

TRƯỜNG ĐẠI HỌC SÀI GÒN
KHOA CÔNG NGHỆ THÔNG TIN

Khoa học máy mình



Môn học: Học máy

Đề tài: Phân Loại Cảm Xúc Trên Ngữ Liệu Tiếng Anh

Giảng viên hướng dẫn. TS Nguyễn Thanh Phước

Sinh viên thực hiện

MSSV
3122410040
3122410185

Họ và tên
Đặng Văn Chiến
Nguyễn Anh Khoa

Thành phố Hồ Chí Minh, tháng 12 năm 2024

Mục Lục

Phần 1: Định nghĩa bài toán	6
1. Lý do chọn đề tài	6
2. Mục tiêu nghiên cứu	6
3. Phạm vi nghiên cứu	7
4. Các bài toán tương tự trong thực tế	7
Phần 2: Cơ sở lý thuyết	7
1. Tổng quan về machine learning	7
1.1. Machine learning là gì ?	7
1.2. Các phương pháp học trong machine learning	8
2. Các thuật toán học máy áp dụng	10
2.1. Linear Support Vector Classifier (Linear SVC)	10
2.2. Logistic Regression	11
2.3. Naive Bayes	12
2.4. Random Forest	13
3. Phương pháp trích xuất đặc trưng TF- IDF	14
Phần 3: Khám phá dữ liệu	16
1. Nguồn gốc của bộ dữ liệu	16
2. Tổng quan về bộ dữ liệu	16
2.1. Các giá trị trong biến mục tiêu label	17
3. Tiền xử lý dữ liệu	19
3.1. Kiểm tra dữ liệu trùng lặp	19
3.2. Kiểm tra dữ liệu thiếu	19
3.3. Xóa bỏ các đường dẫn URLs	19
3.4. Xóa các ký hiệu đặc biệt và dấu câu	20
3.5. Loại bỏ các khoảng trắng dư thừa	20
3.6. Xóa các giá trị số trong văn bản	21
3.7. Biến đổi văn bản sang chữ thường	21
3.8. Xóa các stop từ	22
4. Một số từ thường gặp trong các cảm xúc	23
4.1. Một số từ thường gặp với ‘sadness’	23
4.2. Một số từ thường gặp với ‘joy’	24
4.3. Một số từ thường gặp với “love”	24

4.4. Một số từ thường gặp trong ‘anger’	25
4.5. Một số từ thường gặp với ‘fear’	26
4.6. Một số từ thường gặp với ‘surprise’	27
Phần 4 Huấn luyện mô hình	27
1. Chia dữ liệu	27
2. Phương pháp đánh giá.....	28
3. Mô hình 1 - Linear Support Vector Classifier (SVC)	29
4. Mô hình 2 - Logistic Regression.....	31
5. Mô hình Naïve Bayes Classifier	34
6. Mô hình randomforest.....	36
7. So sánh các mô hình:	38
Phần 5: Tổng kết	39
1. Tóm tắt	39
2. Hạn chế và hướng phát triển	40
3. Kết luận tổng quát	41
Tài liệu tham khảo:	41

Danh sách hình ảnh

Hình 1: Giá trị ngẫu nhiên trong dataset	17
Hình 2: Kích thước dataset.....	17
Hình 3 Biểu đồ phân phối và số lượng giá trị trong biến mục tiêu label	18
Hình 4: Kiểm tra dữ liệu trùng lặp	19
Hình 5: Kiểm tra dữ liệu thiếu.....	19
Hình 6: Xóa URLs.....	20
Hình 7: Xóa ký tự đặc biệt và dấu câu	20
Hình 8: Loại bỏ khoảng trắng dư thừa	21
Hình 9: Xóa số	21
Hình 10: Chuyển đổi chữ thường	21
Hình 11: Import thư viện nltk.....	22
Hình 12: Xóa stop words	22
Hình 13: 20 từ thường gặp với 'sadness'.....	23
Hình 14: 20 từ thường gặp với 'joy'.....	24
Hình 15: 20 từ thường gặp với 'love'.....	25
Hình 16: 20 từ thường gặp với 'fear'.....	26
Hình 17: 20 từ thường gặp với 'surprise'	27
Hình 18: Chia dữ liệu tập train, test	28
Hình 19: Kích thước tập train, test	28
Hình 20: Mô hình Linear Support Vector Classifier (SVC).....	29
Hình 21: Kết quả thực thi của SVC.....	30
Hình 22: Điều chỉnh các tham số của SVC	31
Hình 23: Kết quả sau khi điều chỉnh tham số của SVC	31
Hình 24: Mô hình Logistic regression.....	32
Hình 25: Kết quả thực thi của Logistic regression	32
Hình 26: Điều chỉnh tham số của Logistic regression.....	33
Hình 27: Kết quả sau khi điều chỉnh tham số của Logistic regression	34
Hình 28: Mô hình Naive Bayes Classifier.....	34
Hình 29: Kết quả thực thi của Naive Bayes Classifier	35
Hình 30: Điều chỉnh mô hình Naive Bayes Classifier	36
Hình 31: Kết quả sau khi điều chỉnh của Naive Bayes Classifier	36
Hình 32: Mô hình Random Forest.....	37
Hình 33: Kết quả thực thi của Random Forest	37
Hình 34: Kết quả sau khi điều chỉnh của Random Forest	38
<i>Hình 1: Điểm hiệu suất F1-score của các mô hình</i>	

Phần 1: Định nghĩa bài toán

1. Lý do chọn đề tài

Trong thời đại số hóa hiện nay, lượng dữ liệu văn bản được tạo ra mỗi ngày là vô cùng lớn từ các nền tảng xã hội, diễn đàn trực tuyến, tin tức, đánh giá sản phẩm, v.v. Những văn bản này không chỉ chứa thông tin, mà còn phản ánh cảm xúc, thái độ và quan điểm của con người. Phân loại cảm xúc văn bản là một bài toán trong lĩnh vực học máy đặc biệt là trong xử lý ngôn ngữ tự nhiên (NLP), nơi mục tiêu là xác định cảm xúc từ văn bản.

Ứng dụng việc phân loại cảm xúc trong văn bản, đặc biệt là tiếng Anh, một ngôn ngữ chung của thế giới đã được ứng dụng trong nhiều lĩnh vực như: quản trị doanh nghiệp, quản trị thương hiệu sản phẩm, quản trị quan hệ khách hàng, khảo sát ý kiến khách hàng hay dễ hiểu hơn là phân tích đánh giá, ý kiến phản hồi của khách hàng về một sản phẩm, Việc dự đoán là vô cùng quan trọng vì ý kiến, đánh giá của khách hàng ngày càng trở nên có giá trị thiết thực hơn.

2. Mục tiêu nghiên cứu

Mục tiêu của bài nghiên cứu này là tìm hiểu các lý thuyết cần thiết, để xây dựng được mô hình giải quyết bài toán nhận diện cảm xúc người dùng qua các ý kiến đánh giá, phản hồi ...qua đó mô hình sẽ được huấn luyện để nhận dạng và phân loại các cảm xúc cơ bản như:

- Buồn (Sadness)
- Vui vẻ (Joy)
- Yêu (Love)
- Giận dữ (Anger)
- Bất ngờ (Surprise)

Bên cạnh đó, mô hình giải quyết bài toán nhận diện cảm xúc trong văn bản tiếng Anh phải được tối ưu về độ chính xác, hiệu suất thời gian thực hiện, giúp giải quyết các vấn đề còn mắc phải trong nhận diện cảm xúc khách hàng nói riêng và xử lý ngôn ngữ tự nhiên ở Việt Nam nói chung.

3. Phạm vi nghiên cứu

Phạm vi nghiên cứu: nhận diện cảm xúc trong văn bản tiếng Anh với các đặc trưng trong các phản hồi, bình luận, các bài đăng trên tweet. Từ kết quả nhận các cảm xúc xây dựng mô hình nhận diện cảm xúc cơ bản cho tiếng Anh.

4. Các bài toán tương tự trong thực tế

- Phân loại thư rác (Spam Email Detection)

Bài toán: Với sự gia tăng của các email quảng cáo và lừa đảo, việc phân loại thư rác là một trong những bài toán quan trọng. Hệ thống cần phân biệt được email hợp lệ (ham) và email rác (spam).

- Dự đoán nguy cơ bệnh tật (Medical Diagnosis)

Bài toán: Trong y tế, việc dự đoán nguy cơ mắc bệnh dựa trên các chỉ số sinh học và tiền sử bệnh nhân là một bài toán quan trọng. Ví dụ, dự đoán bệnh tiểu đường, tim mạch hoặc ung thư.

- Phân tích nhu cầu vay vốn của khách hàng (Loan Approval Prediction)

Bài toán: Các ngân hàng và tổ chức tín dụng cần xác định xem liệu khách hàng có đủ điều kiện được vay vốn hay không, dựa trên hồ sơ tài chính và thông tin cá nhân.

Phần 2: Cơ sở lý thuyết

1. Tổng quan về machine learning

1.1. Machine learning là gì ?

Machine Learning (ML) là một nhánh quan trọng của trí tuệ nhân tạo (Artificial Intelligence - AI), tập trung vào việc phát triển các hệ thống có khả năng tự học hỏi và cải thiện hiệu suất thông qua kinh nghiệm mà không cần lập trình cụ thể cho từng nhiệm vụ. Thay vì được lập trình bằng các quy tắc tĩnh, các thuật toán

Machine Learning phân tích dữ liệu, rút ra các mẫu và thực hiện dự đoán hoặc đưa ra quyết định dựa trên dữ liệu đầu vào.

Trong Machine Learning, các nhiệm vụ được phân loại theo các phương thức học tiếp nhận hoặc phản hồi về được học, được cung cấp cho hệ thống được phát triển.

1.2. Các phương pháp học trong machine learning

1.2.1. Học có giám sát (*Supervised learning*)

Học có giám sát là mô hình học từ một tập dữ liệu đã được gán nhãn rõ ràng (label). Mỗi dữ liệu đầu vào sẽ đi kèm với kết quả đầu ra mong muốn (label). Mục tiêu của mô hình là học mối quan hệ giữa đầu vào và đầu ra để có thể dự đoán đúng nhãn cho dữ liệu mới.

Các loại bài toán chính trong học có giám sát:

- Phân loại (Classification): Dự đoán nhãn rời rạc, như phân loại email là spam hay không spam, nhận dạng hình ảnh.
- Hồi quy (Regression): Dự đoán giá trị liên tục, như dự báo giá nhà, dự đoán nhiệt độ.

Ví dụ: Với việc học có giám sát, một thuật toán có thể được cung cấp dữ liệu với hình ảnh của con mèo được dán nhãn là 'cat' và hình ảnh của con chó được dán nhãn là 'dog'. Bằng cách được huấn luyện về dữ liệu này, thuật toán học có giám sát sẽ có thể sau đó xác định hình ảnh con mèo, con chó không dán nhãn là 'cat' hay 'dog'. Một số ví dụ khác như dự đoán cảm xúc trong văn bản dựa trên nhãn cảm xúc (vui, buồn, tức giận, v.v.), phân loại email là "spam" hay "không spam".

1.2.2. Học không giám sát (*Unsupervised learning*)

Học không giám sát là phương pháp học máy trong đó hệ thống được cung cấp dữ liệu đầu vào mà không có bất kỳ nhãn hay kết quả đầu ra nào được gán sẵn.

Mục tiêu chính của phương pháp này là khám phá các cấu trúc, mẫu và mối quan hệ ẩn trong dữ liệu.

Các kỹ thuật chính của học không giám sát bao gồm:

- Phân cụm (Clustering): Nhóm các điểm dữ liệu tương tự vào các nhóm khác nhau. Ví dụ như thuật toán K-means được sử dụng để: Nhóm khách hàng theo hành vi mua sắm hay phân loại văn bản theo chủ đề.
- Giảm chiều dữ liệu (Dimensionality Reduction): Giúp giảm số lượng các đặc trưng (features) của dữ liệu đồng thời giữ lại thông tin quan trọng nhất. Các kỹ thuật như Principal Component Analysis (PCA) được sử dụng rộng rãi trong việc này.

1.2.3. Học bán giám sát (Semi-Supervised learning)

Phương pháp học kết hợp giữa học có giám sát và học không giám sát. Tập dữ liệu bao gồm một phần nhỏ dữ liệu được gán nhãn và một phần lớn dữ liệu không được gán nhãn.

Điều này sẽ giúp giảm chi phí và thời gian gán nhãn dữ liệu. Tập dữ liệu được lượng lớn dữ liệu chưa được gán nhãn ứng dụng.

Ví dụ như: Nhận dạng hình ảnh y tế với số lượng ảnh được gán nhãn hạn chế.

1.2.4. Học tăng cường (Reinforcement learning)

Là phương pháp học dựa trên mô hình tương tác giữa tác nhân (agent) và môi trường. Tác nhân học cách thực hiện các hành động để đạt được mục tiêu thông qua cơ chế thưởng và phạt.

Các đặc điểm chính:

- Tác nhân học thông qua việc thử và sai, nhận được phản hồi từ môi trường.

- Mục tiêu là tối đa hóa tổng phần thưởng nhận được.
- Không có tập dữ liệu huấn luyện ban đầu, mà học từ trải nghiệm trực tiếp.

Ví dụ điển hình như: Các hệ thống chơi game như AlphaGo của Google DeepMind, điều khiển robot,...

Trong báo cáo này, chúng em sẽ tập trung xây dựng mô hình để phân loại cảm xúc văn bản tiếng Anh bằng cách sử dụng các phương pháp học máy. Dữ liệu sử dụng sẽ được phân tích kỹ lưỡng, sau đó áp dụng các mô hình học máy như Logistic Regression, Random Forest và các kỹ thuật hiện đại khác để đánh giá hiệu quả. Qua đó, chúng em hy vọng làm rõ hơn về khả năng và hạn chế của các phương pháp hiện tại trong việc giải quyết bài toán.

2. Các thuật toán học máy áp dụng

Trong nghiên cứu này, chúng em sẽ tập trung xây dựng mô hình để phân loại cảm xúc văn bản tiếng Anh nên chúng em sẽ sử dụng bốn thuật toán phân loại phổ biến trong lĩnh vực học máy, bao gồm: Linear Support Vector Classifier (Linear SVC), Logistic Regression, Random Forest, và Naive Bayes. Mỗi thuật toán này đều có những ưu điểm và hạn chế riêng.

2.1. Linear Support Vector Classifier (Linear SVC)

Linear SVC là một biến thể của Support Vector Machine (SVM), được thiết kế đặc biệt để giải quyết các bài toán phân loại tuyến tính. Thuật toán này hoạt động bằng cách tìm kiếm một siêu phẳng tối ưu để phân tách các lớp dữ liệu, sao cho khoảng cách giữa siêu phẳng và các điểm dữ liệu gần nhất là lớn nhất. Khoảng cách này được gọi là margin, và việc tối đa hóa margin giúp tăng khả năng tổng quát hóa của mô hình, tức là khả năng dự đoán chính xác trên dữ liệu mới.

Ưu điểm:

- Hiệu quả với dữ liệu tuyến tính: Linear SVC hoạt động rất tốt nếu dữ liệu có thể phân tách bằng một siêu phẳng.
- Đơn giản và nhanh chóng: Tốc độ huấn luyện nhanh hơn so với SVM phi tuyến, đặc biệt với dữ liệu lớn.
- Giảm nguy cơ overfitting: Linear SVC thường áp dụng regularization để tránh overfitting.

Nhược điểm:

- Không xử lý tốt dữ liệu phi tuyến: Nếu dữ liệu không thể phân tách tuyến tính, Linear SVC không phù hợp.
- Nhạy cảm với outliers: Các điểm ngoại lệ có thể làm thay đổi siêu phẳng phân tách.

Linear SVC thường được sử dụng trong các bài toán phân loại văn bản, hình ảnh, và dữ liệu có số chiều cao.

2.2. Logistic Regression

Logistic Regression là một thuật toán hồi quy tuyến tính được sử dụng để dự đoán xác suất của một lớp. Thuật toán này sử dụng hàm sigmoid để ánh xạ đầu vào thành một giá trị nằm trong khoảng từ 0 đến 1, đại diện cho xác suất của một lớp. Logistic Regression thường được sử dụng trong các bài toán phân loại nhị phân, nhưng cũng có thể được mở rộng cho các bài toán phân loại đa lớp.

Ưu điểm:

- Dễ hiểu và dễ triển khai: Logistic Regression có nền tảng toán học đơn giản và kết quả dễ diễn giải.
- Hiệu quả: Mô hình hoạt động tốt với dữ liệu tuyến tính và ít nhiễu.

- Khả năng xác suất: Dự đoán đầu ra dưới dạng xác suất, hữu ích trong nhiều bài toán thực tế.
- Tài nguyên tính toán thấp: Thích hợp cho bài toán vừa và nhỏ với dữ liệu không phức tạp.

Nhược điểm:

- Không xử lý tốt quan hệ phi tuyến tính: Logistic Regression không thể nắm bắt mối quan hệ phi tuyến giữa các đặc trưng.
- Dễ bị ảnh hưởng bởi outliers: Điểm dữ liệu ngoại lệ có thể làm sai lệch mô hình.

2.3. Naive Bayes

Naive Bayes là một thuật toán phân loại dựa trên định lý Bayes, giả định rằng các đặc trưng độc lập với nhau. Thuật toán này tính toán xác suất của một lớp dựa trên xác suất tiên nghiệm của lớp đó và xác suất có điều kiện của các đặc trưng.

Ưu điểm:

- Nhanh chóng và hiệu quả: Tính toán đơn giản dễ thực hiện, không cần nhiều tài nguyên.
- Hoạt động tốt trên dữ liệu nhỏ: Đặc biệt hữu ích khi dữ liệu có số lượng mẫu ít.
- Khả năng mở rộng: Phù hợp với bài toán phân loại nhiều lớp.

Nhược điểm:

- Giả định độc lập: Trong thực tế, các đặc trưng thường không độc lập, dẫn đến giảm độ chính xác.

- Phụ thuộc vào phân phối dữ liệu: Hiệu quả giảm nếu dữ liệu không tuân theo phân phối phù hợp.

2.4. Random Forest

Random Forest là một thuật toán học máy thuộc nhóm ensemble learning, sử dụng nhiều cây quyết định (decision trees) để cải thiện độ chính xác và giảm nguy cơ overfitting.

Ưu điểm:

- Khả năng xử lý dữ liệu phức tạp: Random Forest hiệu quả với dữ liệu phi tuyến tính và nhiều chiều.
- Độ chính xác cao: Do sử dụng nhiều cây và cơ chế ensemble.
- Khả năng kháng nhiễu: Giảm tác động của các đặc trưng không liên quan hoặc nhiễu.
- Tầm quan trọng của đặc trưng: Có thể đánh giá được mức độ quan trọng của các đặc trưng.

Nhược điểm:

- Khó giải thích: Kết quả từ Random Forest thường khó diễn giải do là tập hợp nhiều mô hình.
- Tài nguyên tính toán cao: Yêu cầu tài nguyên lớn hơn so với Logistic Regression, đặc biệt với dữ liệu lớn.

3. Phương pháp trích xuất đặc trưng TF- IDF

TF-IDF là gì? Tf-Idf là từ viết tắt của Term Frequency (tần suất xuất hiện của từ) – Inverse Document Frequency (tần suất nghịch đảo văn bản). Đây là một kỹ thuật sử dụng để chuyển đổi văn bản thành biểu diễn số (vector) dựa trên tần suất và độ quan trọng của các từ trong tập dữ liệu.

Vậy tần suất xuất hiện của từ là gì?

Term Frequency (tần suất xuất hiện) là con số thể hiện thuật ngữ đó xuất hiện bao nhiêu lần trong tài liệu này. Nếu thuật ngữ đó xuất hiện càng nhiều thì trọng số càng cao. Có thể hiểu đơn giản là khi một từ được nhắc tới 5 lần thì từ đó sẽ có khả năng liên quan hơn so với một nội dung chỉ nhắc tới từ đó 1 lần.

TF- *term frequency* – tần số xuất hiện của 1 từ trong 1 văn bản. Công thức tính:

$$tf(t, d) = \frac{f(t, d)}{\max\{f(w, d) : w \in d\}}$$

- Thương của số lần xuất hiện 1 từ trong văn bản và số lần xuất hiện nhiều nhất của một từ bất kỳ trong văn bản đó. (giá trị sẽ thuộc khoảng $[0, 1]$)
- **$f(t, d)$** – số lần xuất hiện từ t trong văn bản d .
- **$\max\{f(w, d) : w \in d\}$** – số lần xuất hiện nhiều nhất của một từ bất kỳ trong văn bản.

Tần suất nghịch đảo văn bản

IDF – *inverse document frequency* - tần số nghịch của 1 từ trong tập văn bản.

Tính *IDF* để giảm giá trị của những từ phổ biến. Mỗi từ chỉ có 1 giá trị *IDF* duy nhất trong tập văn bản.

$$\text{idf}(t, D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$$

- $|D|$: tổng số từ trong tập D .
- $|\{d \in D : t \in d\}|$: số văn bản chứa từ nhất định, với điều kiện t xuất hiện trong văn bản d (i.e., $\text{tf}(t, d) > 0$). Nếu từ đó không xuất hiện ở bất cứ 1 văn bản nào trong tập thì mẫu số sẽ bằng 0 \Rightarrow phép chia cho không không hợp lệ, vì thế người ta thường thay bằng mẫu thức $1 + |\{d \in D : t \in d\}|$.

Cơ số logarit trong công thức này không thay đổi giá trị của 1 từ mà chỉ thu hẹp khoảng giá trị của từ đó. Vì thay đổi cơ số sẽ dẫn đến việc giá trị của các từ thay đổi bởi một số nhất định và tỷ lệ giữa các trọng lượng với nhau sẽ không thay đổi (nói cách khác, thay đổi cơ số sẽ không ảnh hưởng đến tỷ lệ giữa các giá trị *IDF*). Tuy nhiên việc thay đổi khoảng giá trị sẽ giúp tỷ lệ giữa *IDF* và *TF* tương đồng để dùng cho công thức *TF-IDF* như bên dưới.

Giá trị **TF-IDF**:

$$\text{tfidf}(t, d, D) = \text{tf}(t, d) \times \text{idf}(t, D)$$

Những từ có giá trị *TF-IDF* cao là những từ càng có liên quan trong tài liệu cụ thể.

Ví dụ về TF-IDF:

Hãy xem xét 1 tài liệu dài 100 từ trong đó từ *SEO* xuất hiện 3 lần. Tần suất xuất hiện của từ khóa *SEO* (tức *TF*) là $(3/100) = 0,03$.

Bây giờ, giả sử chúng ta có 10 triệu tài liệu và từ SEO xuất hiện trong 1000 tài liệu. Khi đó tần số nghịch đảo văn bản (tức là IDF) được tính là $\log(10.000.000/1000) = 4$.

Do đó, trọng số TF-IDF là $0,03 * 4 = 0,12$.

Phần 3: Khám phá dữ liệu

1. Nguồn gốc của bộ dữ liệu

Trong bài nghiên cứu này chúng em sử dụng bộ dữ liệu Emotions Dataset, đây là một tập dữ liệu công khai có sẵn trên nền tảng Kaggle [<https://www.kaggle.com/datasets/nelgiriyeewithana/emotions>], bộ dữ liệu được thu thập dựa trên nội dung của các bài post, bình luận, tin nhắn trên nền tảng Twitter bởi tác giả **Nidula Elgiriyeewithana**. Bộ dữ liệu được có số điểm vote là 309 và hơn 13.000 lượt tải về sử dụng trên kaggle.[1]

2. Tổng quan về bộ dữ liệu

Bộ dữ liệu gồm 416809 dòng dữ liệu, với 2 cột chính là “text” chứa nội dung của các bài trong “twitter” và cột “label” chứa các nhãn cảm xúc tương ứng:

- 0 : Buồn
- 1 : Vui vẻ
- 2 : Yêu
- 3 : Giận dữ
- 4 : Sợ hãi
- 5 : Bất ngờ

Ví dụ về bộ dữ liệu:

Unnamed: 0		text	label
0	0	i just feel really helpless and heavy hearted	4
1	1	ive enjoyed being able to slouch about relax a...	0
2	2	i gave up my internship with the dmrp and am f...	4
3	3	i dont know i feel so lost	0
4	4	i am a kindergarten teacher and i am thoroughl...	4

Hình 2: Giá trị ngẫu nhiên trong dataset

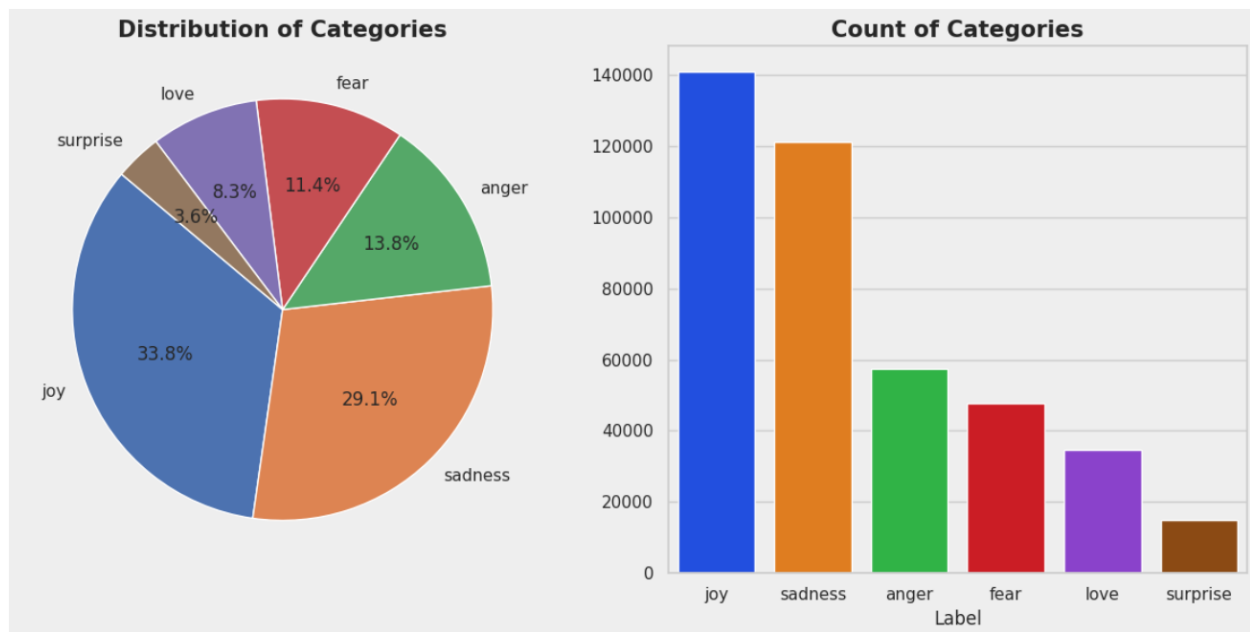
```
mydata.shape
```

```
(416809, 3)
```

Hình 3: Kích thước dataset

Bộ dữ liệu có 416809 dòng dữ liệu và 3 cột dữ liệu.

2.1. Các giá trị trong biến mục tiêu label



Hình 4 Biểu đồ phân phối và số lượng giá trị trong biến mục tiêu label

Qua 2 biểu đồ trong hình trên có thấy:

Nhãn joy chiếm tỷ lệ lớn nhất, khoảng 33.8% có số lượng mẫu hơn 14000 mẫu.

Nhãn 'sadness' chiếm tỷ lệ lớn thứ 2, khoảng 29.1% có số lượng hơn 12000 mẫu.

Các nhãn còn lại (2, 3, 4, 5) chiếm tỷ lệ nhỏ hơn, đặc biệt nhãn 5 chỉ chiếm 3.6%, cho thấy dữ liệu về nhãn này rất ít.

Điều này cho thấy dữ liệu có sự mất cân đối, khi nhãn "joy" và "sadness" chiếm ưu thế trong tập dữ liệu. Trong khi các nhãn khác chiếm tỷ lệ nhỏ hơn. Điều này có thể dẫn đến các mô hình học máy tập trung vào nhãn phổ biến, bỏ qua các nhãn ít dữ liệu.

Tỷ lệ cao của "joy" và "sadness" có thể phản ánh rằng các cảm xúc này xuất hiện phổ biến hơn và "Surprise" xuất hiện rất ít, có thể là do cảm xúc này ít được thể hiện trong ngữ cảnh dữ liệu được thu thập.

3. Tiền xử lý dữ liệu

3.1. Kiểm tra dữ liệu trùng lặp

```
print(f"Có dữ liệu trùng lặp nào không : {mydata.duplicated().values.any()}\n")
```

```
Có dữ liệu trùng lặp nào không : False
```

Hình 5: Kiểm tra dữ liệu trùng lặp

Không có dữ liệu trùng lặp trong tập dữ liệu

3.2. Kiểm tra dữ liệu thiếu

```
valuesnull = mydata.isnull().sum()  
print(valuesnull)
```

```
Index    0  
Text     0  
Label    0  
dtype: int64
```

Hình 6: Kiểm tra dữ liệu thiếu

Tập dữ liệu không có dữ liệu thiếu

3.3. Xóa bỏ các đường dẫn URLs

Các đường dẫn thường không mang ý nghĩa nhiều về mặt cảm xúc. Vậy nên việc loại bỏ chúng sẽ giúp tập trung vào các nội dung quan trọng khác để huấn luyện mô hình.

Ví dụ: I am very happy to know you through the website <https://joingwithme.kh>, trong câu này <https://joingwithme.kh> không liên quan đến cảm xúc.

```
# bước 1: xóa URLs
mydata['Text'] = mydata['Text'].str.replace(r'http\S+', '', regex=True)
print(mydata.head())
```

	Index	Text	Label
0	0	i just feel really helpless and heavy hearted	4
1	1	ive enjoyed being able to slouch about relax a...	0
2	2	i gave up my internship with the dmrg and am f...	4
3	3	i dont know i feel so lost	0
4	4	i am a kindergarten teacher and i am thoroughl...	4

Hình 7: Xóa URLs

3.4. Xóa các ký hiệu đặc biệt và dấu câu

Các ký hiệu đặc biệt và dấu câu thường không mang ý nghĩa ngữ nghĩa quan trọng trong phân tích cảm xúc và có thể làm tăng kích thước không gian đặc trưng một cách không cần thiết. Việc xóa chúng giúp giảm thiểu nhiễu và tập trung vào các từ khóa quan trọng.

```
mydata['Text'] = mydata['Text'].str.replace(r'[\W\s]', '', regex=True)
print(mydata.head())
```

	Index	Text	Label
0	0	i just feel really helpless and heavy hearted	4
1	1	ive enjoyed being able to slouch about relax a...	0
2	2	i gave up my internship with the dmrg and am f...	4
3	3	i dont know i feel so lost	0
4	4	i am a kindergarten teacher and i am thoroughl...	4

Hình 8: Xóa ký tự đặc biệt và dấu câu

3.5. Loại bỏ các khoảng trắng dư thừa

Khoảng trắng dư thừa không mang ý nghĩa ngữ nghĩa và có thể ảnh hưởng đến quá trình trích xuất đặc trưng. Việc loại bỏ giúp chuẩn hóa văn bản và giảm

kích thước dữ liệu.

```
mydata['Text'] = mydata['Text'].str.replace(r'\s+', ' ', regex=True)
print(mydata.head())
```

Hình 9: Loại bỏ khoảng trắng dư thừa

3.6. Xóa các giá trị số trong văn bản

Các giá trị số thường không mang nhiều ý nghĩa về mặt cảm xúc, trừ khi chúng là một phần của một biểu thức cảm xúc (ví dụ: "10/10 would recommend"). Giúp loại bỏ nhiễu và tập trung vào nội dung văn bản.

```
[ ] mydata['Text'] = mydata['Text'].str.replace(r'\d+', '', regex=True)
mydata.head()
```

Hình 10: Xóa số

3.7. Biến đổi văn bản sang chữ thường

Chuyển đổi tất cả các ký tự thành chữ thường giúp giảm kích thước không gian đặc trưng và tránh việc coi các từ viết hoa và viết thường là khác nhau. Giúp đơn giản hóa văn bản và cải thiện hiệu quả của mô hình.

```
mydata['Text'] = mydata['Text'].str.lower()
print(mydata.head())
```

	Index	Text	Label
0	0	i just feel really helpless and heavy hearted	4
1	1	ive enjoyed being able to slouch about relax a...	0
2	2	i gave up my internship with the dmrg and am f...	4
3	3	i dont know i feel so lost	0
4	4	i am a kindergarten teacher and i am thoroughl...	4

Hình 11: Chuyển đổi chữ thường

3.8. Xóa các stop từ

Stop words là những từ phổ biến (ví dụ: "the", "a", "is") xuất hiện nhiều trong văn bản nhưng không mang nhiều ý nghĩa ngữ nghĩa. Giúp giảm kích thước không gian đặc trưng và tập trung vào các từ khóa quan trọng hơn.

Bộ dữ liệu stopwords chúng em tải về từ thư viện 'nltk'. Thư viện [NLTK](#) - Natural Language Toolkit là một trong những thư viện open-source xử lý ngôn ngữ tự nhiên. Thư viện cung cấp hơn 50 kho dữ liệu văn bản khác nhau (corpora) và nhiều chức năng để xử lý dữ liệu văn bản để phục vụ cho nhiều mục đích khác nhau.[2]

```
[13] #import các thư viện
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
#tải dữ liệu
nltk.download('punkt')
nltk.download('stopwords')
```

Hình 12: Import thư viện nltk



The screenshot shows a Jupyter Notebook cell with the following code:

```
stop = stopwords.words('english')
mydata["Text"] = mydata["Text"].apply(lambda x: ' '.join([word for word in x.split() if word not in (stop)]))
mydata.head()
```

Below the code, the first five rows of the DataFrame are displayed:

	Index	Text	Label
0	0	feel really helpless heavy hearted	4
1	1	ive enjoyed able slouch relax unwind frankly n...	0
2	2	gave internship dmrg feeling distraught	4
3	3	dont know feel lost	0
4	4	kindergarten teacher thoroughly weary job take...	4

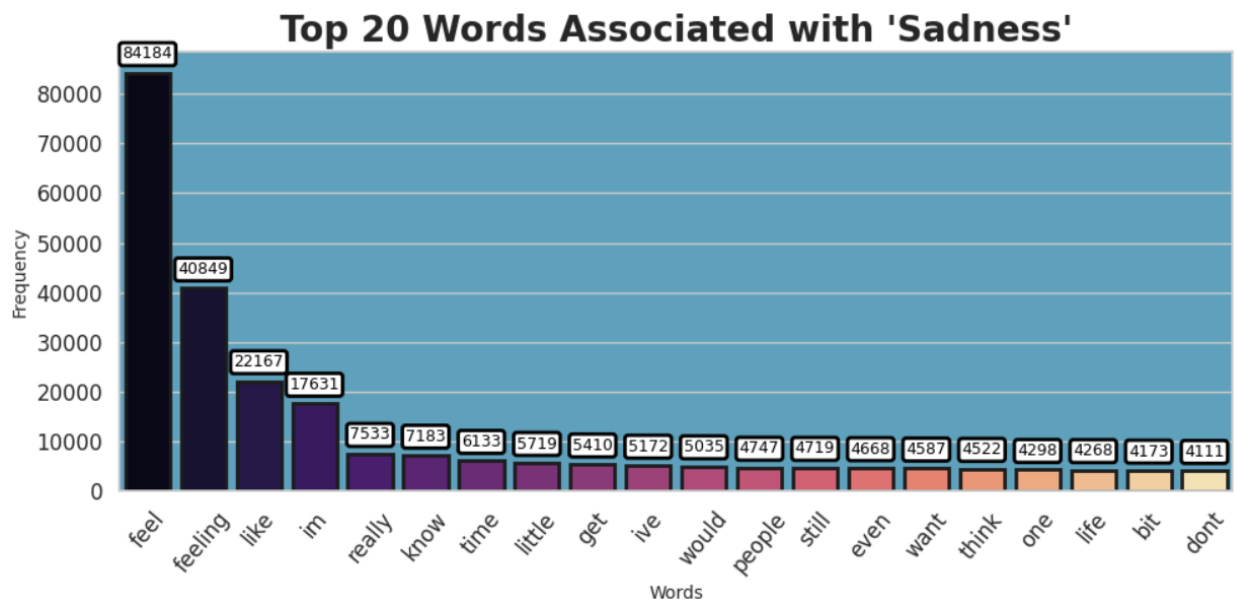
Hình 13: Xóa stop words

4. Một số từ thường gặp trong các cảm xúc

Sau khi tiền xử lý chúng ta có được một tập dữ liệu sạch hơn, tập trung vào những từ ngữ thực sự mang ý nghĩa về cảm xúc sẽ chất lượng hơn.

Nếu phân tích tần suất từ trên dữ liệu chưa qua xử lý, kết quả có thể bị sai lệch do sự xuất hiện của những từ không mong muốn. Ví dụ, stop words có thể xuất hiện với tần suất cao nhưng không mang nhiều thông tin về cảm xúc. Tiền xử lý giúp loại bỏ những từ này, từ đó đảm bảo kết quả phân tích từ thường gặp phản ánh đúng xu hướng cảm xúc trong văn bản. từ đó kết quả của dữ liệu sẽ đáng tin cậy hơn.

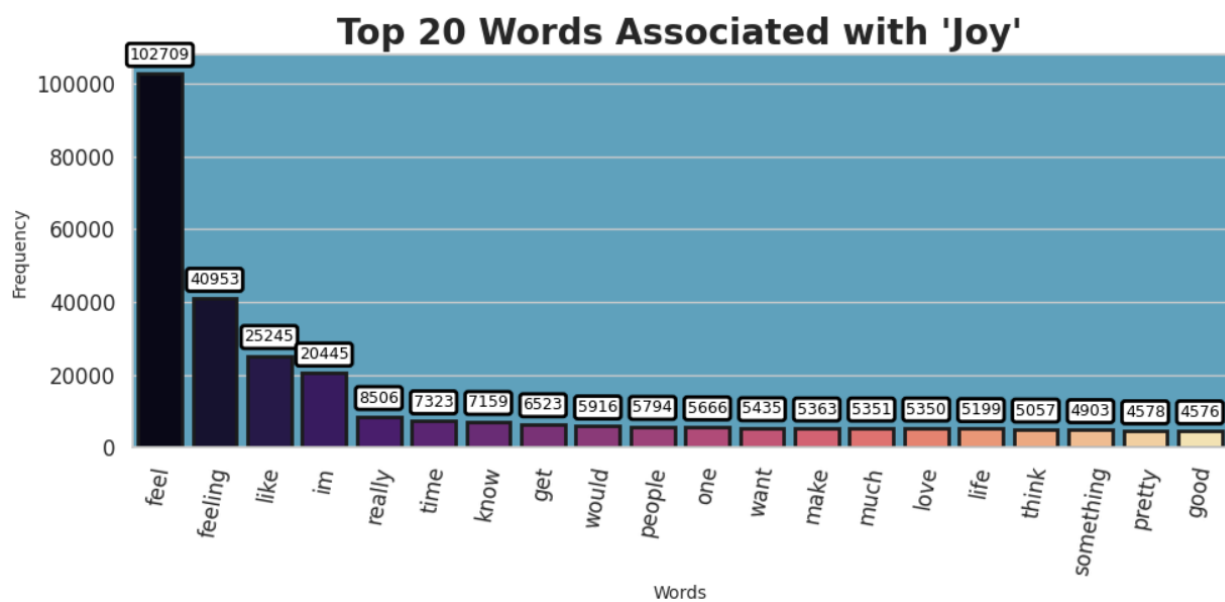
4.1. Một số từ thường gặp với 'sadness'



Hình 14: 20 từ thường gặp với 'sadness'

Các từ feel , feeling, like, im, really, know có tần suất xuất hiện nhiều nhất trong tập dữ liệu, nhưng vì đây là các từ biểu thị cho các cảm xúc chung nên chúng ta sẽ không đề cập tới, thay vào đó chúng ta có các từ như: don't, bit, life, think even, still.

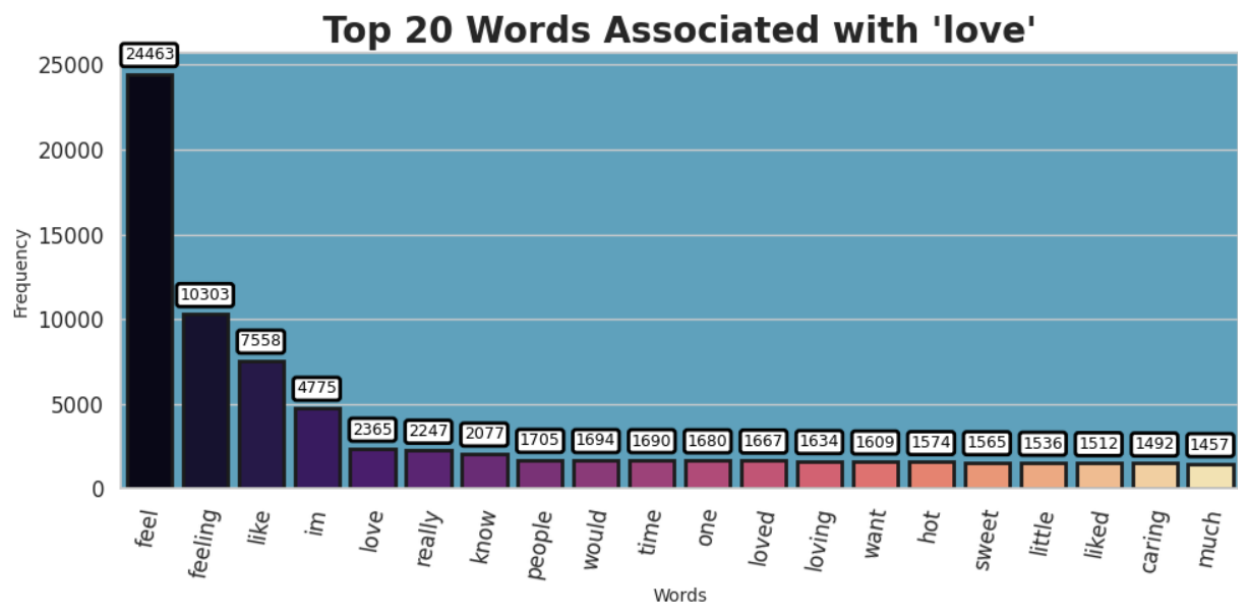
4.2. Một số từ thường gặp với ‘joy’



Hình 15: 20 từ thường gặp với 'joy'

Một số từ đặc trưng như: good, pretty, something, love, much...

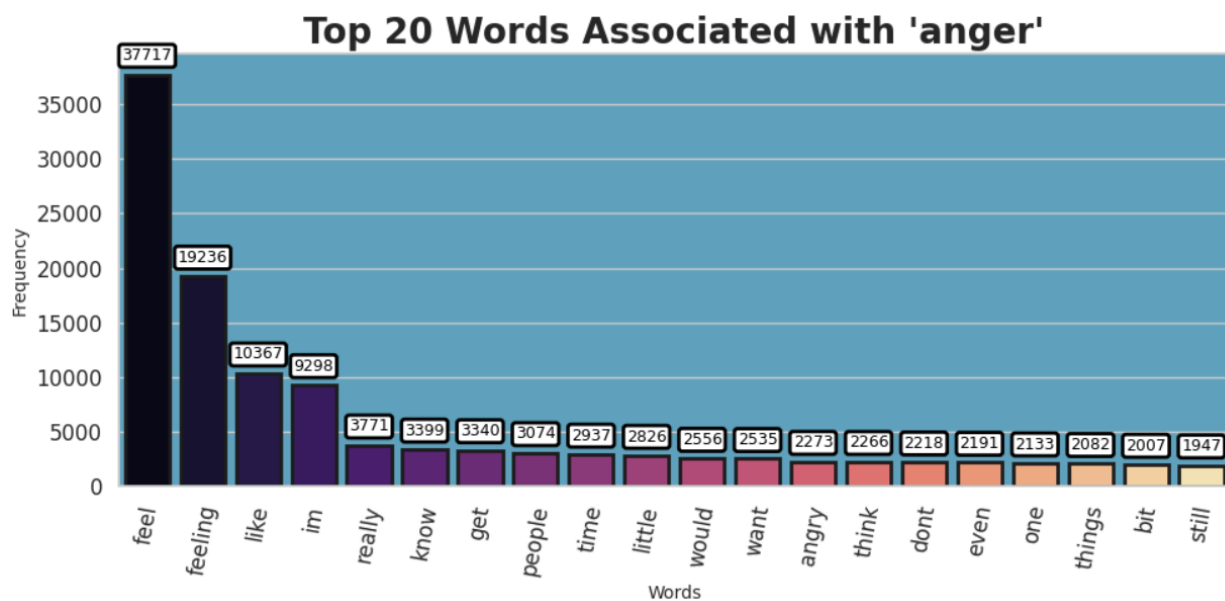
4.3. Một số từ thường gặp với “love”



Hình 16: 20 từ thường gặp với 'love'

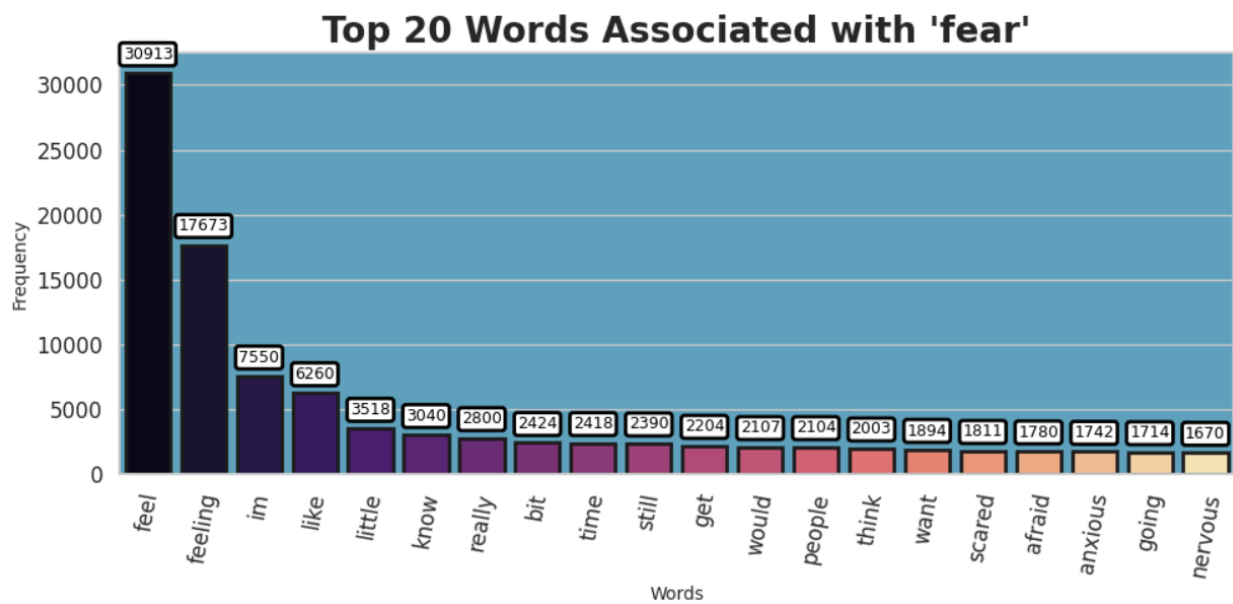
Một số từ thường gặp với 'love' như: caring, liked, little, sweet, loving, loved...

4.4. Một số từ thường gặp trong 'anger'



Một số từ thường gặp với 'anger' như: still, don't, even, angry...

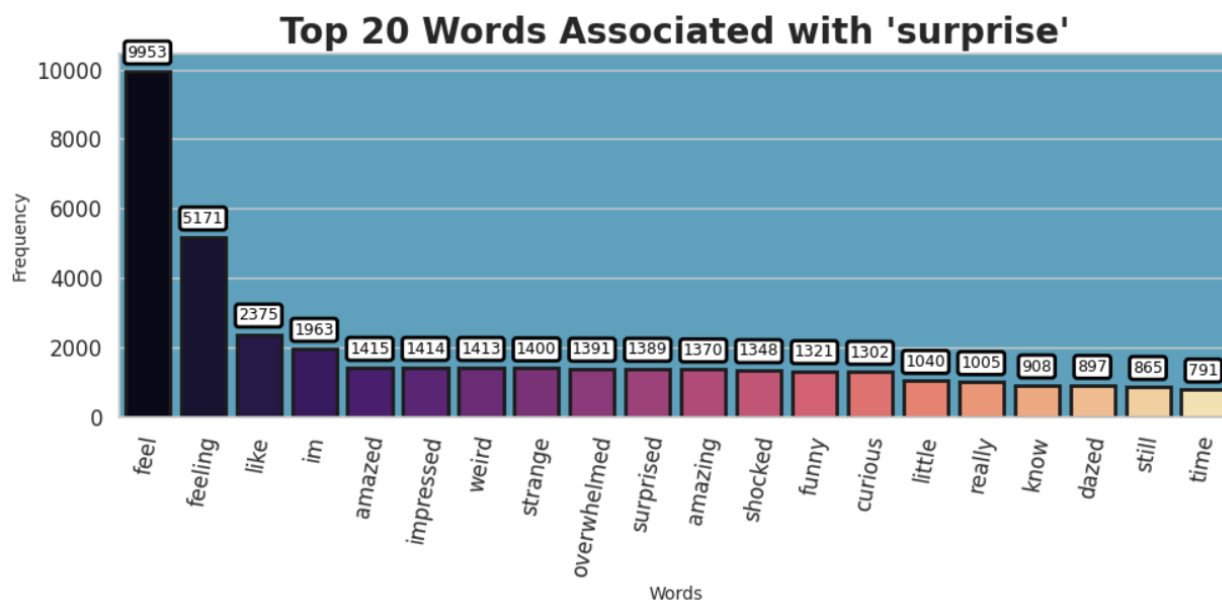
4.5. Một số từ thường gặp với ‘fear’



Hình 17: 20 từ thường gặp với 'fear'

Một số từ thường gặp với ‘fear’ như: nervous, anxious, afraid, scared,...

4.6. Một số từ thường gặp với 'surprise'



Hình 18: 20 từ thường gặp với 'surprise'

Một số từ thường gặp với 'surprise' như: amazed, impressed, weird, strange, surprised, amazing, shocked, ...

Phần 4 Huấn luyện mô hình

1. Chia dữ liệu

Với dữ liệu của có phân phối không đều giữa các nhãn mục tiêu, việc chia dữ liệu cần đảm bảo tính đại diện của các nhãn trong cả tập huấn luyện, tập kiểm tra.

```
[37] from sklearn.model_selection import train_test_split
      # chia dữ liệu cho training (80 %) và testing sets (20 %) theo từng nhãn
      X_train, X_test, y_train, y_test = train_test_split(mydata['Text'], mydata['Label'], test_size=0.2, stratify=mydata['Label'], random_state=42)
```

Hình 19: Chia dữ liệu tập train, test

Thay vì chia tỉ lệ theo cả tập dữ liệu chúng em sử dụng phương pháp **Stratified Sampling**, chia dữ liệu theo từng nhãn mục tiêu (**stratify=mydata['Label']**), với là 80% cho tập training và 20% cho tập testing.

```
[40] # prompt: kích thước của tập train và tập test sau khi chia dữ liệu
      print(f"Training set size: {X_train.shape} : {y_train.shape}")
      print(f"Testing set size: {X_test.shape} : {y_test.shape}")
```

```
⇒ Training set size: (333447,) : (333447,)
   Testing set size: (83362,) : (83362,)
```

Hình 20: Kích thước tập train, test

Sau khi chia dữ liệu, kích thước của tập training là (333447) dòng dữ liệu và tập testing là 83362 dòng dữ liệu.

2. Phương pháp đánh giá

Chúng em sử dụng chỉ số F1-score để đánh giá mô hình trong bài toán này vì những lý do sau:

- F1-score là trung bình điều hòa của Precision và Recall. Precision đo lường tỷ lệ dự đoán đúng trong số các dự đoán tích cực, trong khi Recall đo lường tỷ lệ dự đoán đúng trong số các trường hợp thực sự tích cực. F1-score cân bằng giữa hai số liệu này, giúp đánh giá hiệu suất tổng thể của mô hình một cách toàn diện hơn.

- Trong dữ liệu của chúng em, các cảm xúc, các lớp phân bố không đều joy và sadness xuất hiện nhiều hơn love, fear, surprise. Các cảm xúc có thể có tần suất xuất hiện khác nhau. Sử dụng F1-score sẽ giúp đảm bảo rằng mô hình được đánh giá công bằng trên tất cả các lớp.
- F1-score là một số liệu dễ hiểu và diễn giải, giúp dễ dàng so sánh hiệu suất của các mô hình khác nhau.

3. Mô hình 1 - Linear Support Vector Classifier (SVC)

```
def linear_SVC_train(): #tạo mô hình
    # chia dữ liệu cho training (80 %) và testing sets (20 %) theo từng nhãn
    X_train, X_test, y_train, y_test = train_test_split(mydata['Text'], mydata['Label'], test_size=0.2, stratify=mydata['Label'], random_state=42)

    # chuẩn hóa vector số học cho văn bản bằng TF-IDF
    vectorizer = TfidfVectorizer(max_features=1000) #chỉ sử dụng 1000 từ quan trọng nhất làm đặc trưng
    X_train_vectorized = vectorizer.fit_transform(X_train) # fit_transform(): áp dụng TF-IDF cho tập training
    X_test_vectorized = vectorizer.transform(X_test) # Vectorizer: sử dụng Vectorizer đã huấn luyện để biến đổi tập testing

    # Cài đặt các tham số mặc định cho mô hình
    model_params = {
        'C': 1.0,
        'loss': 'squared_hinge',
        'max_iter': 1000,
        'penalty': 'l2',
        'dual': True,
        'tol': 1e-4
    }
```

Hình 21: Mô hình Linear Support Vector Classifier (SVC)

Xây dựng mô hình SVC với các thông số mặc định ban đầu của mô hình chúng em được kết quả sau:

Classification Report:					
	precision	recall	f1-score	support	
0	0.93	0.90	0.92	24238	
1	0.84	0.93	0.88	28214	
2	0.81	0.74	0.77	6911	
3	0.88	0.82	0.85	11463	
4	0.85	0.78	0.82	9542	
5	0.78	0.74	0.76	2994	
accuracy			0.87	83362	
macro avg	0.85	0.82	0.83	83362	
weighted avg	0.87	0.87	0.87	83362	

Hình 22: Kết quả thực thi của SVC

Kết quả: mô hình đã đạt được điểm F1-score tổng thể là 0,83

Điểm F1-score thấp hơn so với kỳ vọng nên chúng em quyết định cải thiện hiệu suất thông qua các tham số

Phương pháp cải thiện

1. Vector hóa TF-IDF:

- Tăng số lượng đặc trưng trong vector TF-IDF từ **1000** lên **5000** để nắm bắt nhiều thông tin chi tiết hơn từ dữ liệu văn bản.

2. Điều chỉnh siêu tham số của mô hình:

- Tham số điều chuẩn (C): Giảm từ **1.0** xuống **0.1** để giảm overfitting và cải thiện khả năng tổng quát hóa của mô hình.
- Hàm mất mát: Thay đổi từ '**squared_hinge**' sang '**hinge**' để đạt hiệu suất tốt hơn.
- Số lần lặp tối đa: Tăng từ 1000 lên **5000** để cho phép mô hình hội tụ đến giải pháp tốt hơn.
- Chuẩn hóa: Giữ nguyên **L2** để thực hiện điều chuẩn.

- Dual: Giữ nguyên True để giải bài toán tối ưu hóa.

```
model_params = {
    'C': 0.1,
    'loss': 'hinge',
    'max_iter': 5000,
    'penalty': 'l2',
    'dual': True,
    'tol': 1e-4
}
```

Hình 23: Điều chỉnh các tham số của SVC

Kết quả:

Classification Report:				
	precision	recall	f1-score	support
0	0.93	0.95	0.94	24238
1	0.90	0.96	0.93	28214
2	0.90	0.70	0.79	6911
3	0.91	0.90	0.90	11463
4	0.86	0.86	0.86	9542
5	0.90	0.64	0.75	2994
accuracy			0.90	83362
macro avg	0.90	0.83	0.86	83362
weighted avg	0.90	0.90	0.90	83362

Hình 24: Kết quả sau khi điều chỉnh tham số của SVC

Kết Quả Điều Chỉnh Sau khi điều chỉnh lại các tham số điểm F1-sscore tổng thể của mô hình từ **0.82** đã tăng lên **0.86**

4. Mô hình 2 - Logistic Regression

Xây dựng mô hình với các tham số mặc định ban đầu của mô hình

```

from sklearn.linear_model import LogisticRegression
def logistic_regression_classifier():
    X_train, X_test, y_train, y_test = train_test_split(mydata['Text'], mydata['Label'], test_size=0.2, random_state=42)

    vectorizer = TfidfVectorizer(max_features=1000)
    X_train_vectorized = vectorizer.fit_transform(X_train)
    X_test_vectorized = vectorizer.transform(X_test)
    # Các tham số mặc định của hàm mô hình
    model_params = {
        'C': 0.1,
        'penalty': 'l2',
        'max_iter': 5000,
        'solver': 'saga',
        'random_state': 1
    }

    model = LogisticRegression(**model_params)
    model.fit(X_train_vectorized, y_train)

    return model, vectorizer, X_test_vectorized, y_test

model_lr, trained_vectorizer, X_test, y_test = logistic_regression_classifier()
evaluate_model(model_lr, X_test, y_test)

```

Hình 25: Mô hình Logistic regression

Kết quả:

↔ Classification Report:					
	precision	recall	f1-score	support	
0	0.91	0.91	0.91	24201	
1	0.83	0.94	0.88	28164	
2	0.84	0.70	0.76	6929	
3	0.90	0.79	0.84	11441	
4	0.85	0.79	0.82	9594	
5	0.83	0.67	0.74	3033	
accuracy			0.86	83362	
macro avg	0.86	0.80	0.83	83362	
weighted avg	0.87	0.86	0.86	83362	

Hình 26: Kết quả thực thi của Logistic regression

Hiệu suất của mô hình: Điểm F1-Score của mô hình thứ 2 đạt được là **0.83**, một điểm số trung bình nên chúng em quyết định cải thiện mô hình xem có tốt hơn không.

Phương pháp cải thiện :

1. Vector hóa TF-IDF:

- Tăng số lượng đặc trưng trong vector TF-IDF từ 1000 lên **5000**.

2. Điều chỉnh siêu tham số của mô hình:

- Hình phạt điều chuẩn: Thay đổi từ **L2** sang **L1** để tạo sự thưa thớt (sparsity) và có khả năng cải thiện việc lựa chọn đặc trưng bằng cách giảm ảnh hưởng của các đặc trưng kém quan trọng.
- Số lần lặp tối đa: Tăng từ **5000** lên **10000** để cho phép thời gian hội tụ lâu hơn, giúp tối ưu hóa tốt hơn các tham số của mô hình.

```
vectorizer = TfidfVectorizer(max_features=5000)
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)

model_params = {
    'C': 0.1,
    'penalty': 'l1',
    'max_iter': 10000,
    'solver': 'saga',
    'random_state': 1
}
model = LogisticRegression(**model_params)
model.fit(X_train_vectorized, y_train)
```

Hình 27: Điều chỉnh tham số của Logistic regression

Classification Report:					
	precision	recall	f1-score	support	
0	0.94	0.95	0.94	24201	
1	0.91	0.94	0.93	28164	
2	0.84	0.78	0.81	6929	
3	0.91	0.91	0.91	11441	
4	0.87	0.86	0.86	9594	
5	0.83	0.72	0.77	3033	
accuracy			0.91	83362	
macro avg	0.88	0.86	0.87	83362	
weighted avg	0.91	0.91	0.91	83362	

Hình 28: Kết quả sau khi điều chỉnh tham số của Logistic regression

Kết quả tinh chỉnh

- Những điều chỉnh này đã tác động đáng kể đến F1-score, minh chứng sự cải thiện vượt bậc trong hiệu suất.
- F1-score ban đầu khoảng **0.83** đã được nâng lên **0.87**

5. Mô hình Naïve Bayes Classifier

Xây dựng mô hình với các tham số ban đầu

```
from sklearn.naive_bayes import MultinomialNB
def naive_bayes_classifier():
    X_train, X_test, y_train, y_test = train_test_split(mydata['Text'], mydata['Label'], test_size=0.2, random_state=42)

    vectorizer = TfidfVectorizer(max_features=5000)
    X_train_vectorized = vectorizer.fit_transform(X_train)
    X_test_vectorized = vectorizer.transform(X_test)

    model = MultinomialNB()
    model.fit(X_train_vectorized, y_train)

    return model, vectorizer, X_test_vectorized, y_test

model_mnb, trained_vectorizer, X_test, y_test = naive_bayes_classifier()
evaluate_model(model_mnb, X_test, y_test)
```

Hình 29: Mô hình Naive Bayes Classifier

Kết quả

Classification Report:					
	precision	recall	f1-score	support	
0	0.86	0.95	0.90	24201	
1	0.80	0.97	0.88	28164	
2	0.95	0.49	0.64	6929	
3	0.94	0.79	0.86	11441	
4	0.89	0.73	0.80	9594	
5	0.97	0.31	0.47	3033	
accuracy			0.85	83362	
macro avg	0.90	0.71	0.76	83362	
weighted avg	0.86	0.85	0.84	83362	

Hình 30: Kết quả thực thi của Naive Bayes Classifier

Hiệu suất mô hình: Điểm số F1_score của mô VB khá là thấp chỉ được **0.72**.

Điều chỉnh Mô hình 3

Trong phiên bản cải thiện của mô hình phân loại Naive Bayes, các siêu tham số đã được điều chỉnh nhằm cải thiện hiệu suất.

Tinh chỉnh Siêu tham số của mô hình:

- Force Alpha: Tắt tính năng này (force_alpha được đặt thành False) để ngăn việc làm mịn tự động.
- Giá trị Alpha: Đặt bằng 0, không sử dụng làm mịn Laplace, nhằm tránh hiện tượng overfitting có thể xảy ra với việc làm mịn tự động.
- Fit Prior: Đặt thành False để tắt tính năng khớp xác suất tiên nghiệm của các lớp, cung cấp sự linh hoạt hơn trong việc xử lý phân phối lớp không cân bằng.

```
def naive_bayes_classifier():
    X_train, X_test, y_train, y_test = train_test_split(mydata['Text'], mydata['Label'], test_size=0.2, random_state=42)

    vectorizer = TfidfVectorizer(max_features=5000)
    X_train_vectorized = vectorizer.fit_transform(X_train)
    X_test_vectorized = vectorizer.transform(X_test)

    model = MultinomialNB(force_alpha=False, alpha=0, fit_prior=False)
    model.fit(X_train_vectorized, y_train)

    return model, vectorizer, X_test_vectorized, y_test

model_mnb, trained_vectorizer, X_test, y_test = naive_bayes_classifier()
evaluate_model(model_mnb, X_test, y_test)
```

Hình 31: Điều chỉnh mô hình Naive Bayes Classifier

Classification Report:					
	precision	recall	f1-score	support	
0	0.95	0.89	0.92	24201	
1	0.95	0.85	0.90	28164	
2	0.67	0.91	0.77	6929	
3	0.88	0.90	0.89	11441	
4	0.83	0.85	0.84	9594	
5	0.60	0.87	0.71	3033	
accuracy			0.88	83362	
macro avg	0.81	0.88	0.84	83362	
weighted avg	0.89	0.88	0.88	83362	

Hình 32: Kết quả sau khi điều chỉnh của Naive Bayes Classifier

Kết quả điều chỉnh

Những thay đổi trên đã mang lại những cải thiện đáng kể cho mô hình và điểm F1-score từ **0.72** đến **0.83**

6. Mô hình randomforest


Xây dựng mô hình với các thông số ban đầu:

```
X_train, X_test, y_train, y_test = train_test_split(mydata['Text'], mydata['Label'], test_size=0.2,
vectorizer = TfidfVectorizer(max_features=1000)
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)

model = RandomForestClassifier(n_estimators=10, random_state=42)
model.fit(X_train_vectorized, y_train)
```

Hình 33: Mô hình Random Forest

Kết quả:

 Classification Report:

	precision	recall	f1-score	support
0	0.89	0.88	0.89	24238
1	0.84	0.86	0.85	28214
2	0.66	0.65	0.66	6911
3	0.81	0.81	0.81	11463
4	0.76	0.74	0.75	9542
5	0.63	0.66	0.64	2994
accuracy			0.82	83362
macro avg	0.77	0.77	0.77	83362
weighted avg	0.82	0.82	0.82	83362

Hình 34: Kết quả thực thi của Random Forest

Hiệu suất mô hình: Điểm số F1_score của mô RandomForest cũng đạt 0.77

Phương pháp cải thiện:

1. Vector hóa TF-IDF: Tăng số lượng đặc trưng trong vector TF-IDF từ **1000** lên **5000** để nắm bắt nhiều thông tin chi tiết hơn từ dữ liệu văn bản.

2. Điều chỉnh siêu tham số của mô hình:

Tăng số lượng mẫu tối thiểu cần thiết để tạo một nút lá trong cây quyết định giá trị cao hơn sẽ ngăn chặn overfitting, **min_samples_leaf: 4**

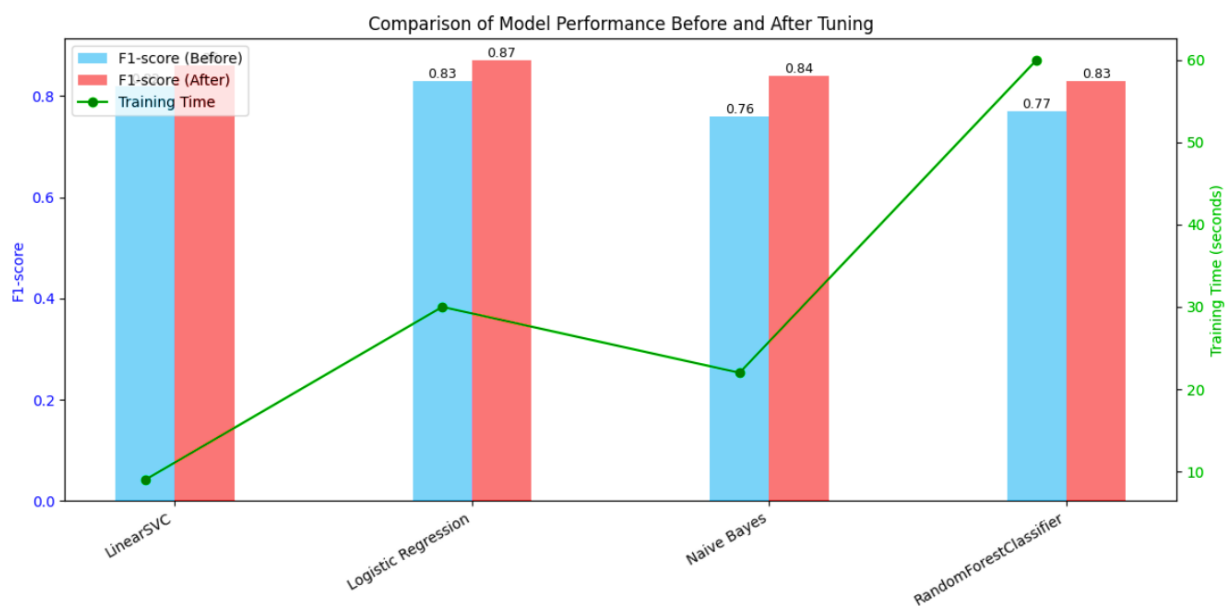


Classification Report:

	precision	recall	f1-score	support
0	0.92	0.93	0.92	24238
1	0.92	0.90	0.91	28214
2	0.78	0.71	0.74	6911
3	0.86	0.90	0.88	11463
4	0.80	0.83	0.82	9542
5	0.71	0.68	0.69	2994
accuracy			0.88	83362
macro avg	0.83	0.82	0.83	83362
weighted avg	0.88	0.88	0.88	83362

Hình 35: Kết quả sau khi điều chỉnh của Random Forest

7. So sánh các mô hình:



Hình 36: Điểm hiệu suất F1-score của các mô hình

Qua các mô hình có thể thấy hiệu suất đã được cải thiện vượt trội sau khi mô hình được tinh chỉnh. Đặc biệt mô hình Logistic Regression cho thấy sự cải thiện

vượt trội sau khi được tinh chỉnh, trở thành mô hình có hiệu suất tốt nhất trong bài toán của chúng ta. đứng thứ 2 là mô hình LinearSVC và sau đó là mô hình Naïve Bayes và Random Forest.

Qua đó cũng cho thấy rằng việc lựa chọn và điều chỉnh siêu tham số phù hợp có thể giúp cải thiện đáng kể hiệu suất của một mô hình.

Phần 5: Tổng kết

1. Tóm tắt

Hiệu suất mô hình:

Nghiên cứu này đã khám phá và đánh giá hiệu suất của bốn mô hình học máy phổ biến trong việc phân loại cảm xúc từ văn bản tiếng Anh: LinearSVC, Logistic Regression, Naïve, Random Forest. Kết quả cho thấy mô hình Logistic Regression, sau khi được tinh chỉnh siêu tham số và áp dụng kỹ thuật vector hóa TF-IDF, đã đạt được hiệu suất tốt nhất với điểm F1 ấn tượng là 0.87. Điểm số này vượt trội hơn so với các mô hình khác. Điều này chứng tỏ rằng Logistic Regression, khi được điều chỉnh phù hợp, có khả năng phân loại cảm xúc từ văn bản tiếng Anh một cách hiệu quả và chính xác hơn.

Tinh chỉnh siêu tham số:

Việc tinh chỉnh các siêu tham số đóng vai trò quan trọng trong việc cải thiện hiệu suất của các mô hình. Ví dụ, đối với mô hình Logistic Regression, việc thay đổi hình phạt điều chuẩn từ L2 sang L1, tăng số lần lặp tối đa và điều chỉnh tham số điều chuẩn C đã mang lại sự cải thiện đáng kể về điểm F1. Tương tự, đối với mô hình Naive Bayes, việc tắt tính năng làm mịn tự động và khớp xác suất tiên nghiệm của các lớp cũng giúp tăng hiệu suất. Điều này cho thấy tầm quan trọng của việc lựa chọn và tinh chỉnh các siêu tham số phù hợp để tối ưu hóa hiệu suất của mô hình.

Kỹ thuật xử lý đặc trưng:

Việc sử dụng vector hóa TF-IDF, với số lượng đặc trưng được tăng từ 1000 lên 5000, đã đóng góp đáng kể vào việc cải thiện hiệu suất của các mô hình. Việc tăng số lượng đặc trưng cho phép mô hình nắm bắt được nhiều thông tin chi tiết hơn từ văn bản, từ đó mô hình cải thiện khả năng phân loại cảm xúc hơn.

Khả năng diễn giải mô hình:

Một ưu điểm của mô hình Logistic Regression là khả năng diễn giải. Mô hình này cho phép chúng ta hiểu rõ hơn các yếu tố ảnh hưởng đến dự đoán của nó, ví dụ như tầm quan trọng của các từ hoặc cụm từ trong việc xác định cảm xúc. Khả năng diễn giải này rất hữu ích trong các ứng dụng thực tế, nơi tính minh bạch và khả năng giải thích là rất quan trọng.

2. Hạn chế và hướng phát triển

Mặc dù đã đạt được kết quả khả quan, nghiên cứu này vẫn còn một số hạn chế. Đầu tiên, tập dữ liệu sử dụng trong nghiên cứu chưa đủ để đại diện cho tất cả các trường hợp cảm xúc trong thực tế. Thứ hai, dữ liệu một cảm xúc quá ít (surprise, anger) dẫn đến dữ liệu không đồng đều, mô hình có thể chưa xử lý tốt các cảm xúc có tần suất thấp, như "Ngạc nhiên" và "Sợ hãi".

Trong tương lai, các nghiên cứu tiếp theo có thể tập trung khắc phục những điểm yếu này bằng cách tăng cường dữ liệu đầy đủ giữa các trường hợp và đa dạng hơn, bao gồm nhiều loại văn bản và cảm xúc khác nhau, hay nghiên cứu các phương pháp xử lý đặc trưng và xây dựng mô hình hóa phức tạp hơn, như mạng nơ-ron, cũng có thể giúp cải thiện hiệu suất phân loại cảm xúc. Cuối cùng, việc áp dụng mô hình vào các ứng dụng thực tế, như phân tích ý kiến khách hàng, chatbot và hỗ trợ tâm lý, cũng là một hướng phát triển tiềm năng.

3.Kết luận tổng quát

Trong quá trình thực hiện đề tài, nhóm em đã trải qua các giai đoạn từ xây dựng mô hình, đánh giá hiệu suất của mô hình. Qua đó, nhóm đã có thêm nhiều kinh nghiệm quý báu về quy trình xây dựng mô hình học máy và cũng như các kỹ thuật tinh chỉnh mô hình. Trong đề tài nghiên cứu phân loại cảm xúc đã đạt được những kết quả khả quan, đặc biệt với mô hình hồi quy logistic sau khi tinh chỉnh đã đạt được số điểm kh F1-score là 0.87, cũng khẳng định hiệu quả của các phương pháp đã áp dụng. Mặc dù vẫn còn một số hạn chế, cần nghiên cứu sâu hơn để nâng cao độ chính xác và khả năng tổng quát của mô hình, nhưng đây cũng là cơ sở để chúng em có thể phát triển hệ thống tiên tiến hơn trong tương lai, với tiềm năng ứng dụng rộng rãi trong nhiều lĩnh vực.

Tài liệu tham khảo: