementation/>

## 1.3 Các mô hình học máy

### 1.3.1 Logistic Regression

Logistic Regression (LR) là một trong ba thuật toán phân loại phổ biến, nhằm khảo sát khả năng nhận biết các kiểu tính chất thường gặp trong văn bản tiếng Việt trên không gian số. Mô hình này đại diện cho hướng tiếp cận học máy giám sát (supervised machine learning), được lựa chọn nhờ vào khả năng học các đặc trưng ngôn ngữ quan trọng và đưa ra dự đoán ổn định trong quá trình phân tích. LR được gọi là hồi quy (regression) nhưng thực hiện phân loại (classification). LR thường được sử dụng khi biến phụ thuộc không phải là giá trị số mà là phản hồi dạng nhị phân, ví dụ như phản hồi "yes/no". Hàm Logistic Sigmoid được áp dụng cho kết quả hồi quy để lấy xác suất thuộc về một trong hai lớp. LR phân loại biến phụ thuộc vào một trong các lớp dựa trên xác suất cao hơn

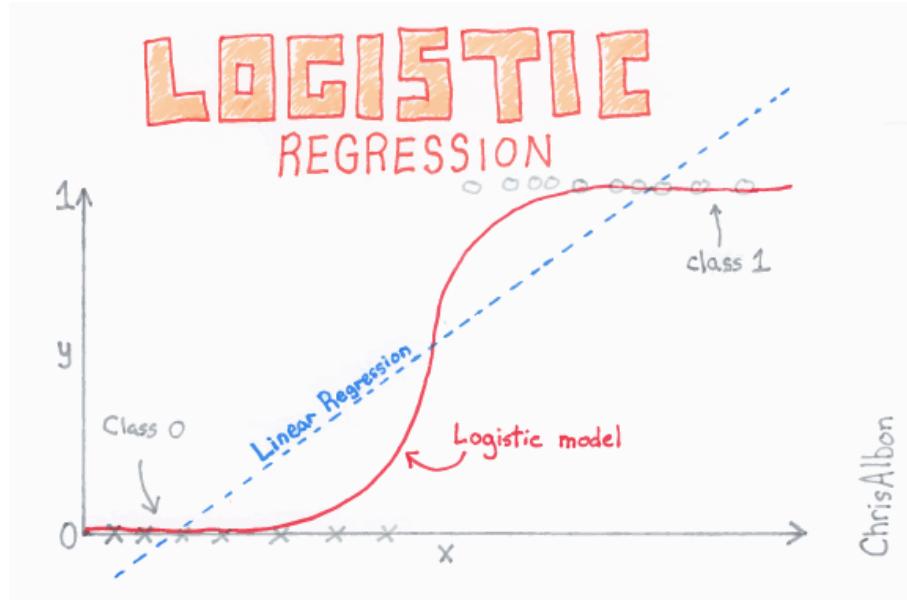
Nguồn hình ảnh: <https://builtin.com/data-science/supervised-machine-learning-classification>

#### 1. Phương pháp Tối ưu Tham số (sử dụng LogisticOptimizer)

Quá trình tối ưu hóa siêu tham số (hyperparameters) cho mô hình LR được thực hiện thông qua lớp `LogisticOptimizer`. Lớp này sử dụng phương pháp tìm kiếm lưới `GridSearchCV` kết hợp đánh giá chéo (k-fold Cross-Validation) để xác định tập siêu tham số tốt nhất. Lưới tham số được định nghĩa để dò tìm bao gồm:

- `C`: Độ mạnh của Regularization, thử nghiệm các giá trị [0.1, 1, 10, 100].
- `solver`: Thuật toán tối ưu, thử nghiệm các giá trị ['`lbfgs`', '`liblinear`'].

Thước đo được sử dụng trong `GridSearchCV` là `f1_macro`. Việc sử dụng `f1_macro` giúp đảm bảo mô hình học tốt tất cả các lớp, đặc biệt trong trường hợp dữ liệu bị mất cân bằng.



Hình 1.3: Logistic Regression

Sau khi tối ưu, các tham số tốt nhất (`best_params_`) và điểm số cao nhất (`best_score_`) được thu thập. Hàm `train_best_model` trong `LogisticOptimizer` sau đó sẽ tạo đối tượng `LogisticRegressionModel` và huấn luyện mô hình bằng các tham số tối ưu này.

## 2. Phương pháp Huấn luyện Mô hình (sử dụng `LogisticRegressionModel`)

Module `LogisticRegressionModel` triển khai quy trình huấn luyện và đánh giá mô hình LR.

- **Chia dữ liệu:** Module sử dụng phương thức `split_data` để chia tập dữ liệu đầu vào ( $X$ ,  $y$ ) thành:
  - tập huấn luyện: `_X_train`, `_y_train`
  - tập kiểm tra: `_X_test`, `_y_test`

Việc chia dữ liệu được thực hiện bằng hàm `train_test_split`. Đặc biệt, kỹ thuật phân tầng (`stratify = y`) được áp dụng để đảm bảo phân bố nhãn đồng đều giữa tập huấn luyện và tập kiểm tra.

- **Huấn luyện:** Nếu không sử dụng bộ tối ưu, mô hình sẽ được huấn luyện với các tham số mặc định:
  - `max_iter = 1000`
  - `solver = 'lbfgs'`

Khi sử dụng tham số tối ưu, mô hình `LogisticRegression` được khởi tạo bằng các giá trị tốt nhất và huấn luyện trên tập huấn luyện.

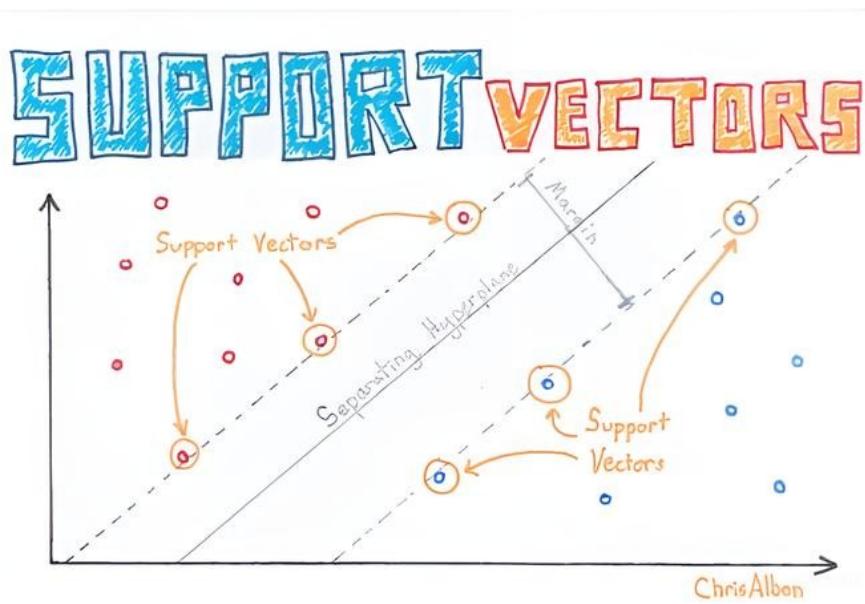
- **Đánh giá:** Sau khi huấn luyện, mô hình được đánh giá trên tập kiểm tra `_X_test`. Các thước đo đánh giá bao gồm:
  - Độ chính xác: `accuracy_score`

- Báo cáo phân loại chi tiết: `classification_report`
- Ma trận nhầm lẫn: `confusion_matrix`

Độ chính xác (Accuracy) phản ánh tỷ lệ dự đoán đúng của mô hình, trong khi ma trận nhầm lẫn (confusion matrix) cho biết chi tiết số lượng mẫu dự đoán đúng/sai cho từng lớp.

### 1.3.2 Support Vector Machine

Support Vector Machine (SVM) là mô hình học máy giám sát, mục tiêu của thuật toán là tìm ra một siêu phẳng (hyperplane) sao cho có thể phân tách tối ưu các điểm dữ liệu thuộc các lớp khác nhau. “Tối ưu” ở đây nghĩa là tìm ra siêu phẳng tạo ra khoảng cách lớn nhất (margin) giữa các lớp dữ liệu. Các điểm dữ liệu có khoảng cách nhỏ nhất đến siêu mặt phẳng (các điểm gần nhất) được gọi là các vector hỗ trợ (support vectors). Margin là khoảng không gian giữa hai siêu phẳng hỗ trợ, dải này càng rộng thì mô hình SVM càng tốt. Dải đó không có điểm dữ liệu nào nằm trong và song song hoàn toàn với siêu phẳng phân loại



Hình 1.4: Support Vector Machine

SVM không chỉ hỗ trợ phân loại nhị phân và tách các điểm dữ liệu thành hai lớp, SVM còn mở rộng để hỗ trợ các bài toán phân loại đa lớp thông qua cơ chế chia nhỏ bài toán đa lớp thành nhiều mô hình nhị phân.

#### a) Phương pháp OvO (One-vs-One)

Phương pháp OvO xây dựng bộ phân loại nhị phân cho mỗi cặp lớp trong tập dữ liệu. Với tập dữ liệu gồm  $K$  lớp, số lượng mô hình được tạo ra là:

$$N_{\text{mô hình}} = \frac{K(K - 1)}{2}.$$

Mỗi mô hình được huấn luyện để phân biệt hai lớp cụ thể. Khi dự đoán, mỗi mô hình đưa ra một phiếu bầu cho một trong hai lớp, và lớp nhận được nhiều phiếu nhất sẽ được chọn làm kết quả cuối cùng. Phương pháp OvO thường hiệu quả khi số lượng lớp không quá lớn và mỗi mô hình nhị phân tương đối nhẹ.

Ví dụ: ta có 3 lớp phân loại *Red*, *Blue* và *Green* và một mẫu mới  $x$ . Phương pháp OvO sẽ tạo ra:

$$\frac{3(3 - 1)}{2} = 3$$

mô hình nhị phân:

- Mô hình 1: *Red* vs *Blue*

Nếu mô hình dự đoán *Red*  $\Rightarrow$  *Red* nhận 1 phiếu, ngược lại *Blue* nhận 1 phiếu.

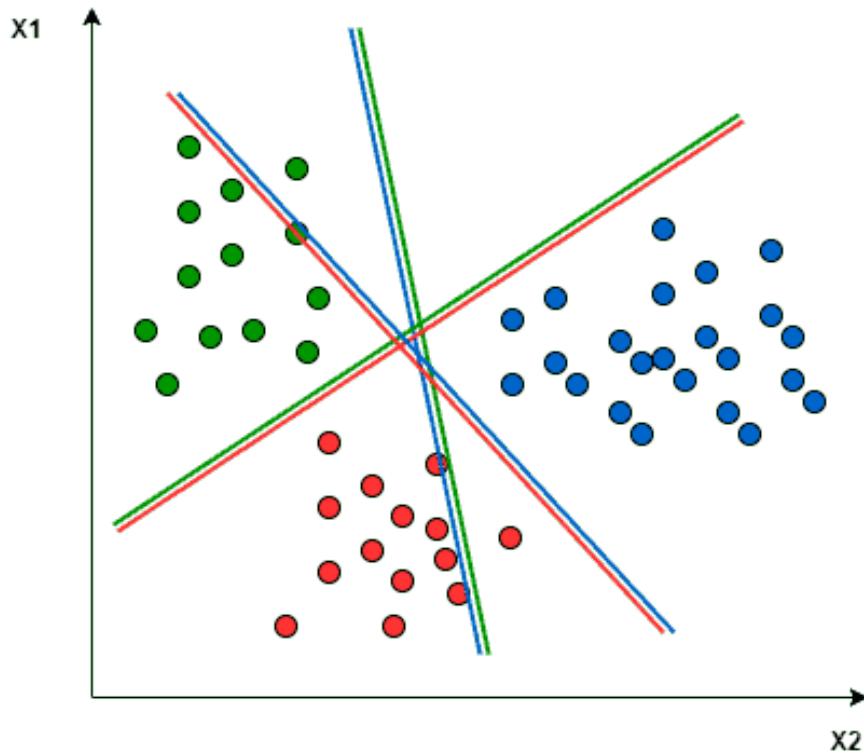
- Mô hình 2: *Red* vs *Green*

Nếu mô hình dự đoán *Red*  $\Rightarrow$  *Red* nhận 1 phiếu, ngược lại *Green* nhận 1 phiếu.

- Mô hình 3: *Blue* vs *Green*

Nếu mô hình dự đoán *Blue*  $\Rightarrow$  *Blue* nhận 1 phiếu, ngược lại *Green* nhận 1 phiếu.

Sau cùng, mẫu  $x$  được gán vào lớp có nhiều phiếu bầu nhất. Nếu xảy ra trường hợp hòa phiếu, có thể dùng *decision function* để phá hòa.



Hình 1.5: Mô phỏng phương pháp One-vs-One (OvO)

b) Phương pháp OvR (One-vs-Rest)

Phương pháp OvR (OvA) chia dữ liệu đa lớp thành  $K$  bài toán phân loại nhị phân độc lập, trong đó  $K$  là số lượng lớp. Với mỗi lớp  $C_k$ , mô hình phân loại được xây dựng để phân biệt lớp  $C_k$  với  $(K - 1)$  lớp còn lại. Mỗi mô hình con học cách nhận diện một lớp cụ thể.

Khi dự đoán một mẫu mới  $x$ , mỗi mô hình con sẽ trả về một giá trị hàm quyết định (*decision score*) — tức khoảng cách có dấu từ điểm  $x$  đến siêu phẳng phân tách của mô hình đó. Score càng lớn cho thấy mô hình càng “tự tin” rằng mẫu thuộc lớp  $C_k$ .

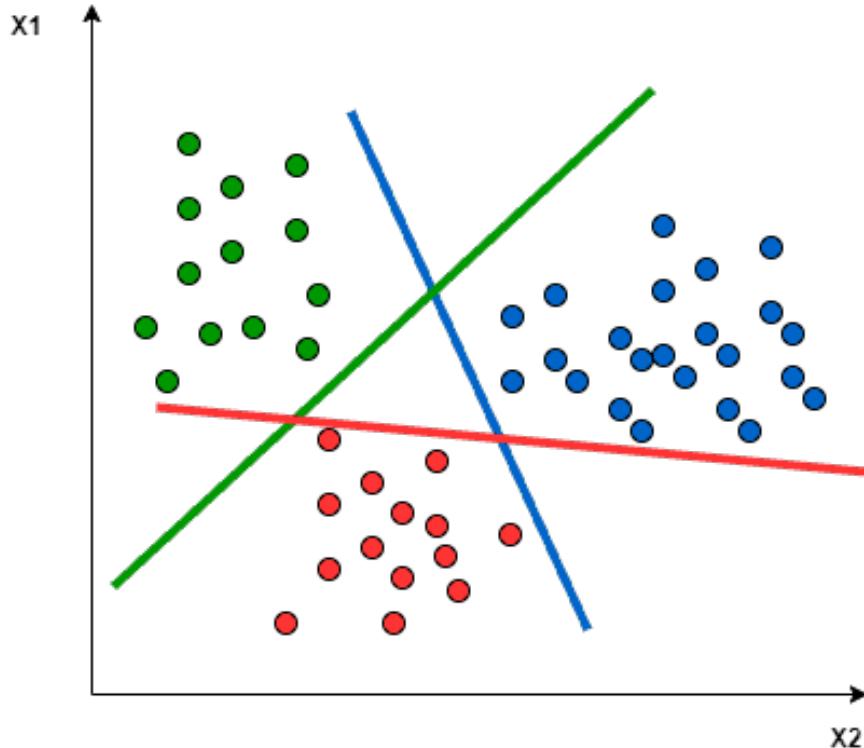
Ví dụ: ta có 3 lớp phân loại *Red*, *Blue* và *Green*, và một mẫu mới  $x$ . Phương pháp OvR sẽ tạo ra 3 mô hình con như sau:

- Mô hình 1: *Red* vs (*Blue*, *Green*) → output:  $\text{score}_{\text{Red}}(x)$
- Mô hình 2: *Blue* vs (*Red*, *Green*) → output:  $\text{score}_{\text{Blue}}(x)$
- Mô hình 3: *Green* vs (*Red*, *Blue*) → output:  $\text{score}_{\text{Green}}(x)$

Sau khi tính toán score cho mẫu mới  $x$ , lớp dự đoán được xác định theo quy tắc:

$$\hat{y} = \arg \max_k \text{score}_k(x),$$

tức là mẫu  $x$  được gán vào lớp có giá trị score cao nhất.



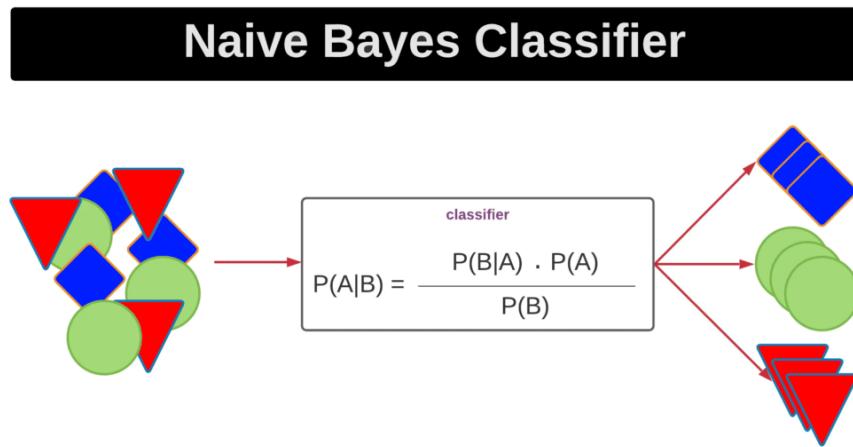
Hình 1.6: Mô phỏng phương pháp OvR

### 1.3.3 Naive Bayes

Naive Bayes là một mô hình phân loại xác suất dựa trên Định lý Bayes, hoạt động bằng cách ước lượng mức độ “khả dĩ” của từng lớp đối với mẫu dữ liệu mới dựa trên thông tin thống kê thu được từ tập huấn luyện. Nhờ cơ chế dựa trên xác suất, mô hình cho phép thực hiện phân loại một cách hiệu quả ngay cả trong không gian đặc trưng có số chiều lớn.

Cốt lõi của Naive Bayes nằm ở giả định *độc lập có điều kiện* giữa các đặc trưng: các đặc trưng  $x_1, x_2, \dots, x_n$  được xem là độc lập với nhau khi biết lớp  $C$ . Giả định này giúp đơn giản hóa quá trình ước lượng, bởi mô hình chỉ cần xem xét từng đặc trưng riêng lẻ để đánh giá mức độ phù hợp giữa mẫu và lớp, mặc dù điều này hiếm khi hoàn toàn đúng trong thực tế.

Trong quá trình dự đoán, mô hình ước lượng *xác suất tiên nghiệm*  $P(C)$  của từng lớp và *xác suất xuất hiện của mỗi đặc trưng khi biết lớp đó*  $P(x_i | C)$ . Khi một mẫu mới  $X$  được đưa vào, Naive Bayes kết hợp hai loại xác suất này để xác định xác suất hậu nghiệm  $P(C | X)$  và gán mẫu vào lớp có giá trị cao nhất, tức là lớp có khả năng giải thích tốt nhất các đặc trưng quan sát được.



Hình 1.7: Minh họa cơ chế phân loại của Naive Bayes dựa trên Định lý Bayes

Nguồn: <https://mlarchive.com/machine-learning/the-ultimate-guide-to-naive-bayes/>

Trong thực tế, việc ước lượng các xác suất  $P(x_i | C)$  được thực hiện khác nhau tùy thuộc vào bản chất của dữ liệu, thông qua các biến thể phổ biến của Naive Bayes như Bernoulli, Multinomial hoặc Gaussian.

a) Gaussian Naive Bayes

Gaussian Naive Bayes là một dạng của Naive Bayes dùng cho dữ liệu liên tục. Mô hình giả định rằng mỗi đặc trưng  $x_i$  tuân theo phân phối chuẩn (Gaussian) khi biết lớp  $\omega$ . Khi đó, xác suất có điều kiện của một đặc trưng được mô hình hoá bởi hàm mật độ Gaussian:

$$P(x_{ik} | \omega) = \frac{1}{\sqrt{2\pi\sigma_\omega^2}} \exp\left(-\frac{(x_{ik} - \mu_\omega)^2}{2\sigma_\omega^2}\right),$$

trong đó  $\mu_\omega$  và  $\sigma_\omega$  được ước lượng từ dữ liệu huấn luyện.

Dưới giả định độc lập có điều kiện, xác suất lớp có điều kiện cho mẫu nhiều chiều được tính bằng tích các xác suất thành phần:

$$P(x | \omega) = \prod_{k=1}^d P(x_{ik} | \omega).$$

Do tuân theo giả định độc lập có điều kiện giữa các đặc trưng, Gaussian Naive Bayes có thể tính toán nhanh và hiệu quả ngay cả khi số chiều dữ liệu lớn. Sau khi mô hình được huấn luyện, việc phân loại mẫu mới được thực hiện bằng cách kết hợp xác suất tiên nghiệm của từng lớp với xác suất có điều kiện nói trên, và lựa chọn lớp có xác suất hậu nghiệm lớn nhất. Nhờ cấu trúc giải tích đơn giản và quá trình dự đoán nhanh, Gaussian Naive Bayes được xem là một mô hình *eager learner* - tức mô hình học từ dữ liệu một lần, sau đó có thể dự đoán tức thời mà không cần truy cập lại toàn bộ tập huấn luyện

### b) Multinomial Naive Bayes

Multinomial Naive Bayes là một dạng của Naive Bayes được thiết kế cho dữ liệu rời rạc có dạng tần suất xuất hiện, đặc biệt phù hợp với bài toán phân loại văn bản. Mô hình sử dụng số lần một từ  $x_i$  xuất hiện trong tài liệu để ước lượng xác suất có điều kiện khi biết lớp  $\omega_j$ . Tần suất thô thường được chuẩn hoá theo độ dài tài liệu và được dùng để tính xấp xỉ tối đa hợp lý như sau:

$$\hat{P}(x_i | \omega_j) = \frac{\sum tf(x_i, d \in \omega_j) + \alpha}{\sum N_{d \in \omega_j} + \alpha V},$$

trong đó  $tf(x_i, d \in \omega_j)$  là tổng số lần xuất hiện của từ  $x_i$  trong các tài liệu thuộc lớp  $\omega_j$ ,  $V$  là kích thước từ vựng và  $\alpha$  là tham số làm trơn. Dưới giả định độc lập có điều kiện, xác suất lớp có điều kiện của một tài liệu được tính bằng:

$$P(x | \omega_j) = \prod_{i=1}^m \hat{P}(x_i | \omega_j).$$

Nhờ cách mô hình hóa dựa trên tần suất, Multinomial Naive Bayes thường đạt hiệu quả cao trong phân loại văn bản, đặc biệt khi sự khác biệt về phân bố từ giữa các lớp là rõ rệt.

### c) Bernoulli Naive Bayes

Bernoulli Naive Bayes là một dạng của Naive Bayes được thiết kế cho dữ liệu nhị phân, nơi mỗi đặc trưng cho biết một từ *xuất hiện* (1) hoặc *không xuất hiện* (0) trong tài liệu. Mỗi đặc trưng được mô hình hóa như một thử nghiệm Bernoulli, trong đó xác suất xuất hiện của từ  $x_i$  trong lớp  $\omega_j$  được ước lượng dựa trên tỉ lệ số tài liệu thuộc lớp đó có chứa từ  $x_i$ , kèm theo làm tròn Laplace để tránh xác suất bằng 0.

Mô hình tính xác suất có điều kiện của một tài liệu  $x$  theo công thức Bernoulli:

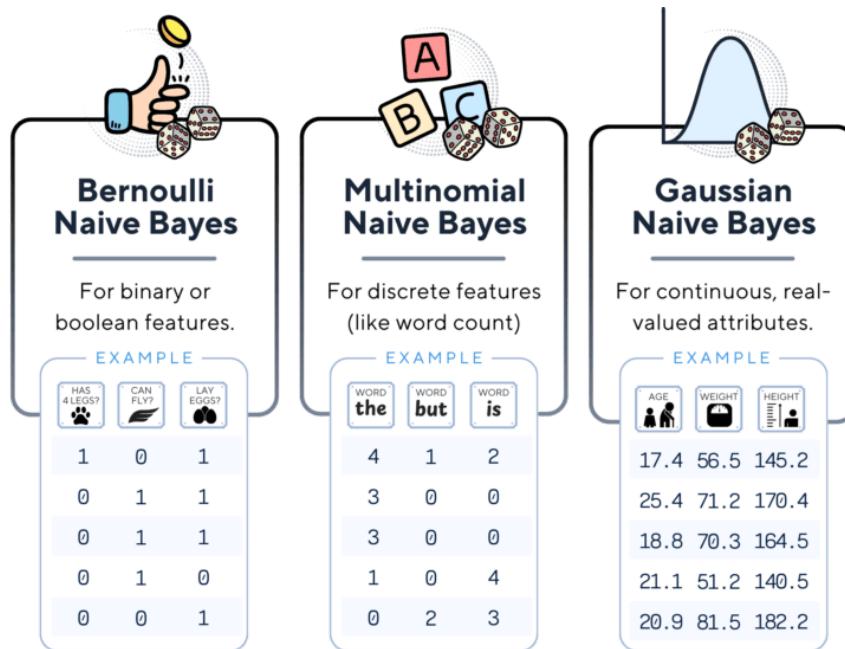
$$P(x | \omega_j) = \prod_{i=1}^m P(x_i | \omega_j)^{b_i} (1 - P(x_i | \omega_j))^{(1-b_i)}, \quad b_i \in \{0, 1\},$$

trong đó ước lượng xác suất xuất hiện của từ được tính bởi:

$$\hat{P}(x_i | \omega_j) = \frac{df_{x_i,j} + 1}{df_j + 2},$$

với  $df_{x_i,j}$  là số tài liệu thuộc lớp  $\omega_j$  có chứa từ  $x_i$ , còn  $df_j$  là tổng số tài liệu của lớp  $\omega_j$ .

Trong quá trình dự đoán, mô hình tính xác suất hậu nghiệm  $P(\omega_j | x)$  và chọn lớp có giá trị lớn nhất. Bernoulli Naive Bayes đặc biệt phù hợp cho phân loại văn bản nhị phân (chẳng hạn phát hiện spam), nơi sự xuất hiện của từ mang ý nghĩa quan trọng hơn tần suất.



Hình 1.8: Ba biến thể phổ biến của Naive Bayes: Bernoulli, Multinomial và Gaussian

Nguồn: <https://towardsdatascience.com/bernoulli-naive-bayes-explained-a-visual-guide-with-code-examples-for-beginners-aec39771ddd6/>

# Chương 2 DỮ LIỆU VÀ CÔNG CỤ

## 2.1 Giới thiệu bộ dữ liệu

Dữ liệu được thu thập từ **The UIT NLP Group** – một nhóm nghiên cứu khoa học về Xử lý Ngôn ngữ Tự nhiên (NLP) tại Trường Đại học Công nghệ Thông tin – DHQG TP. HCM. Đây là nguồn dữ liệu uy tín, đã được xây dựng và công bố phục vụ cho các nghiên cứu học thuật về phân tích văn bản tiếng Việt.

Nhóm tác giả đã xây dựng một tập dữ liệu chuẩn về **Cảm xúc trên Mạng xã hội Tiếng Việt (UIT-VSMEC)** với khoảng **6.927** câu được dán nhãn thủ công theo 7 cảm xúc:

1. Enjoyment
2. Sadness
3. Fear
4. Anger
5. Surprise
6. Disgust
7. Other

Đồng thời, nhóm nghiên cứu cũng đã tiến hành đánh giá hiệu năng của các mô hình học máy và mạng nơ-ron sâu trên bộ dữ liệu này.

Tuy nhiên, trong khuôn khổ môn học, nhóm xin phép sử dụng bộ dữ liệu UIT-VSMEC làm đầu vào và tiến hành tinh giản, điều chỉnh một số trường thông tin nhằm giảm độ phức tạp và đảm bảo phù hợp với mục tiêu bài toán phân loại của đồ án, đồng thời vẫn giữ được tính đại diện của dữ liệu gốc.

Link tài liệu gốc: <https://drive.google.com/drive/folders/1HooABJyrddVGz117fgkJ6VzkGXuWfRu>

Bạn đọc có thể tham khảo thêm bài nghiên cứu: Vong Ho, Duong Nguyen, Danh Nguyen, Linh Pham, Kiet Nguyen and Ngan Nguyen, Emotion Recognition for Vietnamese Social Media Text, 2019 16th International Conference of the Pacific Association for Computational Linguistics (PACLING 2019), October 11-13, 2019, Ha Noi, Vietnam.

Link bài nghiên cứu: <https://arxiv.org/pdf/1911.09339>

Bộ dữ liệu gồm 3 cột: Unnamed, Emotion và Sentence Dữ liệu gốc:

Bảng 2.1: Ví dụ một số câu trong bộ dữ liệu UIT-VSMEC

ID	Emotion	Sentence
941	Other	tính tao tao biết , chẳng có chuyện gì có thể làm tao phát điên cả
142	Enjoyment	lại là lao cai , tự hào quê mình quá :))
1164	Sadness	bị từ chối rồi
479	Anger	thế đường không cấm đỗ , bạn lấy quyền gì không cho người ta đỗ vậy
185	Fear	tao sợ ăn xong nóng đêm gãi toét chân
1082	Disgust	bạn không có quyền đuổi người ta . khó chịu thì gọi ca giao thông thôi . bạn đuổi họ bạn sai lè lè
1618	Surprise	nhìn mẫu này lạ mắt mà đẹp quá

Nhóm điều chỉnh nhãn phân loại như sau:

Tích cực bao gồm các nhãn *Enjoyment, Surprise*.

Tiêu cực bao gồm các nhãn *Fear, Sadness, Anger, Disgust*.

Trung tính bao gồm nhãn *Other*.

**Ghi chú:** nhãn *Surprise* có thể mang hai sắc thái tiêu cực và tích cực, tuy nhiên, qua phân tích ngữ nghĩa và tần suất trong dữ liệu, đa số các câu gán nhãn *Surprise* mang khuynh hướng cảm xúc tích cực. Do đó, nhóm quyết định gộp *Surprise* vào nhóm **Tích cực**.

Bài toán trở thành phân loại cảm xúc bình luận trên mạng xã hội tiếng Việt với 3 nhãn **Tích cực**, **Tiêu cực** và **Trung tính**. Việc gộp nhãn vẫn đảm bảo tính đại diện của dữ liệu, đồng thời giúp mô hình dễ học hơn.

Dữ liệu sau khi gộp nhãn:

Bảng 2.2: Ví dụ dữ liệu sau khi gộp nhãn

ID	Emotion	Sentence	new_emotion
941	Other	tính tao tao biết , chẳng có chuyện gì có thể làm tao phát điên cả	Trung tính
142	Enjoyment	lại là lao cai , tự hào quê mình quá :))	Tích cực
1164	Sadness	bị từ chối rồi	Tiêu cực
479	Anger	thế đường không cấm đỗ , bạn lấy quyền gì không cho người ta đỗ vậy	Tiêu cực
185	Fear	tao sợ ăn xong nóng đêm gãi toét chân	Tiêu cực
1082	Disgust	bạn không có quyền đuổi người ta . khó chịu thì gọi ca giao thông thôi . bạn đuổi họ bạn sai lè lè	Tiêu cực
1618	Surprise	nhìn mẫu này lạ mắt mà đẹp quá	Tích cực

## 2.2 Khám phá bộ dữ liệu

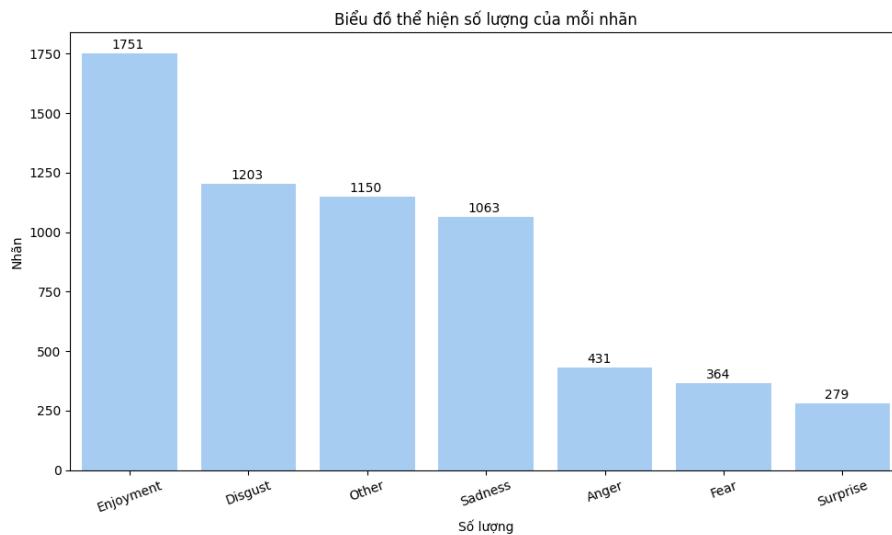
### 2.2.1 Kiểm tra dữ liệu trống

```
▶ v      data.info()
[4]   ✓  0.0s
...
...  <class 'pandas.core.frame.DataFrame'>
RangeIndex: 6241 entries, 0 to 6240
Data columns (total 3 columns):
 #   Column     Non-Null Count  Dtype  
--- 
 0   Unnamed: 0  6241 non-null   int64  
 1   Emotion    6241 non-null   object  
 2   Sentence   6241 non-null   object  
dtypes: int64(1), object(2)
memory usage: 146.4+ KB
```

Hình 2.1: Kiểm tra dữ liệu trống

Dữ liệu gồm 6241 dòng, trong đó không có dòng nào bị khuyết dữ liệu

### 2.2.2 Kiểm tra phân phối nhãn



Hình 2.2: Biểu đồ tháp hiện số lượng của mỗi nhãn dán của tập dữ liệu gốc

Như đã nói ở mục trên, nhóm quyết định gộp nhãn dán với 3 nhãn: Tích cực, Tiêu cực và Trung tính

**NHẬN XÉT:** sau khi gộp nhãn, dữ liệu đang có sự mất cân bằng giữa các nhãn, điều này sẽ khiến cho mô hình học thiên vị => cần cân bằng dữ liệu trước khi huấn luyện mô hình.

Emotion	
Enjoyment	28.056401
Disgust	19.275757
Other	18.426534
Sadness	17.032527
Anger	6.905945
Fear	5.832399
Surprise	4.470437
Name: proportion, dtype: float64	

Hình 2.3: Thống kê tỉ lệ của mỗi nhãn dán trong dữ liệu

### 2.2.3 Kiểm tra độ dài chuỗi

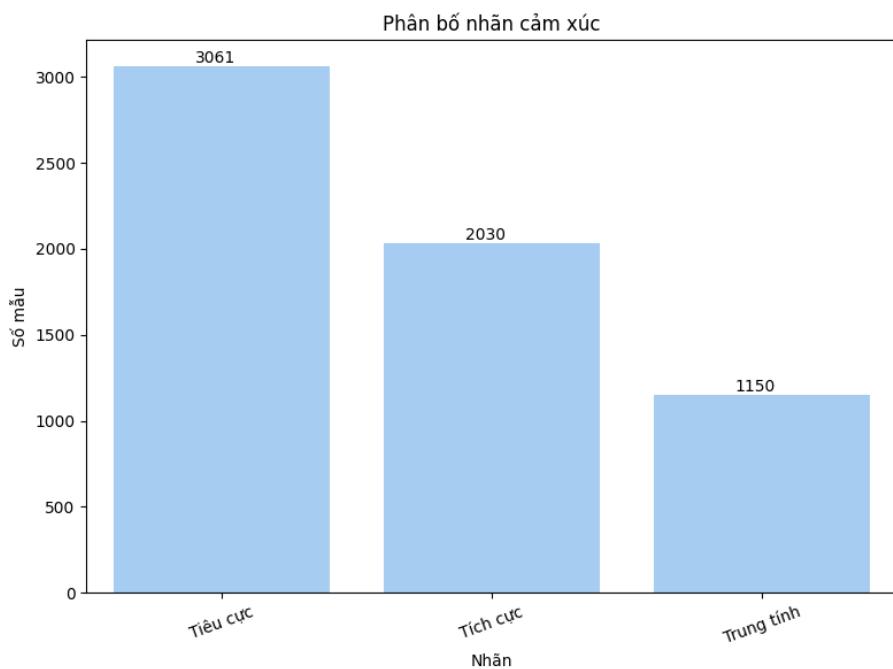
**Nhận xét:** Văn bản đầu vào có độ dài mỗi câu không tương đồng; tuy nhiên, mỗi câu đều mang ý nghĩa cảm xúc nên nhóm không thực hiện loại bỏ các câu quá ngắn (ví dụ: “im”, ...).

Tuy nhiên, để mô hình học tốt hơn, nhóm đề xuất việc lọc ra những văn bản quá dài nhưng mang nội dung *spam*, hoặc những văn bản quá ngắn và không mang giá trị cho mô hình.

Trong khuôn khổ đồ án, nhóm tạm chấp nhận rằng tất cả các văn bản đầu vào đều mang ý nghĩa cảm xúc và được giữ lại cho quá trình huấn luyện.

### 2.2.4 Kiểm tra emoji

**Nhận xét:** Trong tập dữ liệu đầu vào, các emoji xuất hiện với tần suất đáng kể (ví dụ: xuất hiện 100 lần, ...). Do đó, chúng đóng vai trò quan trọng trong việc huấn luyện mô hình. Để khai thác thông tin này, nhóm đã xây dựng tệp `emoji_dict.json` (được đính kèm cùng dữ liệu của đồ án), trong đó mỗi emoji được ánh xạ đến nhãn cảm xúc tương ứng.



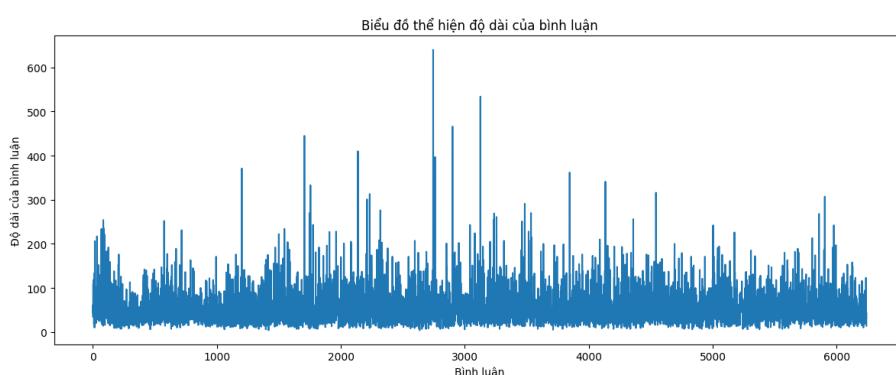
Hình 2.4: Biểu đồ thể hiện số lượng của mỗi nhãn dán sau khi gộp nhãn

```

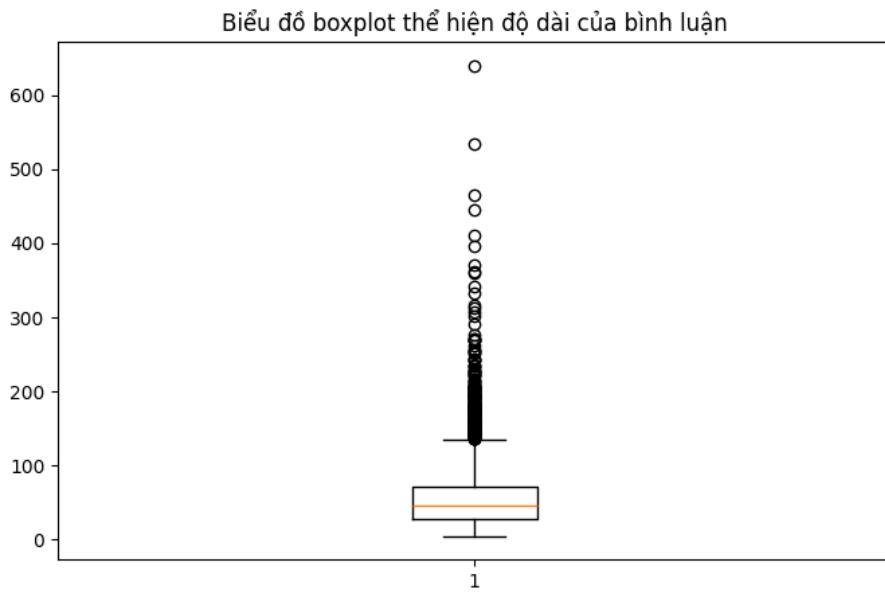
new_emotion
Tiêu cực      49.046627
Tích cực      32.526839
Trung tính    18.426534
Name: proportion, dtype: float64

```

Hình 2.5: Thống kê tỉ lệ của mỗi nhãn dán sau khi gộp



Hình 2.6: Biểu đồ thể hiện độ dài mỗi văn bản đầu vào



Hình 2.7: Biểu đồ box plot thể hiện độ dài của văn bản đầu vào

```

len_sentence = cate_data['Sentence'].apply(len)
print("Bình luận ngắn nhất là: \n{}\n với độ dài là {}".format(min(len_sentence), min(len_sentence)))
print("Bình luận dài nhất là: \n{}\n với độ dài là {}".format(max(len_sentence), max(len_sentence)))
[1]: 0.0s
... Bình luận ngắn nhất là:
      Emotion Sentence
      1417 Anger     imm
      2636 Disgust    dien
với độ dài là 4
Bình luận dài nhất là:
      Emotion Sentence
      2745 Disgust    toi ke cac ban nghe 1 chuyen co dinh 1 ty voi ...
với độ dài là 640

```

Hình 2.8: Kiểm tra văn bản dài nhất và ngắn nhất

```

Emoji
emoji_pattern = re.compile("["
    "\ud83c\udc00-\ud83c\udcff" # mặt cười
    "\ud83d\udcbb-\ud83d\udcfd" # biểu tượng
    "\ud83d\udcbb-\ud83d\udcfd" # xe cộ
    "\ud83d\udcbb-\ud83d\udcfd" # cờ
    "]+", flags=re.UNICODE)

cate_data["emoji"] = cate_data['Sentence'].apply(lambda x: emoji_pattern.findall(x))
[1]: 0.0s
... # lấy số lượng emoji có trong dữ liệu huấn luyện
emoji = set()
for x in cate_data['emoji']:
    for emo_str in x:
        for emo in emo_str: # lặp từng ký tự
            emoji.add(emo)

print(emoji)
print(len(emoji))
[1]: 79
... {'\:)', ':(', ':o', ':D', ':P', ':S', ':>', ':<', ':>>', ':<<', ':>>>', ':<<<', ':>>>>', ':<<<<', ':>>>>>', ':<<<<<', ':>>>>>>', ':<<<<<<', ':>>>>>>>', ':<<<<<<<', ':>>>>>>>>'}

```

Hình 2.9: Kiểm tra số lượng emoji khác nhau của bộ văn bản đầu vào

"\:face-with-hearts:"	: "tích cực",
"\:frowning_face:"	: "tiêu cực",
"\:grimacing_face:"	: "tiêu cực",
"\:neutral_face_with_sweat:"	: "trung tính",
"\:partying_face:"	: "tiêu cực",
"\:neutral_face:"	: "trung tính",
"\:screaming_face_with_tongue:"	: "tích cực",
"\:neutral_face_with_drooling:"	: "tiêu cực",
"\:neutral_face_with_heart_eyes:"	: "tích cực",
"\:pentagon_star::pentagon_star::pentagon_star::pentagon_star::pentagon_star:"	: "trung tính",
"\:sobbing_face:"	: "tiêu cực",

Hình 2.10: Top 10 emoji xuất hiện nhiều nhất

# Chương 3 XÂY DỰNG MÔ HÌNH PHÂN TÍCH

## 3.1 Module Loader

### 3.1.1 Các thư viện chính

Hệ thống sử dụng các thư viện mã nguồn mở sau:

- **Scikit-learn (sklearn):** Thư viện cốt lõi dùng để xây dựng các mô hình học máy như *Logistic Regression*, *Naive Bayes* và *SVM*. Thư viện cung cấp đầy đủ công cụ để:
  - trích xuất đặc trưng văn bản bằng *TF-IDF Vectorizer*;
  - tiền xử lý dữ liệu (train-test split, chuẩn hóa);
  - huấn luyện và đánh giá mô hình.

Các tham số quan trọng thường dùng:

- `C` (SVM): điều chỉnh mức phạt khi mô hình phân loại sai;
  - `kernel` (SVM): lựa chọn hàm nhân (*linear*, *rbf*, ...);
  - `alpha` (Naive Bayes): hệ số làm trơn Laplace;
  - `max_features` (TF-IDF): giới hạn số lượng đặc trưng.
- **Pandas:** Dùng để đọc dữ liệu từ các định dạng như `.csv` và `.xlsx`, đồng thời hỗ trợ thao tác linh hoạt trên *DataFrame* (lọc, gộp, thống kê, biến đổi dữ liệu).

Tham số quan trọng:

- `read_csv()`, `read_excel()`: đọc dữ liệu;
  - `dropna()`: loại bỏ hàng bị thiếu;
  - `astype()`: chuyển kiểu dữ liệu.
- **PyVi:** Thư viện xử lý tiếng Việt chuyên dụng, dùng để tách từ (*tokenization*), giúp văn bản tiếng Việt được phân chia theo cụm từ có nghĩa trước khi đưa vào mô hình. Điều này cải thiện độ chính xác của TF-IDF và mô hình phân loại.

Phương thức chính:

- `ViTokenizer.tokenize(sentence)`: tách từ trong câu đầu vào.
- **Joblib**: Dùng để tuần tự hóa (serialize) mô hình đã huấn luyện và lưu xuống đĩa, cho phép tái sử dụng mô hình mà không cần huấn luyện lại.  
Hàm quan trọng:
  - `joblib.dump(model, path)`: lưu mô hình;
  - `joblib.load(path)`: tải mô hình.
- **Matplotlib & Seaborn**: Hai thư viện trực quan hóa dữ liệu giúp vẽ các biểu đồ thống kê và ma trận nhầm lẫn (*Confusion Matrix*). Trong đồ án, nhóm sử dụng chúng để đánh giá trực quan hiệu năng mô hình.  
Các hàm phổ biến:
  - `plt.figure(figsize)`: tạo khung hình;
  - `sns.heatmap()`: vẽ ma trận nhầm lẫn;
  - `plt.xlabel()`, `plt.ylabel()`: thêm nhãn trục.
- **OpenPyXL**: Thư viện hỗ trợ đọc và ghi dữ liệu vào định dạng Excel (.xlsx), được sử dụng khi bộ dữ liệu ban đầu ở dạng Excel hoặc khi cần xuất dữ liệu đã xử lý.  
Hàm quan trọng:
  - `load_workbook()`: đọc file Excel;
  - `worksheet.append()`: thêm dòng dữ liệu;
  - `workbook.save()`: lưu tệp Excel.

## 3.2 Module Preprocessor

## 3.3 Module feature

## 3.4 Tối ưu tham số

### 3.4.1 Logistic Regression

### 3.4.2 Support Vector Machine

### 3.4.3 Naive Bayes

## 3.5 Huấn luyện mô hình

### 3.5.1 Logistic Regression

### 3.5.2 Support Vector Machine

### 3.5.3 Naive Bayes

# **Chương 4 KẾT QUẢ THỰC NGHIỆM**

**4.1 Đánh giá mô hình**

**4.2 So sánh mô hình**

**4.3 Ứng dụng thực tế**