ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÁO CÁO BÀI TẬP LỚN
NHẬP MÔN HỌC MÁY VÀ KHAI PHÁ DỮ LIỆU

ĐỀ TÀI: XÁC ĐỊNH TÍNH CHÍNH XÁC CỦA CÁC REVIEWS SẢN PHẨM TRÊN TRANG THƯƠNG MẠI ĐIỆN TỬ

GVHD :	TS. Nguyễn Thị Kim Anh
Nhóm:	02
Danh sách sinh viên:	Mạc Văn An - 20210007
	Lê Quang Chiến - 20214999
	Bùi Thị Hương Trà - 20215150
	Võ Sơn Long - 20215082
	Thái Tuấn Nam - 20215100

Ngày 13 tháng 12 năm 2024

Lời nói đầu

Trong thời đại bùng nổ của công nghệ thông tin và trí tuệ nhân tạo, học máy (Machine Learning) và khai phá dữ liệu (Data Mining) đã trở thành những lĩnh vực nền tảng quan trọng, góp phần không nhỏ trong việc giải quyết các bài toán thực tiễn trong nhiều lĩnh vực như kinh tế, y tế, giáo dục, và công nghiệp. Việc hiểu và áp dụng các phương pháp, mô hình học máy không chỉ giúp khai thác hiệu quả nguồn dữ liệu khổng lồ, mà còn tạo tiền đề để phát triển những giải pháp thông minh và hiệu quả.

Báo cáo này được thực hiện trong khuôn khổ môn học Nhập Môn Học Máy và Khai Phá Dữ Liệu, nhằm cung cấp một cái nhìn tổng quan về quá trình phân tích dữ liệu và xây dựng mô hình giúp "Xác định tính chính xác của các reviews sản phẩm trên trang thương mại điện tử". Trong quá trình thực hiện, nhóm đã trải qua các bước từ thu thập dữ liệu, tiền xử lý, phân tích đặc trưng, đến việc áp dụng các mô hình học máy để giải quyết bài toán cụ thể. Báo cáo không chỉ trình bày kết quả đạt được, mà còn nhấn mạnh những khó khăn, bài học kinh nghiệm, và những kiến thức hữu ích thu được.

Nhóm thực hiện xin chân thành cảm ơn sự hướng dẫn tận tình của giảng viên TS. Nguyễn Thị Kim Anh và sự hỗ trợ từ các bạn trong lớp. Những góp ý quý báu đã giúp nhóm hoàn thiện tốt hơn trong quá trình thực hiện và báo cáo dự án này.

Chúng em hy vọng rằng nội dung của báo cáo không chỉ giúp bản thân các thành viên trong nhóm nắm bắt rõ hơn các kiến thức về học máy và khai phá dữ liệu, mà còn mang lại giá trị tham khảo hữu ích cho các bạn đọc quan tâm.

Phân công nhiệm vụ

Họ và tên	**MSSV**	**Vai trò**	**Nhiệm vụ**	**Đánh giá hoàn thành**
Mạc Văn An	20210007	Nhóm trưởng	Triển khai nhiệm vụ và quản lý tiến độ nhóm, Tìm hiểu và xây dựng model, Viết báo cáo.	120%
Lê Quang Chiến	20214999	Thành viên	Tìm hiểu và xây dựng model, Viết báo cáo.	110%
Võ Sơn Long	20215082	Thành viên	Crawl data và xây dựng dataset	90%
Thái Tuấn Nam	20215100	Thành viên	Crawl data và xây dựng dataset	90%
Bùi Thị Hương Trà	20215150	Thành viên	Tiền xử lý dữ liệu, Slide thuyết trình	90%

Mục lục

1 Tổng quan đề tài

1.1 Bối cảnh

Thương mại điện tử ngày càng phát triển, trở thành nơi mua sắm tiện lợi cho hàng triệu người dùng trên toàn thế giới. Một yếu tố quan trọng ảnh hưởng đến quyết định mua hàng của người tiêu dùng là các đánh giá (reviews) từ những người đã sử dụng sản phẩm. Các đánh giá này bao gồm hai phần chính: nội dung đánh giá (comment) và điểm số đánh giá (star).
Tuy nhiên, trong thực tế, không phải lúc nào nội dung đánh giá và điểm số đánh giá cũng đồng nhất và phản ánh đúng thực tế. Ví dụ:

- Có những đánh giá với nội dung tiêu cực, chê bai sản phẩm, nhưng lại cho số sao cao bất thường (4-5 sao).
- Ngược lại, có những đánh giá tích cực, ca ngợi sản phẩm, nhưng lại chỉ cho số sao thấp (1-2 sao).

Những mâu thuẫn này có thể xuất phát từ nhiều nguyên nhân: hiểu sai quy tắc đánh giá, lỗi chủ quan của người dùng, hoặc thậm chí là hành vi cố ý thao túng để gây hiểu lầm. Điều này không chỉ gây khó khăn cho người tiêu dùng trong việc đưa ra quyết định mua hàng, mà còn làm giảm độ tin cậy của hệ thống đánh giá sản phẩm trên các nền tảng thương mại điện tử.

1.2 Mục tiêu

Vận dụng các kiến thức trong học phần "Nhập môn Học máy và Khai phá dữ liệu" vào thực tiễn. Cụ thể, xác định tính chính xác và nhất quán giữa nội dung và số sao trong các đánh giá sản phẩm, thông qua các bước sau:

- Phân tích dữ liệu:
  - Xác định các đặc trưng trong nội dung văn bản đánh giá (comment)
  - Phát hiện các trường hợp bất thường giữa nội dung đánh giá và điểm số đánh giá.
- Xây dựng mô hình dự đoán:
  - Ứng dụng các kỹ thuật học máy để phân loại các đánh giá dựa trên tính nhất quán giữa nội dung.
  - So sánh số sao được sinh ra từ mô hình với số sao thực tế để đưa ra kết luận về tính chính xác của review sản phẩm.
- Đánh giá hiệu quả:
  - Sử dụng các chỉ số như độ chính xác (accuracy), độ nhạy (recall), F1-score để đo lường hiệu quả mô hình.
  - Phân tích những trường hợp mô hình dự đoán sai để cải thiện thuật toán.
- Đề xuất giải pháp: Có thể sử dụng làm tiền để để xây dưng các công cụ cung cấp các gợi ý cải thiện hệ thống đánh giá trên các nền tảng thương mại điện tử, chẳng hạn như kiểm tra tự động hoặc cảnh báo khi có sự mâu thuẫn giữa nội dung và số sao.

1.3 Ý nghĩa thực tiễn

Việc phát hiện và xử lý các đánh giá không phù hợp mang lại nhiều lợi ích cho nhiều bên liên quan:

- Người tiêu dùng: Giảm thiểu sự nhầm lẫn, giúp đưa ra quyết định mua hàng chính xác hơn.

- Người bán hàng: Tăng cường uy tín và bảo vệ thương hiệu khỏi những đánh giá sai lệch.
- Nền tảng thương mại điện tử: Tăng cường độ tin cậy và sự hài lòng của người dùng, cải thiện trải nghiệm mua sắm.

2 Bộ dữ liệu

2.1 Mô tả bộ dữ liệu

Dữ liệu được sử dụng trong dự án này là một tập hợp các đánh giá và bình luận (bằng tiếng Anh) của khách hàng về các sản phẩm trên Amazon. Dataset bao gồm các thông tin như điểm số đánh giá (stars) mà khách hàng dành cho sản phẩm, cùng với những bình luận (comments) mô tả trải nghiệm thực tế khi sử dụng sản phẩm.

Các biến trong bộ dữ liệu
comment:

- Mô tả: Nội dung bình luận của khách hàng về sản phẩm.
- Loại dữ liệu: Chuỗi văn bản (string).
- Phạm vi giá trị: Nội dung của từng bình luận, có thể có nội dung rất ngắn hoặc rất dài, phụ thuộc vào người mua.

star:

- Mô tả: Số sao đánh giá của sản phẩm, giá trị từ 1 (rất tệ) đến 5 (rất tốt).
- Loại dữ liệu: Số nguyên (integer).
- Phạm vi giá trị: 1, 2, 3, 4, 5.

Bộ dữ liệu thu thập được chứa 500 bình luận cho mỗi mức đánh giá sao, đảm bảo sự cân bằng và đa dạng trong các phản hồi từ khách hàng đối với các sản phẩm.

2.2 Quá trình thu thập

Dữ liệu cho dự án này được thu thập từ trang thương mại điện tử Amazon, cụ thể là các đánh giá của khách hàng về sản phẩm.

2.2.1 Crawl dữ liệu từ Amazon

Việc thu thập dữ liệu được thực hiện thông qua việc sử dụng một API của dịch vụ bên thứ ba để lấy thông tin từ các trang sản phẩm trên Amazon. Quá trình này bao gồm:

- Gửi yêu cầu đến API để lấy chi tiết sản phẩm, trong đó đặc biệt chú trọng đến các đánh giá từ người mua.
- Trích xuất các thông tin như nội dung bình luận (`reviewText`) và xếp hạng (`reviewRating`) từ phản hồi của API.
- Sau khi thu thập, dữ liệu được lưu trữ dưới dạng file JSON, phục vụ cho các bước xử lý tiếp theo.

2.2.2 Lưu trữ và cập nhật dữ liệu

Dữ liệu thu thập được từ API sau đó được lưu vào một file JSON trên máy (và được lưu trữ trên cả GitHub repository). Trong trường hợp đã có dữ liệu trước đó, quy trình sẽ kiểm tra và cập nhật file hiện có bằng cách hợp nhất các đánh giá mới vào tập dữ liệu hiện có. Điều này đảm bảo rằng toàn bộ dữ liệu luôn được lưu trữ tập trung và dễ dàng truy xuất khi cần.

Việc cập nhật dữ liệu được thực hiện định kỳ khi có các đánh giá mới để đảm bảo tính toàn vẹn và đầy đủ của tập dữ liệu.

Tổng số lượng record thu thập được là **10260** bản.

2.2.3 Xử lý và lọc dữ liệu

Sau khi thu thập, dữ liệu cần được xử lý để loại bỏ những phần không cần thiết và chỉ giữ lại các thông tin có giá trị cho quá trình phân tích. Đầu tiên, tập trung vào việc lọc các đánh giá dựa trên yếu tố:

Ngôn ngữ: Vì mục tiêu của dự án là phân tích các đánh giá bằng tiếng Anh, nên chỉ những đánh giá có nội dung bằng tiếng Anh sẽ được giữ lại. Các đánh giá bằng ngôn ngữ khác sẽ bị loại bỏ để đảm bảo tính đồng nhất của dữ liệu.

Star	Count
1	1553
2	509
3	1065
4	1414
5	3999

Bảng 2: Số lượng các comment được đánh sao sau khi lọc ngôn ngữ

2.3 Tiền xử lý dữ liệu

2.3.1 Làm sạch và tối ưu bộ dữ liệu

Trong quá trình chuẩn bị dữ liệu cho mô hình phân tích, việc làm sạch dữ liệu là bước quan trọng để đảm bảo chất lượng và tính chính xác của các phân tích sau này. Các bước làm sạch dữ liệu bao gồm:

- **Loại bỏ dữ liệu thiếu:** Các bản ghi bị thiếu dữ liệu trong các trường quan trọng (như điểm số đánh giá hoặc bình luận) sẽ được xác định và loại bỏ khỏi bộ dữ liệu để tránh ảnh hưởng đến chất lượng của mô hình.
- **Loại bỏ dữ liệu không liên quan:** Những bình luận không chứa thông tin hữu ích hoặc không liên quan đến sản phẩm sẽ được loại bỏ. Điều này có thể bao gồm các bình luận quảng cáo hoặc không chứa bất kỳ đánh giá nào về sản phẩm.
- **Loại bỏ các bình luận trùng lặp:** Các bình luận trùng lặp sẽ được xác định và loại bỏ để tránh việc phân tích bị sai lệch.

Sau khi thực hiện các bước làm sạch, chọn ra mỗi mức đánh giá sao 500 comment để lưu trữ. Bộ dữ liệu sẽ trở nên chính xác, cân bằng và sẵn sàng để được phân tích sâu hơn.

2.3.2 Chuyển đổi định dạng và lưu trữ dữ liệu

Sau khi xử lý, dữ liệu được chuyển đổi sang định dạng CSV để dễ dàng phân tích và xử lý trong các công cụ Machine Learning. Tập dữ liệu CSV này bao gồm hai cột chính: nội dung đánh giá (comment) và xếp hạng sản phẩm (star). File CSV sau đó được lưu trữ trên máy và đưa lên Kaggle, sẵn sàng cho các bước tiếp theo trong quá trình phân tích và xây dựng mô hình.

Kết quả của quá trình thu thập và xử lý này là một tập dữ liệu đánh giá sản phẩm hợp lệ, được chuẩn hóa và có chất lượng, từ đó hỗ trợ các bước tiếp theo trong việc xây dựng mô hình Machine Learning để xác định tính chính xác của các đánh giá sản phẩm.

2.3.3 Chuyển đổi dữ liệu

Trong bài toán phân loại đánh giá sản phẩm, dữ liệu văn bản cần được chuyển đổi từ dạng thô (text) sang dạng số để mô hình có thể xử lý. Các bước tiền xử lý được thực hiện như sau (Dưới đây chỉ là sơ lược về phương pháp sử dụng trong quá trình tiền xử lý dữ liệu, cụ thể về các thư viện sẽ được mô tả chi tiết trong phần 3":

- **Xử lý văn bản thô:**
  - **Loại bỏ stopwords:**
    - **Thư viện sử dụng:** `stopwords from scikit-learn`
    - Stopwords là các từ thông dụng (như "the", "is", "and" trong tiếng Anh) không đóng góp nhiều vào ý nghĩa của câu. Việc loại bỏ chúng giúp giảm nhiễu và tập trung vào các từ quan trọng hơn.
    - **Ví dụ:**
      - Văn bản ban đầu: `"This product is very good and highly recommended."`
      - Sau khi loại bỏ stopwords: `"product good highly recommended"`
  - **Tách từ (Tokenization):**
    - **Thư viện sử dụng:** `nltk.tokenize`
    - Chia văn bản thành danh sách các từ (tokens).
    - **Ví dụ:**
      - Văn bản: `"product good highly recommended"`
      - Sau khi tokenization: `['product', 'good', 'highly', 'recommended']`
- **Biểu diễn dữ liệu văn bản bằng TF-IDF:**
  - **Thư viện sử dụng:** `TfidfVectorizer from scikit-learn`
  - TF-IDF (*Term Frequency-Inverse Document Frequency*) là kỹ thuật chuyển văn bản thành các vector số dựa trên:
    - **Term Frequency (TF):** Số lần một từ xuất hiện trong một tài liệu.
    - **Inverse Document Frequency (IDF):** Tần suất nghịch đảo của tài liệu chứa từ đó, giúp giảm độ quan trọng của các từ xuất hiện phổ biến trong nhiều tài liệu.
  - **Ví dụ:** Với các texts sau:
    - Text 1: `"product is good"`
    - Text 2: `"good product"`

    TF-IDF sẽ tạo ra một ma trận như sau:

	product	is	good
Text 1	0.5	0.7	0.5
Text 2	0.5	0.0	0.5

- **Giảm chiều dữ liệu bằng PCA:**
  - **Thư viện sử dụng:** `PCA from scikit-learn`
  - PCA (*Principal Component Analysis*) là kỹ thuật giảm chiều dữ liệu:
    - **Lý do cần giảm chiều:** TF-IDF có thể tạo ra các vector rất lớn (nhiều chiều), dẫn đến hiệu suất tính toán thấp.
    - PCA chọn ra các chiều quan trọng nhất (*Principal Components*) để giữ lại phần lớn thông tin.
  - **Ví dụ:** PCA giảm vector từ hàng ngàn chiều xuống còn 256 chiều (`n_components=256`).

3 Mô hình

3.1 Random Forest Classifier

3.1.1 Tổng quan về mô hình

Random Forest là một thuật toán học máy thuộc nhóm *ensemble learning*, tức là sự kết hợp nhiều mô hình cơ bản (ở đây là Decision Trees - các cây quyết định) để tạo ra mô hình mạnh hơn. Random Forest hoạt động dựa trên hai ý tưởng chính:

- Bagging (Bootstrap Aggregating):
  - Dữ liệu huấn luyện được lấy mẫu ngẫu nhiên nhiều lần với việc thay thế (with replacement), tạo ra các tập dữ liệu con. Mỗi cây quyết định được huấn luyện trên một tập dữ liệu con.
  - Bagging giúp giảm phương sai (variance) của mô hình, làm cho kết quả ít bị ảnh hưởng bởi các dữ liệu bất thường.
- Random Feature Selection:
  - Tại mỗi nút trong cây quyết định, chỉ một tập hợp con của các đặc trưng được chọn ngẫu nhiên để tìm kiếm thuộc tính phân chia tốt nhất.
  - Điều này làm cho các cây quyết định trong rừng có sự đa dạng, giúp giảm nguy cơ quá khớp (overfitting).

Decision Tree là thuật toán cấu tạo nên Random Forest. Nó hoạt động bằng cách chia dữ liệu dựa trên các thuộc tính để tạo thành các nút và nhánh, với mục tiêu tối ưu hóa một hàm chi phí. Cụ thể, với đề tài của nhóm:

- Dữ liệu đầu vào là các bình luận đánh giá sản phẩm (comment) và số sao (star) phù hợp tương ứng.
- Một cây quyết định sẽ cố gắng tìm các từ khóa quan trọng (ví dụ: "good", "bad", "excellent"...) để phân loại đánh giá theo số sao.

Vậy Decision Tree sẽ dựa vào đâu để xác định được các từ khóa quan trọng, hay nói cách khác là từ khóa để dùng đó làm thuộc tính để phân lớp các ví dụ học gắn với nút đó?

Entropy đo lường mức độ hỗn loạn/không chắc chắn trong 1 tệp dữ liệu,
với công thức tổng quát:

$$\text{Entropy}(S) = -\sum_{i=1}^{c} p_i \log_2 p_i$$

Trong đó, p_i là tỷ lệ các ví dụ trong tập S thuộc vào lớp i, và $0 \log_2 0 = 0$.

- Entropy = 0, nếu tất cả các ví dụ thuộc cùng một lớp.

- Entropy = 1, nếu các lớp dữ liệu phân bố đều nhau.
- Entropy = (0,1), nếu số lượng các ví dụ trong các lớp khác nhau.

Ví dụ: Giả sử có 100 đánh giá phân bố như sau:

- 1 sao: 10 đánh giá (p1 = 0.1)
- 2 sao: 15 đánh giá (p2 = 0.15)
- 3 sao: 25 đánh giá (p3 = 0.25)
- 4 sao: 30 đánh giá (p4 = 0.3)
- 5 sao: 20 đánh giá (p5 = 0.2)

Khi đó,

$$\text{Entropy} = -(0.1 \cdot \log_2(0.1) + 0.15 \cdot \log_2(0.15) + 0.25 \cdot \log_2(0.25) + 0.3 \cdot \log_2(0.3) + 0.2 \cdot \log_2(0.2)) \approx 2.21$$

Information Gain (IG) đo lường mức độ giảm entropy sau khi phân chia dữ liệu theo một thuộc tính.

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$$

Trong đó, Values(A) là tập các giá trị có thể của thuộc tính A, và

$$S_v = \{x \mid x \in S,\, x_A = v\}.$$

Ví dụ, xét từ khóa "excellent" trong 100 đánh giá:

- 20 đánh giá có từ "excellent":
  - 15 đánh giá 5 sao ($p_5 = 0.75$)
  - 4 đánh giá 4 sao ($p_4 = 0.2$)
  - 1 đánh giá 3 sao ($p_3 = 0.05$)

  $$\text{Entropy("excellent")} = -(0.75 \cdot \log_2(0.75) + 0.2 \cdot \log_2(0.2) + 0.05 \cdot \log_2(0.05)) \approx 0.97$$
- 80 đánh giá không có từ "excellent":
  - Giả sử phân bố đều hơn giữa các mức sao

    $$\text{Entropy("không excellent")} \approx 2.15$$

Khi đó (Gain):

$$\text{Gain} = 2.21 - \left(\frac{20}{100} \cdot 0.97 + \frac{80}{100} \cdot 2.15 \right) \approx 0.28$$

Điều này cho thấy từ "excellent" là một từ khóa tốt để phân chia vì:

- Khi có từ này, đánh giá thường có xu hướng là 4-5 sao
- Giảm đáng kể entropy của tập dữ liệu
- Gain dương và tương đối cao (0.28)

Tuy nhiên, Decision Tree đơn lẻ có thể dễ bị quá khớp (overfitting) với dữ liệu huấn luyện, tức là nó học rất kỹ các chi tiết của dữ liệu nhưng không tổng quát hóa tốt trên dữ liệu mới. Random Forest khắc phục nhược điểm này bằng cách:

- Tạo ra nhiều cây quyết định khác nhau từ các mẫu ngẫu nhiên của dữ liệu huấn luyện.
- Bỏ phiếu đa số từ kết quả của các cây để đưa ra kết quả cuối cùng.

Với những đặc điểm trên, nhóm quyết định sử dụng Random Forest là một trong hai thuật toán để giải quyết vấn đề của bài toán.

3.1.2 Cài đặt mô hình

Để cài đặt huấn luyện cho mô hình, trước hết cần import và biết cách sử dụng các thư viện.

1. Thư viện NLTK

a. Stopwords

Chức năng của module stopwords: Module này cung cấp một tập hợp các từ thường xuyên xuất hiện trong ngôn ngữ tự nhiên nhưng lại ít mang ý nghĩa ngữ cảnh, ví dụ: "the", "and", "is", "of", ... Những từ này được gọi là từ dừng (stop words).

Lợi ích: Loại bỏ từ dừng giúp tăng cường độ chính xác của kết quả, tạo ra các đặc trưng hiệu quả hơn cho các mô hình phân loại, và loại bỏ nhiễu, xác định các từ khoá quan trọng để đánh giá chính xác cảm xúc của văn bản.

Cơ chế hoạt động:

- Tải tập hợp từ dừng: Tải một tập hợp các từ dừng (stop words) mặc định cho một ngôn ngữ cụ thể (ví dụ: tiếng Anh) từ thư viện NLTK.
- Phân tích văn bản: Văn bản cần xử lý sẽ được chia thành các từ riêng biệt (tokenization).
- So sánh và loại bỏ: Mỗi từ trong văn bản sẽ được so sánh với tập hợp từ dừng đã tải. Nếu một từ xuất hiện trong cả hai tập hợp, nó sẽ bị loại bỏ khỏi văn bản.
- Tùy chỉnh: Tập hợp từ dừng mặc định có thể được tùy chỉnh bằng cách thêm hoặc xóa các từ theo yêu cầu cụ thể của bài toán.

b. Word_tokenizer

Chức năng: Chia đoạn văn thành các token, loại bỏ các dấu câu và các ký tự đặc biệt, giúp phân tích văn bản ở cấp độ từ, tập trung vào các từ có ý nghĩa.

Cơ chế: Hàm word_tokenize sử dụng các quy tắc ngữ pháp và thống kê để xác định ranh giới giữa các từ. Nó dựa vào các thuật toán phân tách từ (tokenization) để thực hiện việc này, giúp tạo ra các từ có ý nghĩa hơn cho các mô hình phân tích văn bản.

2. Thư viện Scikit-learn

a. TfidfVectorizer

TfidfVectorizer là một công cụ quan trọng trong xử lý ngôn ngữ tự nhiên (NLP), được sử dụng để chuyển đổi một tập hợp văn bản thành một ma trận số. Mỗi hàng trong ma trận này đại diện cho một văn bản, và mỗi cột đại diện cho một từ. Giá trị tại giao điểm của một hàng và một cột cho biết tầm quan trọng của từ đó trong văn bản tương ứng.

Chức năng:

- Biểu diễn văn bản dưới dạng số: Máy tính chỉ hiểu được số, vì vậy chúng ta cần chuyển đổi văn bản thành dạng số để máy tính có thể xử lý.
- Đánh giá tầm quan trọng của từ: Không phải tất cả các từ đều quan trọng như nhau. TfidfVectorizer giúp chúng ta xác định những từ nào quan trọng nhất trong một văn bản và trong toàn bộ tập hợp văn bản.

Nguyên lý hoạt động:

TfidfVectorizer dựa trên hai khái niệm chính:

- Tần số từ (TF): Là số lần một từ xuất hiện trong một văn bản. Một từ xuất hiện càng nhiều thì càng được coi là quan trọng trong văn bản đó.
- Tần số nghịch đảo của văn bản (IDF): Là logarit của tổng số văn bản chia cho số văn bản chứa từ đó. Một từ xuất hiện trong nhiều văn bản thì IDF của nó sẽ thấp, nghĩa là nó không đặc trưng cho một văn bản nào cả.

Công thức TF-IDF:

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \cdot \text{IDF}(t)$$

Trong đó:

- $\text{TF}(t, d)$: Tần số của từ t trong văn bản d.
- $\text{IDF}(t)$: Tần số nghịch đảo của văn bản của từ t.
- $\text{TF-IDF}(t, d)$: Trọng số TF-IDF của từ t trong văn bản d.

b. RandomForestClassifier

RandomForestClassifier là một thuật toán học máy mạnh mẽ thuộc họ thuật toán rừng ngẫu nhiên (Random Forest).
Nguyên lý hoạt động

- Cây quyết định: RandomForestClassifier xây dựng một rừng gồm nhiều cây quyết định. Mỗi cây quyết định được xây dựng trên một tập dữ liệu huấn luyện con, được chọn ngẫu nhiên từ tập dữ liệu gốc.
- Ngẫu nhiên hóa:
  - Ngẫu nhiên hóa dữ liệu: Mỗi cây quyết định được huấn luyện trên một tập dữ liệu con ngẫu nhiên, giúp giảm thiểu hiện tượng quá khớp (overfitting).
  - Ngẫu nhiên hóa thuộc tính: Khi xây dựng mỗi nút trong cây quyết định, chỉ một tập hợp con các thuộc tính được xem xét để tìm ra điểm phân chia tốt nhất. Điều này giúp tăng tính đa dạng của các cây quyết định và cải thiện khả năng tổng quát.
- Phân loại: Khi có một mẫu dữ liệu mới cần dự đoán, mỗi cây quyết định trong rừng sẽ đưa ra một dự đoán. Dự đoán cuối cùng được đưa ra bằng cách bỏ phiếu đa số: lớp được phân loại nhiều nhất bởi các cây quyết định sẽ là lớp dự đoán cuối cùng.

Các tham số quan trọng

- n_estimators: Số lượng cây quyết định trong rừng. Số lượng cây càng lớn thì mô hình càng ổn định nhưng thời gian huấn luyện cũng càng lâu.
- max_depth: Độ sâu tối đa của mỗi cây quyết định. Độ sâu càng lớn thì cây càng phức tạp và có khả năng quá khớp cao hơn.
- max_features: Số lượng thuộc tính được xem xét tại mỗi nút khi phân chia.

Ưu điểm:

- Độ chính xác cao: RandomForest thường đạt được độ chính xác cao trên nhiều loại dữ liệu.
- Khả năng tổng quát tốt: Nhờ vào việc ngẫu nhiên hóa, RandomForest có khả năng tổng quát tốt và ít bị ảnh hưởng bởi nhiễu trong dữ liệu.

- RandomForest có thể tự động lựa chọn các thuộc tính quan trọng nhất trong dữ liệu.

Hạn chế:

- Thời gian huấn luyện: Với số lượng cây lớn, thời gian huấn luyện có thể khá lâu.
- Random Forest là một mô hình hộp đen, khó giải thích tại sao lại đưa ra quyết định.

c. Classification_report

Mục đích: Cung cấp một báo cáo chi tiết về hiệu suất của mô hình phân loại trên từng lớp.
Các chỉ số chính:

- precision: Tỷ lệ các dự đoán đúng là dương trong tổng số các dự đoán dương. Nói cách khác, khi mô hình dự đoán một mẫu là dương, thì xác suất nó thực sự là dương là bao nhiêu.
- recall: Tỷ lệ các mẫu dương thực tế được mô hình dự đoán đúng. Nói cách khác, mô hình đã tìm thấy được bao nhiêu phần trăm các mẫu dương thực sự.
- f1-score: Là trung bình điều hòa của precision và recall, cho ta một đánh giá cân bằng về cả hai chỉ số trên.
- support: số lượng mẫu trong mỗi lớp.

d. Accuracy_score

Mục đích: Tính toán tỷ lệ các dự đoán đúng trên tổng số mẫu.
Ý nghĩa:

- Là một chỉ số đánh giá hiệu suất tổng thể của mô hình.
- Tuy nhiên, độ chính xác có thể không phải là chỉ số tốt nhất khi các lớp không cân bằng.
  Công thức tính Accuracy score

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

  Trong đó:

  - **TP** (True Positive): Số lượng mẫu mà mô hình dự đoán chính xác là đúng (true positive).
  - **TN** (True Negative): Số lượng mẫu mà mô hình dự đoán chính xác là sai (true negative).
  - **FP** (False Positive): Số lượng mẫu mà mô hình dự đoán sai là đúng (false positive).
  - **FN** (False Negative): Số lượng mẫu mà mô hình dự đoán sai là sai (false negative).

e. PCA

Chức năng: PCA (Principal Component Analysis) là một phương pháp giảm chiều dữ liệu, giúp tối ưu hóa việc biểu diễn dữ liệu nhiều chiều trong không gian ít chiều hơn mà vẫn giữ được nhiều thông tin nhất có thể. *Luồng hoạt động:*

- Chuẩn hóa dữ liệu (nếu cần): Đầu tiên, dữ liệu thường được chuẩn hóa để đảm bảo các đặc trưng có cùng tỷ lệ ảnh hưởng.
- Tính toán ma trận hiệp phương sai (Covariance Matrix): Phân tích mối quan hệ giữa các đặc trưng để xác định phương sai của từng chiều dữ liệu.

- Tính toán các thành phần chính (Principal Components): Sử dụng phân rã giá trị kỳ dị (SVD - Singular Value Decomposition) để tìm các vector riêng (eigenvectors) tương ứng với các giá trị riêng lớn nhất (eigenvalues).
- Chọn số chiều giảm (n_components): Giữ lại các thành phần chính có phương sai cao nhất theo yêu cầu.
- Ánh xạ dữ liệu gốc vào không gian các thành phần chính để tạo ra tập dữ liệu đã giảm chiều.

f. Train_test_split

Chức năng: chia tập dữ liệu ban đầu thành hai hoặc nhiều phần: tập huấn luyện (training set) và tập kiểm tra (test set), và có thể cả tập xác thực (validation set) nếu cần.
Nhiệm vụ:

- Tạo tập dữ liệu độc lập để đánh giá mô hình sau khi huấn luyện.
- Giảm nguy cơ overfitting (mô hình học quá chi tiết từ dữ liệu huấn luyện).
- Đảm bảo dữ liệu được chia ngẫu nhiên hoặc theo các tỷ lệ cụ thể (như phân phối nhãn).

Luồng hoạt động

- Xác định dữ liệu và nhãn (nếu có), sau đó chia dữ liệu theo các tập con với các tỷ lệ xác định. Nếu tham số random_state được cung cấp, việc chia dữ liệu sẽ cố định, giúp tái tạo được kết quả trong các lần chạy khác nhau.
- Hàm trả về các tập dữ liệu nhỏ hơn (thường là X_train, X_test, y_train, y_test).

-> train_test_split trong đoạn chia dữ liệu đầu vào thành hai phần: tập huấn luyện và tập kiểm tra. Quá trình chia được thực hiện ngẫu nhiên nhưng vẫn đảm bảo phân phối nhãn đồng đều. Điều này giúp mô hình được huấn luyện trên tập dữ liệu lớn hơn, đồng thời có một tập dữ liệu riêng biệt để đánh giá, tăng độ tin cậy cho kết quả mô hình.

3. Pickle

Thư viện pickle trong Python được sử dụng để tuần tự hóa (quá trình chuyển đổi các đối tượng Python thành chuỗi byte để có thể lưu trữ hoặc truyền qua mạng) và giải tuần tự hóa (quá trình khôi phục đối tượng Python từ chuỗi byte đã được tuần tự hóa) các đối tượng Python.

Đây là công cụ để lưu trữ và khôi phục dữ liệu một cách dễ dàng giữa các phiên làm việc của chương trình.

3.2 Hồi Quy Softmax

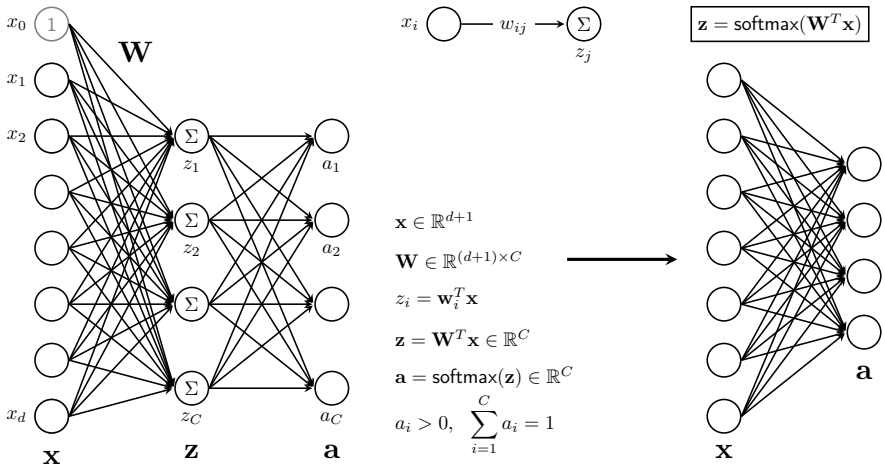
3.2.1 Tổng quan về mô hình

Hồi quy Softmax (hay Hồi quy Logistic đa thức) là một mô hình phân loại xác suất, dự đoán xác suất của một điểm dữ liệu thuộc vào một trong C lớp. Hàm xác suất được biểu diễn dưới dạng:

$$p(y|x;\theta) = \text{Cat}(y; \text{softmax}(Wx + b))$$

Trong đó:

- $x \in \mathbb{R}^D$ là vector đầu vào với D chiều.
- $y \in \{1, 2, \ldots, C\}$ là nhãn của điểm dữ liệu (với C lớp).
- $W \in \mathbb{R}^{C \times D}$ là ma trận trọng số.



Hình 1: Phân loại đa lớp với hồi quy logistic và one-vs-rest.

- $b \in \mathbb{R}^C$ là vector độ chệch.
- Hàm Softmax chuyển đầu ra thành xác suất, được định nghĩa như sau:

$$\text{softmax}\left(\mathbf{x}\right) := \left[\frac{\exp \mathbf{x}_i}{\sum_{k=1}^{C} \exp \mathbf{x}_k}, \ldots, \frac{\exp \mathbf{x}_C}{\sum_{k=1}^{C} \exp \mathbf{x}_k}\right]$$

Xác suất cho một lớp:
Xác suất $p(y = k|x; W, b)$ cho lớp k là:

$$p(y = k|x; W, b) = z_k = \text{softmax}(Wx + b)_k = \frac{\exp((Wx + b)_k)}{\sum_{j=1}^{C} \exp((Wx + b)_j)}$$

Hàm softmax () giúp ràng buộc đầu ra thoả mãn các điều kiện của một bộ tham số phân bố đa thức: các phần tử không âm, có thứ tự và tổng xác suất của các lớp bằng 1.

$$\sum_{k=1}^{C} p(y = k|x; W, b) = 1$$

Xác suất với nhãn One-Hot:
Khi nhãn y được biểu diễn dưới dạng one-hot, vector y có giá trị 1 tại chỉ số lớp đúng y_i và giá trị 0 tại các lớp còn lại. Khi đó, xác suất cho lớp k trở thành:

$$p(y = k|x; W, b) = \sum_{i=1}^{C} y_i z_i$$

với

$$y_i = \begin{cases} 1 & , i = k \\ 0 & , i \neq k \end{cases}$$

Vì là nhãn one-hot, biểu thức này có thể đơn giản hóa thành:

$$p(y = y_i|x; W, b) = z_{y_i} = \text{softmax}(Wx + b)_{y_i}$$

Ví dụ: "Chất lượng tệ, giao hàng nhanh."

Vector đặc trưng: $\mathbf{x} = \begin{bmatrix} 1.5 \\ -0.8 \\ 0.3 \\ 2.1 \end{bmatrix}$ *(đã qua xử lý bằng TF-IDF như mô tả ở phần xử lí dữ liệu).*

Ta áp dụng mô hình Softmax:

Trọng số $\mathbf{W}$ *(ma trận trọng số, kích thước* 5×4*)*:

$$\mathbf{W} = \begin{bmatrix} 0.1 & -0.2 & 0.3 & 0.4 \\ -0.3 & 0.8 & -0.5 & 0.2 \\ 0.7 & -0.6 & 0.1 & -0.1 \\ -0.5 & 0.4 & 0.2 & 0.3 \\ 0.2 & -0.1 & 0.6 & 0.5 \end{bmatrix}.$$

Độ chệch $\mathbf{b}$ *(vector độ chệch, kích thước* 5*)*:

$$\mathbf{b} = \begin{bmatrix} 0.2 \\ -0.1 \\ 0.3 \\ 0.0 \\ 0.1 \end{bmatrix}.$$

Kết quả tuyến tính $\mathbf{z} = \mathbf{W} \cdot \mathbf{x} + \mathbf{b}$*:*

$$\mathbf{z} = \begin{bmatrix} 1.19 \\ -0.31 \\ 1.05 \\ 0.43 \\ 1.87 \end{bmatrix}.$$

Hàm Softmax được áp dụng để chuyển đổi $\mathbf{z}$ *thành xác suất:*

$$\text{softmax}(z)_i = \frac{\exp(z_i)}{\sum_{j=1}^{5} \exp(z_j)}.$$

Xác suất sau khi chuẩn hóa:

$$\text{softmax}(\mathbf{z}) = \begin{bmatrix} 0.22 \\ 0.05 \\ 0.19 \\ 0.10 \\ 0.44 \end{bmatrix}.$$

Xác suất cao nhất là 0.44 *cho mức đánh giá 5 sao. Do đó, theo mô hình, đánh giá này nhiều khả năng được dự đoán là 5 sao.*

Hàm Mất Mát (Loss Function):

Việc tìm nghiệm cho bài toán hồi quy softmax chính là tìm các tham số W và b, ta áp dụng phương pháp cực đại hóa likelihood (Maximum Likelihood Estimation - MLE). Xác suất tổng quát của mô hình cho tập dữ liệu D gồm N mẫu là:

$$p(y|x; W, b) = \prod_{i=1}^{N} p(y_i|x_i; W, b)$$

Log-likelihood cho tập dữ liệu là:

$$\log p(y|x; W, b) = \sum_{i=1}^{N} \log p(y_i|x_i; W, b)$$

Log-likelihood chi tiết là:

$$\log p(y|x; W, b) = \sum_{i=1}^{N} \log \left(\sum_{k=1}^{C} y_{ik} \cdot z_{ik} \right)$$

Vì y_i là one-hot, ta có:

$$\log p(y_i|x_i; W, b) = \log \left(z_{y_i} \right)$$

Do đó, log-likelihood tổng quát là:

$$\log p(y|x; W, b) = \sum_{i=1}^{N} \log \left(z_{y_i} \right)$$

Cực Đại Hóa Log-Likelihood:
Mục tiêu của mô hình là tối đa hóa xác suất dự đoán (log-likelihood) trên tập huấn luyện. Để tìm tham số W và b, ta áp dụng phương pháp cực đại hóa log-likelihood:

$$\begin{aligned} W^*, b^* &= \arg\max_{W,b} \log p(y|x; W, b) \\ &= \arg\max_{W,b} \log \sum_{i=1}^{C} y_i z_i \end{aligned}$$

Thay vì cực đại hoá log likelihood, có thể cực tiểu hoá cận trên của âm log likelihood (là một hàm cross-entropy) dễ tính toán hơn:

$$W^*, b^* = \arg\min_{W,b} - \sum_{i=1}^{C} y_i \log z_i$$

Với tập dữ liệu $\mathcal{D}$, hàm mất mát cần cực tiểu có dạng

$$\mathcal{L}(W, b) = \mathbb{E}_{\mathbf{x}, y \sim \mathcal{D}} \left[- \sum_{i=1}^{C} y_i \log z_i \right]$$

Lý do chọn hàm mất mát này: Hàm mất mát cross-entropy kết hợp với softmax là lựa chọn phổ biến và hiệu quả trong các bài toán phân loại đa lớp. Nó tối thiểu hóa sự khác biệt giữa xác suất dự đoán của mô hình và nhãn thực tế, giúp cải thiện độ chính xác của mô hình.
Regularization và Weight Decay:
Để tránh overfitting, ta thêm regularization vào hàm mất mát, chấp nhận hy sinh độ chính xác trong training set, nhưng giảm độ phức tạp của mô hình, thường là L2 regularization:

$$R(W) = \frac{\lambda}{2} \|W\|^2$$

Hàm mất mát tổng quát trở thành:

$$\mathcal{L}_{\text{reg}}(W, b) = \mathcal{L}(W, b) + \mathcal{R}(W)$$

Với hàm mất mát $\mathcal{L}(W, b)$ là cross-entropy:

$$\mathcal{L}_{\text{reg}}(W, b) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{C} y_{ik} \log(z_{ik}) + \frac{\lambda}{2} \|W\|^2$$

Cập Nhật Trọng Số:
Sử dụng thuật toán Gradient Descent để cập nhật các tham số W và b. Các gradient được tính như sau:

- Gradient theo W:
$$\frac{\partial L_{\text{reg}}}{\partial W} = \frac{1}{N}\sum_{i=1}^{N}(z_i - y_i)x_i^T + \lambda W$$
- Gradient theo b:
$$\frac{\partial L_{\text{reg}}}{\partial b} = \frac{1}{N}\sum_{i=1}^{N}(z_i - y_i)$$

Cập nhật trọng số:
$$W \leftarrow W - \eta\frac{\partial L_{\text{reg}}}{\partial W}, \quad b \leftarrow b - \eta\frac{\partial L_{\text{reg}}}{\partial b}$$
Trong đó, η là tốc độ học (learning rate).

Chứng minh Gradient:

- *Xác suất softmax cho mỗi mẫu i:*
$$z_{ij} = \text{softmax}_j(Wx_i) = \frac{\exp(w_j^T x_i)}{\sum_{j'}\exp(w_{j'}^T x_i)}$$
Trong đó:
  - w_j^T*: hàng thứ j của ma trận W.*
  - x_i*: vector đặc trưng của mẫu i.*
- *Gradient với từng hàng w_j:* Gradient của L_i với w_j được tính như sau:
$$\nabla_{w_j} L_i = -\frac{\partial}{\partial w_j}\left(y_{ij}\log z_{ij}\right)$$
Áp dụng quy tắc chuỗi và tính toán từng thành phần:
$$\begin{aligned}\frac{\partial}{\partial w_j}\log z_{ij} &= \frac{\partial}{\partial w_j}\left(w_j^T x_i - \log\sum_{j'}\exp(w_{j'}^T x_i)\right)\\ &= x_i - z_{ij}x_i\end{aligned}$$
Từ đó:
$$\nabla_{w_j} L_i = (z_{ij} - y_{ij})x_i$$
- *Gradient toàn bộ W:* Với N mẫu, gradient được lấy trung bình:
$$\nabla_W L = \frac{1}{N}\sum_{i=1}^{N}\left[(z_i - y_i)x_i^T\right]$$
- *Biểu diễn dưới dạng ma trận:* Gọi:
  - X*: ma trận đặc trưng ($d \times N$).*
  - Z*: ma trận xác suất ($C \times N$).*
  - Y*: ma trận nhãn ($C \times N$).*

  Gradient có thể viết gọn:
$$\nabla_W L = \frac{1}{N}(Z - Y)X^T$$

- *Thêm thành phần điều chuẩn:* Khi thêm thành phần điều chuẩn $\frac{\lambda}{2}\|W\|^2$, gradient cuối cùng là:

$$\frac{\partial L_{\text{reg}}}{\partial W} = \frac{1}{N}(Z - Y)X^T + \lambda W$$

Lý do chọn Softmax: Mô hình này được chọn vì tính chất phân loại đa lớp của Softmax, giúp dự đoán xác suất của một điểm dữ liệu thuộc vào một trong nhiều lớp, điều này là rất cần thiết trong các bài toán phân loại với nhiều nhóm mục tiêu. Softmax cung cấp một xác suất duy nhất cho mỗi lớp, đảm bảo rằng tổng tất cả xác suất của các lớp bằng 1, giúp mô hình phân loại rõ ràng và có thể giải thích.

Mô hình hồi quy softmax sử dụng cross-entropy và regularization giúp đạt hiệu quả tốt trong các bài toán phân loại đa lớp.

3.2.2 Cài đặt mô hình

1. Pandas

Chức năng:

- Xử lý dữ liệu dạng bảng: Pandas hỗ trợ làm việc với dữ liệu dạng bảng (dữ liệu có hàng và cột) thông qua các đối tượng như DataFrame và Series.
- Đọc và ghi dữ liệu: Pandas hỗ trợ đọc dữ liệu từ nhiều định dạng tệp như CSV, Excel, JSON, SQL, v.v.

2. Torch

Chức năng và nhiệm vụ:

- Xử lý Tensor: Torch cung cấp đối tượng torch.Tensor, hỗ trợ tính toán mảng đa chiều tương tự NumPy nhưng mạnh mẽ hơn nhờ khả năng chạy trên GPU, có thể lập trình song song được. Torch hỗ trợ tính đạo hàm.
- Xây dựng mô hình học sâu: Cung cấp các công cụ để thiết kế, huấn luyện, và tối ưu hóa mạng neural (Neural Networks).
- Hỗ trợ GPU: Tăng tốc xử lý nhờ tích hợp với CUDA, giúp thực hiện các phép tính trên GPU hiệu quả.

Luồng hoạt động

- Khởi tạo dữ liệu: Dữ liệu từ các nguồn (như NumPy arrays hoặc DataFrame) được chuyển đổi thành torch.Tensor. Điều này giúp dữ liệu sẵn sàng cho các mô hình học sâu sử dụng Torch.
- Tính toán trên Tensor: Tensor được sử dụng để thực hiện các phép toán hoặc làm đầu vào cho các mô hình Torch như mạng nơ-ron.
- Huấn luyện và đánh giá mô hình:

  Tensor lưu trữ dữ liệu đầu vào (x) và nhãn (y), làm cơ sở để huấn luyện mô hình (training) và kiểm tra hiệu suất (evaluation).

Ngoài các thư viện kể trên, đối với mô hình Softmax-Regression, ta cũng sử dụng các thư viện như scikit-learn, (TfidfVectorizer, PCA, train_split_test), thư viện Pickle,.. đã được mô tả chi tiết ở mục 3.1.2.

4 Đánh giá kết quả và phân tích

Kết quả được thử nghiệm trên 3 bộ dữ liệu với 2 mô hình, được thể hiện qua các bảng sau.

- Bộ dữ liệu mẫu trên HuggingFace

Số lượng data	Random Forest Classifier	Softmax-Regression
100 comments cho mỗi class	train = 1, test = 0.66	train = 1, test = 0.65
200 comments cho mỗi class	train = 1, test = 0.54	train = 1, test = 0.63

Bảng 3: Kết quả accuracy với bộ dữ liệu mẫu trên HuggingFace

Class	Precision	Recall	F1-Score	Support
1	0.70	0.57	0.63	28
2	0.59	0.71	0.65	14
3	0.47	0.70	0.56	10
4	0.70	0.79	0.75	24
5	0.78	0.58	0.67	24

Bảng 4: Classification Report: Mỗi class gồm 100 comments

Class	Precision	Recall	F1-Score	Support
1	0.38	0.55	0.45	33
2	0.87	0.50	0.63	52
3	0.35	0.53	0.42	34
4	0.61	0.61	0.61	38
5	0.62	0.49	0.55	43

Bảng 5: Classification Report: Mỗi class gồm 200 comments

- Bộ dữ liệu do nhóm tự crawl và xây dựng

Số lượng data	Random Forest Classifier	Softmax-Regression
100 comments cho mỗi class	train = 1, test = 0.44	train = 1, test = 0.56
200 comments cho mỗi class	train = 1, test = 0.5	train = 1, test = 0.58

Bảng 6: Kết quả accuracy với bộ dữ liệu của nhóm

Class	Precision	Recall	F1-Score	Support
1	0.47	0.25	0.33	28
2	0.21	0.43	0.28	14
3	0.16	0.30	0.21	10
4	0.73	0.33	0.46	24
5	0.77	0.83	0.80	24

Bảng 7: Classification Report: Mỗi class gồm 100 comments

Class	Precision	Recall	F1-Score	Support
1	0.41	0.42	0.42	33
2	0.68	0.48	0.56	48
3	0.31	0.35	0.33	37
4	0.32	0.32	0.32	38
5	0.72	0.86	0.78	44

Bảng 8: Classification Report: Mỗi class gồm 200 comments

- Bộ dữ liệu mix (50-50)

Số lượng data	Random Forest Classifier	Softmax-Regression
100 comment cho mỗi class	train = 1, test = 0.47	train = 1, test = 0.59
200 comment cho mỗi class	train = 1, test = 0.56	train = 1, test = 0.62

Bảng 9: Kết quả accuracy với bộ dữ liệu mix

Class	Precision	Recall	F1-Score	Support
1	0.59	0.46	0.52	28
2	0.43	0.43	0.43	14
3	0.15	0.30	0.20	10
5	0.71	0.42	0.43	24
4	0.43	0.62	0.67	24

Bảng 10: Classification Report: Mỗi class gồm 100 comments

Class	Precision	Recall	F1-Score	Support
1	0.47	0.55	0.51	33
2	0.67	0.38	0.48	48
3	0.47	0.62	0.53	37
5	0.47	0.55	0.51	38
4	0.78	0.73	0.75	44

Bảng 11: Classification Report: Mỗi class gồm 200 comments

Dataset mà nhóm tự thu thập có những hạn chế nhất định, ảnh hưởng đến hiệu quả của quá trình huấn luyện mô hình. Một số vấn đề cụ thể bao gồm:

- Số lượng dữ liệu: Dataset do nhóm xây dựng có quy mô nhỏ hơn so với dataset mẫu từ HuggingFace, dẫn đến việc mô hình không đủ thông tin để học các đặc điểm phức tạp.
- Chất lượng dữ liệu: Dữ liệu tự thu thập có thể không đa dạng, có sự chênh lệch về cấu trúc câu hoặc biểu đạt ý kiến.
- Các bước tiền xử lý thủ công có thể chưa tối ưu, làm mất mát một phần thông tin quan trọng.

Kết quả là khi cùng một mô hình được huấn luyện trên dataset tự thu thập, độ chính xác (accuracy) đạt được thấp hơn so với khi sử dụng dataset mẫu. Điều này thể hiện rõ qua các báo cáo đánh giá như các bảng đã trình bày.

So sánh tổng quan giữa 2 mô hình dựa trên kết quả:

Softmax Regression:

- Là một mô hình xác suất, Softmax Regression có khả năng xử lý tốt hơn trong các tập dữ liệu bị nhiễu nhờ vào bản chất tuyến tính và khả năng tối ưu hóa thông qua các regularization trick như L1, L2.
- Mô hình này vững vàng hơn với dữ liệu bị chồng lấp hoặc chứa nhiều yếu tố không liên quan.
- Dễ hiểu và dễ điều chỉnh, nhưng khó nắm bắt các quan hệ phức tạp trong dữ liệu.

Random Forest:

- Là một mô hình "hộp đen" dựa trên nhiều cây quyết định (Decision Trees), RF thường cho kết quả tốt trên các tập dữ liệu nhỏ nhờ khả năng tổng quát hóa cao.
- Tuy nhiên, RF gặp khó khăn trong việc diễn giải kết quả và nhạy cảm hơn với dữ liệu nhiễu, đặc biệt khi dữ liệu chứa các đặc điểm không nhất quán hoặc không đủ thông tin.

Kết quả đánh giá cho thấy, Softmax Regression có phần ổn định hơn trên dữ liệu nhiễu so với Random Forest, mặc dù sự khác biệt phụ thuộc nhiều vào bản chất của dataset.

Cả hai mô hình đều có những hạn chế đáng kể khi áp dụng vào bài toán phân loại 5 lớp số sao :

- Phụ thuộc cấu trúc câu: Bài toán phân loại số sao không chỉ dựa vào các từ khóa riêng lẻ mà còn phụ thuộc vào cấu trúc câu, sắc thái cảm xúc (sentiment), và ngữ cảnh phức tạp.
- Biểu diễn dữ liệu: Phương pháp sử dụng TF-IDF để biểu diễn văn bản còn đơn giản, không thể nắm bắt được mối quan hệ ngữ nghĩa sâu hoặc biểu đạt phức tạp.
- Cả Softmax Regression và Random Forest đều là các mô hình truyền thống, thiếu khả năng tổng quát hóa để xử lý những bài toán mang tính phức tạp cao trong ngôn ngữ tự nhiên.

5 Kết luận

5.1 Tóm tắt kết quả xây dựng hệ thống

Trong quá trình thực hiện, nhóm đã tiến hành xây dựng một pipeline xử lý bài toán dự đoán số sao dựa trên nội dung bình luận.

- Nhóm đã tự thu thập và xây dựng một dataset riêng để huấn luyện mô hình, đồng thời sử dụng các dataset mẫu từ các nguồn như HuggingFace để đối chiếu. Dữ liệu được xử lý qua nhiều bước như làm sạch, loại bỏ từ dừng, chuẩn hóa văn bản, và biểu diễn dữ liệu bằng TF-IDF. Tuy nhiên, dataset tự xây dựng còn hạn chế về quy mô và chất lượng, ảnh hưởng đến kết quả mô hình.
- Hai mô hình chính được sử dụng là Random Forest và Softmax Regression, mỗi mô hình đều có ưu nhược điểm riêng. Random Forest có khả năng tổng quát hóa tốt trên các tập dữ liệu nhỏ nhưng nhạy cảm với nhiễu, trong khi Softmax Regression vững vàng hơn với nhiễu và dễ dàng tối ưu hóa với các thủ thuật hiệu chỉnh.

Tổng kết, nhóm đã đạt được những kết quả nhất định và có cái nhìn sâu hơn về bài toán cũng như các thách thức đi kèm. Đây sẽ là cơ sở để nhóm tiếp tục nghiên cứu và cải thiện trong các bước tiếp theo.

5.2 Thách thức và hạn chế

Trong quá trình làm dự án, nhóm đã gặp nhiều khó khăn và hạn chế qua nhiều giai đoạn:

5.2.1 Khó khăn trong việc thu thập dữ liệu (Crawl Data)

- **Bị chặn truy cập**: Khi tiến hành crawl dữ liệu từ các sàn thương mại điện tử như Shopee và Lazada, nhóm liên tục bị chặn IP do các trang này áp dụng các biện pháp chống bot mạnh mẽ.
- **Lượng dữ liệu thu thập được ít**: Do bị chặn và thiếu kinh nghiệm, nhóm chỉ thu thập được một lượng nhỏ dữ liệu ban đầu, không đủ để huấn luyện mô hình.
- **Không crawl được trong giai đoạn đầu**: Ở giai đoạn bắt đầu, nhóm chưa quen với các công cụ như Python Selenium hoặc Scrapy. Điều này dẫn đến việc mất nhiều thời gian để hiểu cách hoạt động của các công cụ và khắc phục lỗi kỹ thuật.

Giải pháp:

- Sau khi gặp khó khăn với Shopee và Lazada, nhóm đã chuyển hướng tìm kiếm các sàn thương mại điện tử quốc tế khác. Cuối cùng, nhóm chọn Amazon vì sàn này có cấu trúc trang đơn giản hơn và ít hạn chế khi crawl dữ liệu.
- Nhóm áp dụng các kỹ thuật để tránh bị chặn như sử dụng proxy, thay đổi User-Agent, và giảm tốc độ crawl để mô phỏng hành vi người dùng.
- Nhóm cũng tận dụng các API miễn phí khi có thể để hỗ trợ việc thu thập dữ liệu.

5.2.2 Mất cân bằng dữ liệu (Data Imbalance)

- Sau khi thu thập dữ liệu, nhóm nhận thấy số lượng dữ liệu giữa các nhãn (class) không đồng đều. Một số nhãn có số lượng rất lớn, trong khi các nhãn khác chỉ có vài trường hợp.
- Điều này khiến mô hình có xu hướng ưu tiên các nhãn có tần suất cao, dẫn đến kết quả không chính xác với các nhãn ít xuất hiện.

Giải pháp:

- Nhóm quyết định mở rộng phạm vi crawl dữ liệu để thu thập thêm các trường hợp thuộc các nhãn hiếm. Việc này giúp cân bằng dữ liệu tốt hơn và cải thiện chất lượng của mô hình.
- Khi huấn luyện mô hình, nhóm sử dụng các hàm mất mát có trọng số (weighted loss functions) để giảm thiểu tác động của mất cân bằng dữ liệu.

5.2.3 Khó khăn trong việc xây dựng và sử dụng mô hình

- **Làm quen với các nền tảng mới**: Nhóm gặp khó khăn khi làm quen với Google Colab và Kaggle, từ việc cấu hình môi trường đến quản lý tài nguyên GPU/TPU.
- **Hiểu chi tiết các mô hình**: Các mô hình học máy và học sâu như Logistic Regression, Random Forest, và các mạng nơ-ron (Neural Networks) đòi hỏi kiến thức sâu rộng. Nhóm gặp khó khăn trong việc hiểu cách các mô hình hoạt động và tinh chỉnh siêu tham số.
- **Hiệu quả mô hình chưa cao**: Ở giai đoạn đầu, các mô hình được triển khai cho kết quả không như kỳ vọng, cần nhiều thời gian để cải thiện.

Giải pháp:

- Nhóm đã tham gia các khóa học trực tuyến trên Coursera và YouTube để làm quen với cách sử dụng Google Colab và Kaggle.
- Sử dụng các tài liệu hướng dẫn chi tiết từ scikit-learn và TensorFlow để hiểu rõ hơn về cách hoạt động của các mô hình và thuật toán.
- Thực hiện nhiều thử nghiệm nhỏ với các bộ dữ liệu khác để tăng cường kỹ năng triển khai mô hình và tinh chỉnh siêu tham số.

5.3 Định hướng phát triển

Trong tương lai, nhóm sẽ tiếp tục cải thiện và mở rộng dự án theo các hướng sau để nâng cao chất lượng và khả năng ứng dụng thực tế. Một trong những mục tiêu quan trọng là mở rộng nguồn dữ liệu. Mặc dù nhóm đã thu thập dữ liệu từ Amazon, nhưng việc mở rộng phạm vi thu thập từ các nền tảng thương mại điện tử khác, như eBay, Walmart, và các sàn quốc tế, sẽ giúp làm phong phú thêm bộ dữ liệu, cải thiện độ chính xác và khả năng tổng quát của mô hình. Đồng thời, nhóm sẽ tiếp tục cải thiện các phương pháp thu thập dữ liệu để đảm bảo có được thông tin phong phú và đa dạng hơn, từ đó hỗ trợ các mô hình phân tích chính xác hơn.

Một vấn đề quan trọng mà nhóm sẽ tập trung giải quyết trong tương lai là việc xử lý dữ liệu mất cân bằng. Cải thiện cân bằng dữ liệu sẽ là một yếu tố quan trọng trong việc nâng cao hiệu quả của mô hình, đồng thời giảm thiểu sự thiên lệch trong các kết quả dự đoán.

Bên cạnh đó, nhóm sẽ tiếp tục tinh chỉnh các mô hình học máy, đặc biệt là trong việc tối ưu hóa siêu tham số. Sử dụng các phương pháp như Grid Search, Random Search, hay Bayesian Optimization để tối ưu hóa các tham số của mô hình sẽ giúp nhóm đạt được hiệu suất cao hơn. Ngoài ra, nhóm có thể sẽ thử nghiệm với các mô hình học sâu phức tạp hơn như CNNs hoặc LSTMs, đặc biệt khi làm việc với dữ liệu văn bản dài hoặc chuỗi thời gian. Các mô hình học sâu này có thể mang lại những kết quả tốt hơn trong việc xử lý và phân loại dữ liệu.

6 Tài liệu tham khảo

References

[1] Aurélien Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*, O'Reilly Media, 2019.

[2] Steven Bird, Ewan Klein, Edward Loper, *Natural Language Processing with Python*, O'Reilly Media, 2009.

[3] Python libraries, *Scikit-learn Documentation, Scikit-learn Documentation, Pandas Documentation, NumPy Documentation, TF-IDF Documentation*

[4] Haibo He, Edwardo A. Garcia, *Imbalanced Data Handling Techniques*, IEEE Transactions on Knowledge and Data Engineering, 2009

[5] *Softmax Regression for Multiclass Classification*, Coursera, 2011

[6] Philip Thomas, *Practical Hyperparameter Optimization*, ArXiv.org, 2019

[7] Nguyễn Thị Kim Anh, *Slide bài giảng Nhập môn Học máy và Khai phá dữ liệu*