

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO BÀI TẬP THỰC HÀNH 01

ĐỀ BÀI: TIỀN XỬ LÝ DỮ LIỆU

KHAI THÁC DỮ LIỆU & ỨNG DỤNG - CSC14004

NGƯỜI HƯỚNG DẪN

GS. LÊ HOÀI BẮC
ThS. LÊ NHỰT NAM
ThS. NGUYỄN NGỌC ĐỨC

THÔNG TIN SINH VIÊN

MSSV	Họ và Tên	Email
23127187	Trần Lý Nhật Hào	tlnhao23@clc.fitus.edu.vn
23127384	Huỳnh Lê Duy Khánh	hldkhanh23@clc.fitus.edu.vn
22127322	Lê Phước Phát	lpphat22@clc.fitus.edu.vn

THÀNH PHỐ HỒ CHÍ MINH - THÁNG 10, 2025

LỜI NÓI ĐẦU

Trong bối cảnh bùng nổ dữ liệu số, việc khai thác và phân tích dữ liệu hiệu quả trở thành yếu tố then chốt giúp doanh nghiệp nâng cao năng lực quản trị, tiếp thị và cải thiện trải nghiệm khách hàng. Chính vì vậy, việc tiền xử lý dữ liệu là một bước căn bản cực kỳ quan trọng trong bất kỳ hệ thống khai thác dữ liệu tự động và học máy, học sâu, hay các công nghệ trí tuệ nhân tạo hiện đại ngày nay. Thông thường những dữ liệu thô ngoài thực tế thường không có giá trị sâu sắc. Chúng thường có nhiều dạng khác nhau từ các đầu vào khác nhau như các cảm biến, ứng dụng, ... Đối với từng dạng và cấu trúc dữ liệu thô đầu vào, chúng yêu cầu các kỹ thuật tiền xử lý khác nhau.

Xuất phát từ nhu cầu hiện thực và cấp thiết đó, thông qua bài tập thực hành này, bài tập này sẽ cung cấp các kiến thức và kỹ năng cần thiết để xử lý các dạng dữ liệu sau thông qua các chương sau:

- **CHƯƠNG 01. Tiền xử lý dữ liệu hình ảnh**

- Ở cấp độ này, chúng ta cần phải tiền xử lý dữ liệu với đầu vào là những điểm pixel có thể chuyển hóa được.

- **CHƯƠNG 02. Tiền xử lý dữ liệu dạng bảng**

- Ở cấp độ này, chúng ta sẽ thực hiện việc xử lý dữ liệu đối với dữ liệu có cấu trúc (dạng bảng) với những cột và hàng với nhiều loại dữ liệu khác nhau.

- **CHƯƠNG 03. Tiền xử lý dữ liệu dạng văn bản**

- Ở cấp độ này, chúng ta sẽ thực hiện việc xử lý dữ liệu không có cấu trúc cố định mà dữ liệu này yêu cầu sự hiểu ngôn ngữ một cách tự nhiên.

Tóm lại, thông qua bài tập này, chúng ta sẽ phát triển những kỹ năng thực hành bằng cách áp dụng các kỹ thuật tiền xử lý thích hợp đối với nhiều dạng dữ liệu khác nhau. Đồng thời, chúng ta có thể hiểu và vận dụng những kỹ thuật này cho những tác vụ đòi hỏi việc phân tích và khai thác dữ liệu phức tạp.

Thành phố Hồ Chí Minh, Mùa hè 2025.

LỜI CAM ĐOAN

Nhóm số 05 thực hiện bài tập thực hành số 01 về chủ đề **Tiền xử lý dữ liệu** gồm các thành viên **Lê Phước Phát, Lý Nhật Hào, và Huỳnh Lê Duy Khánh** đều là sinh viên thuộc khoa Công nghệ Thông tin Chất lượng cao, thuộc trường Đại học Khoa học Tự nhiên, ĐHQG-HCM. Nhóm cam đoan rằng bài tập nghiên cứu này là do các thành viên trong nhóm tìm hiểu, nghiên cứu và thực hiện dưới sự giám sát và hướng dẫn của thầy **GS. LÊ HOÀI BẮC**, thầy **ThS. LÊ NHỰT NAM**, và thầy **ThS. NGUYỄN NGỌC ĐỨC**. Các dữ liệu được nêu trong đồ án là hoàn toàn trung thực, phản ánh đúng kết quả mô phỏng thực tế. Tất cả các tài liệu được sử dụng trong nghiên cứu này được các thành viên trong nhóm thu thập bằng cách tự thân và từ các nguồn khác nhau, và những tài liệu này được liệt kê đầy đủ trong phần tài liệu tham khảo. Tất cả đều được trích dẫn đúng đắn. Trong trường hợp có vi phạm bản quyền, các thành viên trong nhóm sẽ chịu trách nhiệm cho hành động đó. Do đó, trường **Đại học Khoa học Tự nhiên, ĐHQG-HCM** không chịu trách nhiệm về bất kỳ vi phạm bản quyền nào được thực hiện trong bài tập nghiên cứu này.

TP.HCM, ngày 28 tháng 10 năm 2025

Người cam đoan

Nhóm trưởng

TRẦN LÝ NHẬT HÀO

MỤC LỤC

DANH MỤC HÌNH VẼ	i
DANH MỤC BẢNG BIỂU	ii
THÔNG TIN NHÓM & BẢNG PHÂN CÔNG	iii
Thông Tin Nhóm	iii
Bảng Phân Công Công Việc	iii
CHƯƠNG 01. TIỀN XỬ LÝ DỮ LIỆU HÌNH ẢNH	1
1.1 Mô tả dữ liệu	1
1.1.1 Giới thiệu về tập dữ liệu	1
1.1.2 Phân tích tổng quan dữ liệu	2
1.1.3 Lý do chọn tập dữ liệu	2
1.2 Cơ sở tiền xử lý	3
1.3 Triển khai chi tiết	3
1.3.1 Loading & Resizing (Requirement a)	3
1.3.2 Grayscale Conversion (Requirement b)	3
1.3.3 Normalization (Requirement c)	4
1.3.4 Histogram Equalization, CLAHE và Gaussian Blur	4
1.3.5 Edge Detection (Sobel, Prewitt, Canny)	5
1.4 Phân tích & đánh giá kết quả	6
1.4.1 Phân tích định lượng	6
1.4.2 So sánh trực quan và diễn giải kết quả	6
1.4.3 Tác động đến chất lượng dữ liệu	6
1.5 Kết luận	6
CHƯƠNG 02. TIỀN XỬ LÝ DỮ LIỆU VĂN BẢN	7
2.1 Mô tả dữ liệu	7
2.1.1 Giới thiệu về tập dữ liệu	7
2.1.2 Hướng dẫn cài đặt dữ liệu	9
2.1.3 Phân tích tổng quan dữ liệu	9
2.1.4 Lý do chọn tập dữ liệu này	10
2.2 Cơ sở tiền xử lý	11
2.3 Triển khai chi tiết	12
2.4 Phân tích & đánh giá kết quả	22
2.4.1 Phân tích kết quả định lượng	22
2.4.2 So sánh trực quan và diễn giải kết quả	22

2.4.3	Phân tích tác động về chất lượng dữ liệu	23
CHƯƠNG 03. TIỀN XỬ LÝ DỮ LIỆU VĂN BẢN		24
3.1	Mô tả dữ liệu	24
3.1.1	Giới thiệu về tập dữ liệu	24
3.1.2	Hướng dẫn cài đặt dữ liệu	25
3.1.3	Phân tích tổng quan dữ liệu	25
3.1.4	Lý do chọn tập dữ liệu này ?	25
3.2	Cơ sở tiền xử lý	26
3.3	Triển khai chi tiết	28
3.3.1	Khai phá dữ liệu	28
3.3.2	Làm sạch dữ liệu	28
3.3.3	Tokenization	31
3.3.4	Loại bỏ các Stop-Words và Non-Words	33
3.3.5	Stemming và Lemmatization	34
3.3.6	Vector hoá văn bản (Text Vectorization)	36
3.4	Phân tích & đánh giá kết quả	40
3.4.1	Phân tích kết quả định lượng	40
3.4.2	So sánh trực quan và diễn giải kết quả	41
3.4.3	Phân tích tác động về chất lượng dữ liệu	45

DANH MỤC HÌNH ẢNH

Figure 1.1 Ví dụ hình ảnh X-quang trong bộ dữ liệu: (trái) bình thường, (giữa) viêm phổi do vi khuẩn, (phải) viêm phổi do virus.	2
Figure 1.2 So sánh phân phối cường độ pixel giữa các phương pháp chuẩn hóa.	4
Figure 1.3 So sánh ảnh gốc, histogram equalization, CLAHE, và Gaussian Blur.	5
Figure 1.4 So sánh các phương pháp phát hiện biên: Original / Sobel / Prewitt / Canny.	5
Figure 2.1 Tỷ lệ phần trăm dữ liệu bị thiếu trong các cột của dataset	13
Figure 2.2 Boxplot thể hiện phân phối của latitude trước và sau khi áp dụng Min-MaxScaler	15
Figure 2.3 Boxplot thể hiện phân phối của longitude trước và sau khi áp dụng Min-MaxScaler	15
Figure 2.4 Boxplot thể hiện phân phối của population trước và sau khi áp dụng Log Transform	16
Figure 2.5 Boxplot thể hiện phân phối của dem trước và sau khi áp dụng StandardScaler	17
Figure 2.6 Số lượng giá trị unique của mỗi cột	20
Figure 2.7 Hệ số tương quan của các cột dữ liệu	21
Figure 3.1 BOW: non-zero features per docs	41
Figure 3.2 TF-IDF: non-zero features per docs	42
Figure 3.3 Word2Vec: non-features per docs	43
Figure 3.4 Word Cloud	44
Figure 3.5 Word Frequency	44

DANH MỤC BẢNG BIỂU

Table 1.1	Bảng Phân Công Việc	iii
Table 2.1	Thông tin các biến trong tập dữ liệu	8
Table 2.2	Bảng phân phối các biến số: latitude, longitude, population, dem .	14
Table 2.3	Số lượng giá trị unique của các cột trong dataset	19

THÔNG TIN NHÓM & BẢNG PHÂN CÔNG

Thông Tin Nhóm

Nhóm số 05 bao gồm 3 thành viên với các thông tin chi tiết sau:

- Lê Phước Phát - 22127322
- Trần Lý Nhật Hào - 23127187
- Huỳnh Lê Duy Khánh - 23127384

Bảng Phân Công Công Việc

Bảng 1.1 Bảng Phân Công Công Việc

MSSV	Họ và Tên	Công Việc	Mức độ
22127322	Lê Phước Phát	<p>[Phần thực thi chương trình (code)]</p> <ul style="list-style-type: none"> • Tiền xử lý dữ liệu văn bản (Twitter 15 and 16) <p>[Phần viết tài liệu báo cáo]</p> <ul style="list-style-type: none"> • Viết báo cáo phần CHƯƠNG 03: TIỀN XỬ LÝ VĂN BẢN. • Viết hướng dẫn cài đặt README.md 	100%
23127187	Trần Lý Nhật Hào	<p>[Phần thực thi chương trình (code)]</p> <ul style="list-style-type: none"> • Tiền xử lý dữ liệu hình ảnh (<i>điền tên bộ dataset vào nha ...</i>) <p>[Phần viết tài liệu báo cáo]</p> <ul style="list-style-type: none"> • Viết báo cáo phần CHƯƠNG 01. TIỀN XỬ LÝ DỮ LIỆU HÌNH ẢNH. • Viết hướng dẫn cài đặt README.md 	100%
23127384	Huỳnh Lê Duy Khánh	<p>[Phần thực thi chương trình (code)]</p> <ul style="list-style-type: none"> • Tiền xử lý dữ liệu dạng bảng <i>World Cities Population and Location</i> <p>[Phần viết tài liệu báo cáo]</p> <ul style="list-style-type: none"> • Viết báo cáo phần CHƯƠNG 02. TIỀN XỬ LÝ DỮ LIỆU DẠNG BẢNG. • Viết hướng dẫn cài đặt README.md 	100%

CHƯƠNG 01

TIỀN XỬ LÝ DỮ LIỆU HÌNH ẢNH

Trong chương này, nhóm chúng em trình bày chi tiết quy trình tiền xử lý dữ liệu hình ảnh (image data) được thực hiện trên bộ dữ liệu X-quang phổi (Chest X-Ray Images), với mục tiêu phục vụ cho các mô hình học sâu (deep learning) trong chẩn đoán bệnh viêm phổi (Pneumonia).

Chương này tập trung vào việc chuẩn bị dữ liệu ảnh đầu vào thông qua các bước xử lý tiền đề quan trọng như thay đổi kích thước ảnh (resizing), chuyển đổi không gian màu (grayscale conversion), chuẩn hóa cường độ điểm ảnh (normalization), tăng cường chất lượng ảnh (enhancement), và phát hiện biên (edge detection). Các bước này giúp đảm bảo hình ảnh có chất lượng cao, đồng nhất về kích thước và độ sáng, đồng thời làm nổi bật các đặc trưng quan trọng phục vụ cho việc huấn luyện mô hình thị giác máy tính.

Mục tiêu của chương là xây dựng một quy trình tiền xử lý ảnh thông nhất, khoa học và có khả năng tổng quát cao, đảm bảo rằng dữ liệu hình ảnh đầu vào đạt chất lượng tối ưu trước khi đưa vào các mô hình học sâu ở các chương tiếp theo. Các kết quả trung gian được trực quan hóa và phân tích định lượng, nhằm đánh giá tác động của từng kỹ thuật đối với chất lượng và độ tương phản của ảnh y tế.

1.1 Mô tả dữ liệu

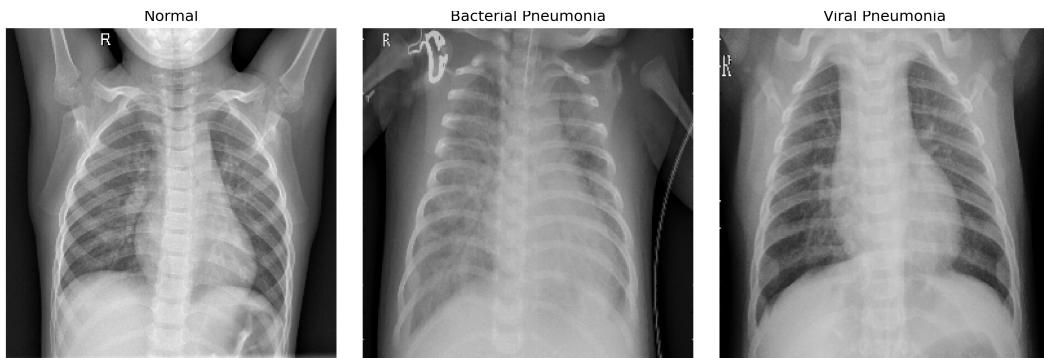
1.1.1 Giới thiệu về tập dữ liệu

Bộ dữ liệu được sử dụng là **Chest X-Ray Images (Pneumonia)** do **Guangzhou Women and Children's Medical Center** công bố và được lưu trữ tại <https://data.mendeley.com/datasets/rscbjbr9sj/2>. Bộ dữ liệu này bao gồm hình ảnh X-quang ngực (chụp theo hướng anterior-posterior) của trẻ em từ 1–5 tuổi, được chẩn đoán bởi hai chuyên gia X-quang độc lập và xác nhận thêm bởi chuyên gia thứ ba để đảm bảo tính chính xác của nhãn.

Cấu trúc dữ liệu:

- Tổ chức thành ba thư mục chính: train, test, val.
- Mỗi thư mục chứa hai thư mục con:
 - NORMAL: ảnh phổi bình thường.
 - PNEUMONIA: ảnh phổi mắc viêm phổi (có thể là do vi khuẩn hoặc virus).
- Tổng cộng 5,863 ảnh X-quang định dạng JPEG.

Ví dụ minh họa:



Hình 1.1 Ví dụ hình ảnh X-quang trong bộ dữ liệu: (trái) bình thường, (giữa) viêm phổi do vi khuẩn, (phải) viêm phổi do virus.

Trong đó, ảnh bình thường thể hiện phổi trong suốt, không có vùng mờ; viêm phổi do vi khuẩn thường tạo vùng đặc thù tập trung (focal lobar consolidation); viêm phổi do virus cho thấy các vùng mờ lan tỏa hai bên phổi.

Nguồn dữ liệu: Kermany et al., 2018, “Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning,” *Cell*. [http://www.cell.com/cell/fulltext/S0092-8674\(18\)30154-5](http://www.cell.com/cell/fulltext/S0092-8674(18)30154-5)

Giấy phép: CC BY 4.0.

với hai thư mục con: NORMAL/ và PNEUMONIA/. Trong quá trình xử lý, thư viện OpenCV, NumPy, Matplotlib, và Seaborn được sử dụng cho việc đọc, hiển thị, và trực quan hóa hình ảnh.

1.1.2 Phân tích tổng quan dữ liệu

Thống kê cơ bản:

- NORMAL: 1,341 ảnh.
- PNEUMONIA: 4,822 ảnh.

Các ảnh có kích thước khác nhau (thường trong khoảng 1000×1000 px), mức độ sáng và tương phản khác biệt. Do đó, việc chuẩn hóa kích thước và cường độ pixel là cần thiết trước khi đưa vào mô hình học sâu.

Thống kê định lượng mẫu:

NORMAL: mean = 121.12, std = 62.98; PNEUMONIA: mean = 124.05, std = 60.38.

Sự khác biệt nhẹ giữa hai nhóm phản ánh đặc điểm vùng mờ (opacity) trong ảnh viêm phổi, thường có xu hướng sáng hơn.

1.1.3 Lý do chọn tập dữ liệu

Bộ dữ liệu này là một trong những bộ dữ liệu X-quang y tế phổ biến nhất, cung cấp dữ liệu thật từ bệnh nhân và được gán nhãn bởi bác sĩ chuyên môn. Nó phù hợp cho các nghiên cứu về:

- Phát hiện và phân loại viêm phổi.

- Tiền xử lý ảnh y tế với độ nhiễu và độ sáng không đồng nhất.
- Ứng dụng học sâu (CNN) trong chẩn đoán hình ảnh.

Ngoài ra, dữ liệu có kích thước vừa phải, thuận tiện cho việc thử nghiệm các pipeline xử lý ảnh và mô hình huấn luyện ban đầu.

1.2 Cơ sở tiền xử lý

Các kỹ thuật được áp dụng:

- Thay đổi kích thước (Resizing)
- Chuyển sang thang xám (Grayscale)
- Chuẩn hóa giá trị pixel (Normalization)
- Cân bằng histogram (Histogram Equalization, CLAHE)
- Làm mượt ảnh (Gaussian Blur)
- Phát hiện biên (Sobel, Prewitt, Canny)

Mỗi kỹ thuật trên nhằm cải thiện chất lượng hình ảnh đầu vào, giảm nhiễu, tăng độ tương phản và làm nổi bật các đặc trưng quan trọng trong cấu trúc phổi.

1.3 Triển khai chi tiết

1.3.1 Loading & Resizing (Requirement a)

Ảnh được đọc từ thư mục và resize về kích thước chuẩn 128×128 . Mục đích của việc này là chuẩn hóa kích thước đầu vào, đảm bảo tất cả hình ảnh có cùng độ phân giải để mô hình CNN có thể xử lý đồng nhất.

Giải thích lựa chọn:

- **Kích thước 128x128:** cân bằng giữa chi tiết và tốc độ tính toán.
- **Hiệu quả:** giảm chi phí lưu trữ và thời gian huấn luyện khoảng 10 lần so với ảnh gốc.
- **Rủi ro:** mất chi tiết nhỏ, song với ảnh X-quang, thông tin tổn thương thường có kích thước vùng lớn nên ảnh hưởng không đáng kể.

1.3.2 Grayscale Conversion (Requirement b)

Ảnh RGB được chuyển sang grayscale để giảm chiều dữ liệu và tập trung vào độ sáng (intensity). Trong ảnh X-quang, màu sắc không mang thông tin, chỉ cần giữ lại thông tin cường độ sáng.

Công thức chuyển đổi:

$$I_{gray} = 0.299R + 0.587G + 0.114B$$

Ưu điểm:

- Giảm 2/3 dung lượng dữ liệu.
- Tập trung vào tương phản, giúp dễ phân biệt vùng mờ bệnh lý.

1.3.3 Normalization (Requirement c)

Để loại bỏ sự khác biệt về độ sáng, hai phương pháp được áp dụng:

(1) Min–Max Normalization:

$$I' = \frac{I - I_{min}}{I_{max} - I_{min}}$$

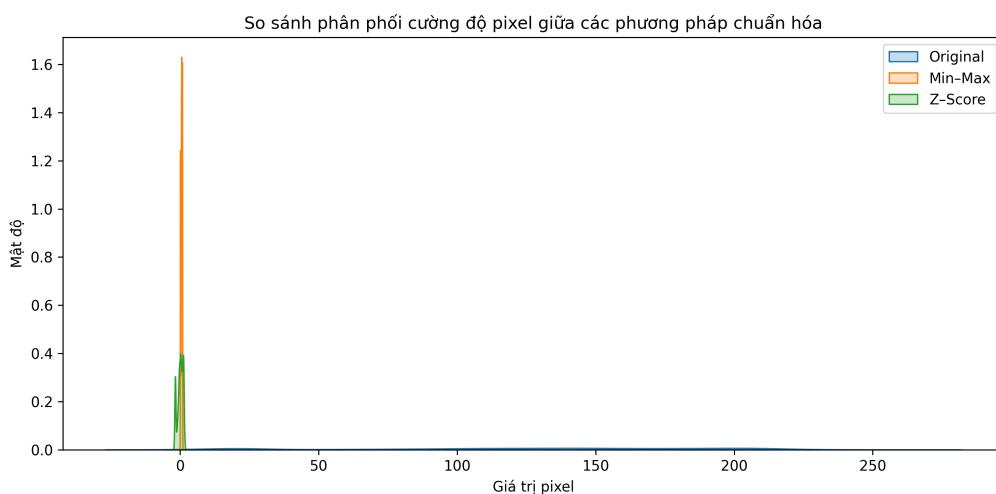
Chuyển giá trị pixel về $[0,1]$, giữ nguyên tương quan giữa các điểm ảnh.

(2) Z–Score Normalization:

$$I' = \frac{I - \mu}{\sigma}$$

Đưa dữ liệu về trung bình 0 và độ lệch chuẩn 1, phù hợp cho các mô hình học sâu.

Phân tích trực quan:



Hình 1.2 So sánh phân phối cường độ pixel giữa các phương pháp chuẩn hóa.

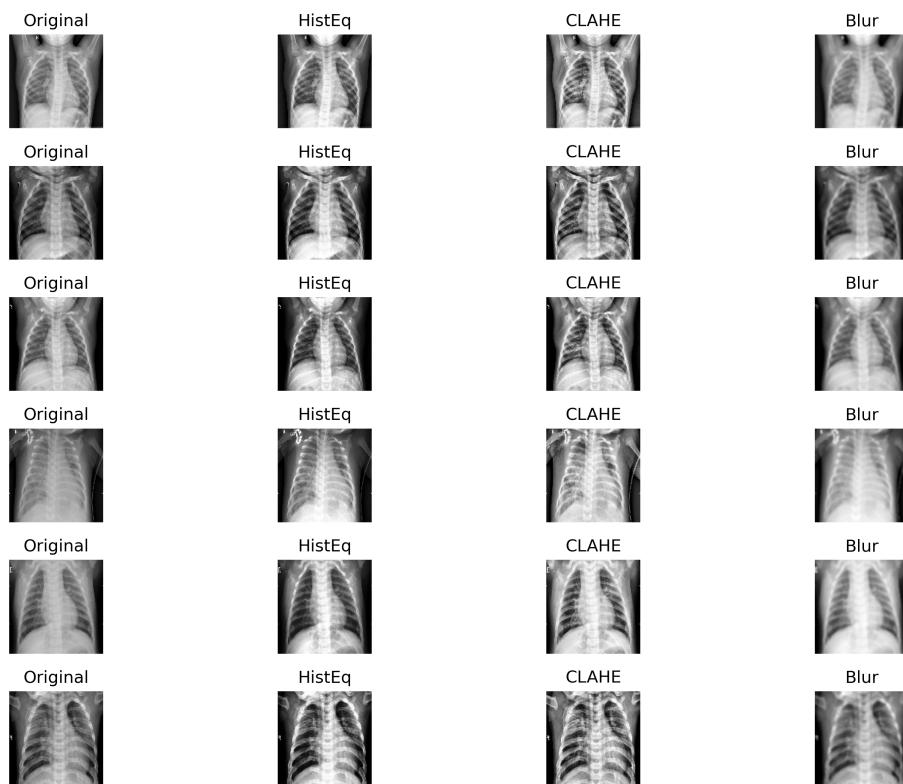
Nhận xét: Phương pháp Min–Max giúp dữ liệu nằm trong phạm vi ổn định cho CNN; Z–Score giúp cân bằng cường độ sáng, hữu ích trong học sâu y tế.

1.3.4 Histogram Equalization, CLAHE và Gaussian Blur

Histogram Equalization: Cân bằng độ sáng bằng cách phân phối lại histogram, giúp tăng độ tương phản tổng thể.

CLAHE (Contrast Limited Adaptive Histogram Equalization): Cải thiện chi tiết vùng nhỏ mà không gây nhiễu quá mức.

Gaussian Blur: Làm mượt ảnh, loại bỏ nhiễu ngẫu nhiên và giảm gradient mạnh.



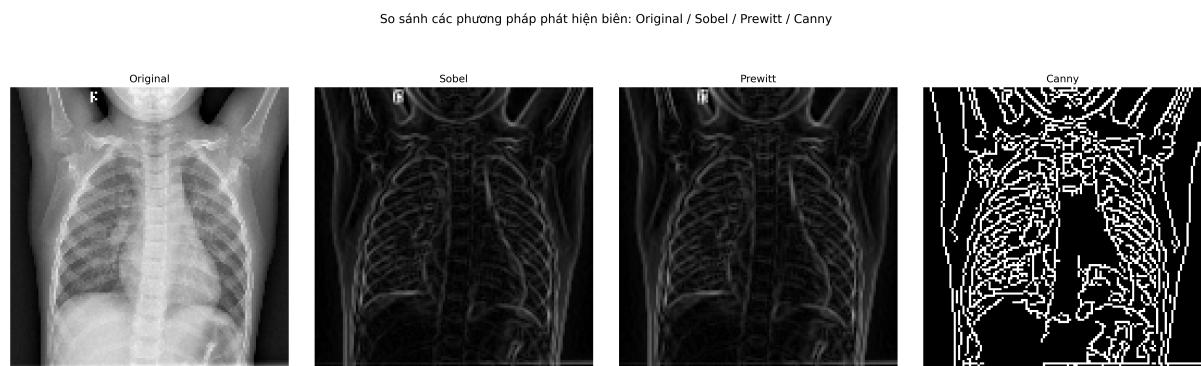
Hình 1.3 So sánh ảnh gốc, histogram equalization, CLAHE, và Gaussian Blur.

Nhận xét: CLAHE cho kết quả nổi bật nhất, tăng rõ chi tiết mô phổi mà vẫn giữ cấu trúc tự nhiên, phù hợp cho trích xuất đặc trưng.

1.3.5 Edge Detection (Sobel, Prewitt, Canny)

Ba phương pháp được áp dụng để phát hiện biên cấu trúc phổi:

- **Sobel:** Dựa trên đạo hàm bậc nhất, nhạy với thay đổi cường độ.
- **Prewitt:** Giống Sobel nhưng đơn giản hơn, ít nhạy với nhiễu.
- **Canny:** Phát hiện biên chính xác hơn, có bước lọc Gaussian và hysteresis threshold.



Hình 1.4 So sánh các phương pháp phát hiện biên: Original / Sobel / Prewitt / Canny.

Nhận xét:

- Sobel làm nổi bật cấu trúc chính của lồng ngực.
- Prewitt cho biên mềm hơn, ít nhiễu.
- Canny cho biên rõ và sắc nét nhất, phù hợp dùng để xác định vùng ROI (Region of Interest).

1.4 Phân tích & đánh giá kết quả

1.4.1 Phân tích định lượng

Trung bình cường độ sáng và độ lệch chuẩn giữa hai lớp (Normal/Pneumonia) cho thấy sự khác biệt rõ:

NORMAL: $\mu = 121.1$, $\sigma = 62.97$; PNEUMONIA: $\mu = 124.0$, $\sigma = 60.38$

Điều này phản ánh sự hiện diện của vùng mờ (consolidation) trong ảnh bệnh lý.

1.4.2 So sánh trực quan và diễn giải kết quả

Sau các bước tiền xử lý:

- Ảnh được chuẩn hóa về kích thước và cường độ sáng.
- CLAHE tăng chi tiết vùng mô phổi, đặc biệt trong ảnh viêm phổi.
- Gaussian Blur loại bỏ nhiễu không mong muốn.
- Phát hiện biên giúp xác định vùng phổi, hỗ trợ trích xuất đặc trưng.

1.4.3 Tác động đến chất lượng dữ liệu

- Giảm nhiễu, tăng độ tương phản, cân bằng ánh sáng.
- Làm nổi bật biên cấu trúc giúp mô hình phát hiện vùng bất thường dễ dàng hơn.
- Giảm phương sai không mong muốn giữa các mẫu dữ liệu, giúp mô hình hội tụ nhanh hơn.

1.5 Kết luận

Sau quá trình tiền xử lý, tập dữ liệu X-quang đã đạt được các tiêu chí:

- Ảnh có kích thước đồng nhất (128x128) và cường độ chuẩn hóa.
- Các kỹ thuật như CLAHE và Gaussian Blur cải thiện đáng kể chất lượng hiển thị và khả năng phân biệt giữa vùng bệnh và bình thường.
- Phát hiện biên (Canny, Sobel, Prewitt) giúp xác định vùng ROI hiệu quả, hỗ trợ các mô hình CNN trong giai đoạn huấn luyện.

Kết quả tiền xử lý không chỉ nâng cao chất lượng dữ liệu mà còn giảm sai lệch do điều kiện chụp, tạo nền tảng vững chắc cho bước trích xuất đặc trưng và huấn luyện mô hình ở các chương tiếp theo.

CHƯƠNG 02

TIỀN XỬ LÝ DỮ LIỆU DẠNG BẢNG

Trong chương này, nhóm chúng em trình bày chi tiết quy trình tiền xử lý dữ liệu dạng bảng (tabular data) được thực hiện trước khi tiến hành huấn luyện mô hình học máy. Chương này sẽ tập trung vào việc làm sạch, chuẩn hoá và biến đổi dữ liệu nhằm đảm bảo dữ liệu đầu vào có chất lượng cao, nhất quán.

Cụ thể, nhóm tiến hành chuẩn hoá kiểu dữ liệu, xử lý giá trị khuyết (missing values), cũng như mã hoá các biến phân loại để phù hợp với yêu cầu của mô hình. Ngoài ra, nhóm còn áp dụng các kỹ thuật lựa chọn đặc trưng (feature selection) nhằm loại bỏ các thuộc tính dư thừa hoặc nhiễu, giúp mô hình học hiệu quả hơn.

Mục tiêu của chương là xây dựng một bộ dữ liệu sạch, đồng nhất và có phân phối hợp lý, tạo nền tảng vững chắc cho các bước phân tích và huấn luyện mô hình học máy ở các chương tiếp theo.

2.1 Mô tả dữ liệu

2.1.1 Giới thiệu về tập dữ liệu

Tập dữ liệu được sử dụng là dữ liệu dạng bảng gồm các cột mô tả đặc trưng của các khu vực địa lý, bao gồm:

Bảng 2.1 Thông tin các biến trong tập dữ liệu

Tên biến	Vai trò	Kiểu dữ liệu	Nhóm thông tin	Mô tả	Đơn vị	Thiếu dữ liệu
geonameid	Feature	String	Định danh	Mã định danh duy nhất cho mỗi thành phố trong cơ sở dữ liệu GeoNames		yes
name	Feature	String	Địa danh	Tên chính thức của thành phố		no
asciiname	Feature	String	Địa danh	Tên thành phố được chuyển sang dạng ASCII		no
alternatenames	Feature	String	Địa danh	Danh sách các tên khác hoặc tên thay thế của thành phố		yes
latitude	Feature	Float	Vị trí địa lý	Vĩ độ của thành phố	Độ (°)	yes
longitude	Feature	Float	Vị trí địa lý	Kinh độ của thành phố	Độ (°)	yes
feature class	Feature	Categorical	Phân loại địa lý	Nhóm đặc trưng của thực thể địa lý		yes
feature code	Feature	Categorical	Phân loại địa lý	Mã chi tiết hơn của loại địa lý		yes
country code	Feature	Categorical	Quốc gia	Mã quốc gia theo chuẩn ISO		yes
cc2	Feature	Categorical	Quốc gia phụ	Mã quốc gia phụ, dùng khi một địa danh thuộc nhiều quốc gia		yes
admin1 code	Feature	Categorical	Hành chính	Mã cấp hành chính 1		yes
admin2 code	Feature	Categorical	Hành chính	Mã cấp hành chính 2		yes
admin3 code	Feature	Categorical	Hành chính	Mã cấp hành chính 3		yes
admin4 code	Feature	Categorical	Hành chính	Mã cấp hành chính 4		yes
population	Feature	Integer	Dân số	Dân số ước tính của thành phố	Người	yes
elevation	Feature	Integer	Địa lý	Độ cao của thành phố so với mực nước biển	Mét	yes
dem	Feature	Integer	Địa hình	Độ cao trung bình từ mô hình số địa hình	Mét	yes
timezone	Feature	Categorical	Thời gian	Múi giờ của thành phố		yes
modification date	Feature	Date	Khác	Ngày cập nhật cuối cùng của bản ghi trong cơ sở dữ liệu GeoNames	YYYY-MM-DD	yes

Kích thước của dataset này là

231,269 × 19.

Trong quá trình đọc dữ liệu bằng `read_csv`, cần sử dụng mã hóa latin1 vì dataset chứa tên

của nhiều thành phố khác nhau trên thế giới.

2.1.2 *Hướng dẫn cài đặt dữ liệu*

Dữ liệu được lưu dưới dạng .csv. Các thư viện được sử dụng trong quá trình xử lý bao gồm:

- **pandas:** Xử lý dữ liệu dạng bảng (DataFrame), đọc/ghi CSV, lọc, nhóm và xử lý giá trị thiếu.
- **numpy:** Thư viện toán học, xử lý mảng, phép toán vector hóa và thống kê cơ bản.
- **matplotlib.pyplot:** Vẽ đồ thị cơ bản như: line, scatter, bar, histogram v.v. `plt.style.use('ggplot')` tạo kiểu đồ thị đẹp.
- **seaborn:** Vẽ đồ thị nâng cao dựa trên matplotlib, dễ trực quan hóa mối quan hệ và phân phối dữ liệu.
- **sklearn.preprocessing:**
 - **MinMaxScaler:** Chuẩn hóa dữ liệu về khoảng [0,1].
 - **StandardScaler:** Chuẩn hóa dữ liệu về phân phối chuẩn (mean=0, std=1).
 - **OneHotEncoder:** Biến dữ liệu phân loại thành vector nhị phân.
 - **LabelEncoder:** Mã hóa nhãn (target) thành số nguyên.

2.1.3 *Phân tích tổng quan dữ liệu*

Số lượng và loại biến

- Tổng cộng 19 biến.
- Loại biến:
 - **Integer/Float (định lượng):** latitude, longitude, population, elevation, dem.
 - **Categorical (định tính):** feature class, feature code, country code, cc2, admin1 code, admin2 code, admin3 code, admin4 code, timezone.
 - **String:** geonameid, name, asciname, alternatenames.
 - **Date:** modification date.
- Nhận xét: dữ liệu hỗn hợp → cần chuẩn hóa số liệu và mã hóa biến phân loại trước khi dùng mô hình.

Các nhóm thông tin

- **Định danh:** geonameid
- **Địa danh:** name, asciname, alternatenames

- **Vị trí địa lý:** latitude, longitude
- **Phân loại địa lý:** feature class, feature code
- **Quốc gia:** country code, cc2
- **Hành chính:** admin1 code đến admin4 code
- **Dân số:** population
- **Địa hình:** elevation, dem
- **Thời gian:** timezone
- **Khác:** modification date

Giá trị thiếu

- Nhiều biến có giá trị thiếu, ngoại trừ name và asciiname.
- Các biến quan trọng có khả năng thiếu: geonameid, latitude, longitude, population, elevation, dem → cần xử lý missing trước khi mô hình hóa.
- Biến danh mục như feature class, feature code, country code, admin1 code đến admin4 code cũng có missing → cần điền hoặc mã hóa phù hợp.

Tóm tắt tổng quan

- Dữ liệu dạng bảng hỗn hợp: số, phân loại, chuỗi, ngày tháng.
- Nhiều giá trị thiếu → cần xử lý trước khi dùng mô hình học máy.
- Ứng dụng tiềm năng: phân tích địa lý, dự đoán dân số, clustering khu vực, trực quan hóa bản đồ.
- Tiền xử lý cần thiết:
 - Điền missing values.
 - Chuẩn hóa số liệu (StandardScaler / MinMaxScaler).
 - Mã hóa biến phân loại (OneHotEncoder / LabelEncoder).
 - Biến đổi log cho các biến lệch (population).

2.1.4 Lý do chọn tập dữ liệu này

Tập dữ liệu này là một ví dụ điển hình cho dạng dữ liệu thực tế thường gặp trong các bài toán học máy. Nó bao gồm cả thuộc tính số và thuộc tính phân loại (categorical), đồng thời chứa nhiều vấn đề phổ biến cần được xử lý trước khi đưa vào mô hình, chẳng hạn như giá trị ngoại lai (outlier), phân phối lệch, và sự chênh lệch tỷ lệ giữa các thuộc tính.

2.2 Cơ sở tiền xử lý

Các kỹ thuật tiền xử lý dữ liệu được áp dụng

- **Xử lý giá trị thiếu (Missing Values Handling):**

- **Drop cột có tỷ lệ missing cao:**

- * Loại bỏ các cột quá nhiều giá trị thiếu để tránh làm sai lệch mô hình.
 - * *Điểm mạnh:* Đơn giản, nhanh chóng, loại bỏ dữ liệu kém chất lượng.
 - * *Điểm yếu:* Có thể mất thông tin quan trọng nếu cột chứa feature hữu ích.
 - * *Khi sử dụng:* Khi cột có tỷ lệ missing quá cao và không quan trọng.

- **Mode cho biến định tính (String):**

- * Thay giá trị missing bằng giá trị xuất hiện nhiều nhất.
 - * *Điểm mạnh:* Giữ lại tất cả các dòng, đơn giản.
 - * *Điểm yếu:* Có thể gây tập trung quá mức vào giá trị phổ biến, làm mất đa dạng dữ liệu.

- **Median cho biến định lượng (Float/Integer):**

- * Thay giá trị missing bằng trung vị.
 - * *Điểm mạnh:* Ít bị ảnh hưởng bởi outlier so với mean.
 - * *Điểm yếu:* Không phản ánh biến động dữ liệu chi tiết.

- **Chuẩn hóa và biến đổi dữ liệu (Normalization & Transformation):**

- **MinMaxScaler:**

- * Chuyển dữ liệu về khoảng [0,1] theo công thức:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

- * *Điểm mạnh:* Giữ tỷ lệ quan hệ giữa các giá trị, phù hợp cho thuật toán nhạy với scale (ví dụ KNN, neural network).
 - * *Điểm yếu:* Bị ảnh hưởng mạnh bởi outlier.

- **Log Transform:**

- * Biến đổi dữ liệu để giảm độ lệch phải:

$$x' = \log(x + 1)$$

- * *Điểm mạnh:* Giảm độ lệch, giảm ảnh hưởng outlier.
 - * *Điểm yếu:* Chỉ áp dụng với giá trị $x \geq 0$.

- **StandardScaler:**

- * Chuẩn hóa dữ liệu về phân phối chuẩn (mean=0, std=1) theo công thức:

$$x' = \frac{x - \mu}{\sigma}$$

trong đó μ là trung bình, σ là độ lệch chuẩn.

- * *Điểm mạnh:* Hiệu quả cho các thuật toán dựa trên khoảng cách và gradient (ví dụ: KNN, logistic regression, neural network).
- * *Điểm yếu:* Bị ảnh hưởng bởi outlier.

- **Mã hóa biến phân loại (Encoding):**

- **One-Hot Encoding:**

- * Chuyển biến phân loại thành vector nhị phân.
 - * *Điểm mạnh:* Giữ nguyên thông tin, phù hợp cho hầu hết mô hình.
 - * *Điểm yếu:* Tăng số chiều dữ liệu, không hiệu quả với nhiều category.

- **Label Encoding:**

- * Chuyển nhãn hoặc biến phân loại thành số nguyên.
 - * *Điểm mạnh:* Gọn nhẹ, không tăng chiều dữ liệu.
 - * *Điểm yếu:* Không phù hợp với biến phân loại không có thứ tự (introduces ordinal relationship).

- **Lựa chọn đặc trưng (Feature Selection):**

- **Drop cột unique:** Loại bỏ các cột có giá trị duy nhất cho mỗi dòng (không mang thông tin phân loại).
- **Quan sát mối tương quan:** Xác định các cột có mối tương quan cao với nhau để tránh multicollinearity.
- *Điểm mạnh:* Giảm số lượng biến, giảm overfitting, tăng hiệu suất mô hình.
- *Điểm yếu:* Có thể bỏ sót một số thông tin quan trọng nếu đánh giá chưa chính xác.
- *Khi sử dụng:* Khi muốn giảm chiều dữ liệu hoặc loại bỏ biến dư thừa, đặc biệt với dữ liệu bảng lớn.

2.3 Triển khai chi tiết

1. Đọc dữ liệu

Dữ liệu này được đọc vào Python bằng hàm `read_csv` của thư viện pandas. Vì dataset chứa tên của nhiều thành phố khác nhau trên thế giới, nên cần sử dụng mã hóa latin1 để tránh lỗi khi đọc các ký tự đặc biệt.

```
df = pd.read_csv(path, encoding="latin1")
```

2. Khám phá dữ liệu

Khám phá dữ liệu giúp chúng ta hiểu rõ hơn về dữ liệu và thuận tiện hơn trong việc thực hiện các bước tiền xử lý sau này.

Bằng cách sử dụng các hàm có sẵn trong thư viện pandas như `head()`, `tail()`, `shape()`, `dtypes()`, chúng ta có thể nhanh chóng nắm bắt được cấu trúc và đặc điểm của dataset.

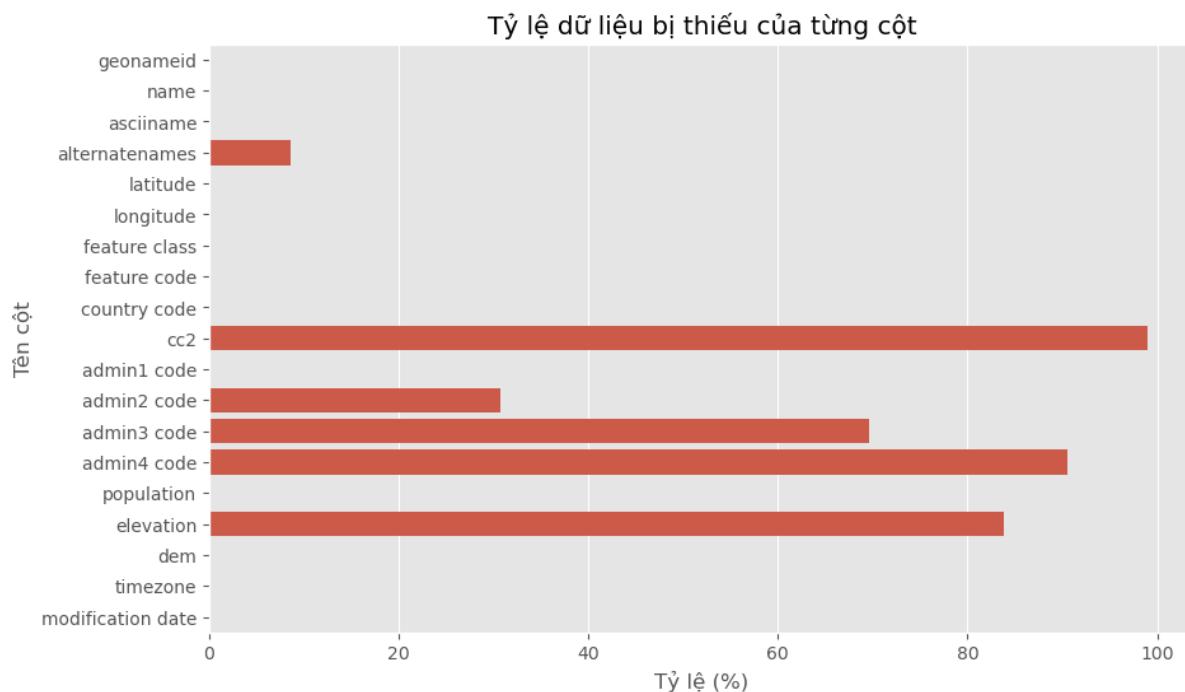
3. Kiểm tra và thống kê dữ liệu thiếu

Đầu tiên, chúng ta kiểm tra số lượng giá trị thiếu của từng cột và tỷ lệ phần trăm dữ liệu bị thiếu:

- Sử dụng `df.isnull().sum()` để đếm số giá trị missing.
- Tính tỷ lệ phần trăm:

$$\text{Missing \%} = \frac{\text{Số giá trị missing}}{\text{Tổng số dòng}} \times 100$$

- Vẽ biểu đồ trực quan bằng seaborn để quan sát tỷ lệ missing.



Hình 2.1 Tỷ lệ phần trăm dữ liệu bị thiếu trong các cột của dataset

Có thể thấy rằng tỷ lệ dữ liệu bị thiếu có thể chia thành ba nhóm:

- **Nhóm 1:** `admin3 code`, `admin4 code`, `admin2 code`, `elevation`, `cc2` — các cột này có tỷ lệ dữ liệu bị thiếu rất lớn (trên 30%).
- **Nhóm 2:** `alternatenames` — cột này có tỷ lệ dữ liệu bị thiếu ở mức vừa phải (khoảng 8%).

- **Nhóm 3:** country code, admin1 code, population, longitude, latitude, timezone, dem, modification date, feature class, feature code, geonameid — các cột này có tỷ lệ dữ liệu bị thiếu rất nhỏ (dưới 0.1%).

Các bước xử lý dữ liệu thiếu được thực hiện như sau:

- **Loại bỏ các cột có hơn 30% missing:** loại bỏ các cột ảnh hưởng mạnh đến mô hình do thiếu dữ liệu quá nhiều.
- **Loại bỏ các cột không cần thiết:**
 - alternatenames: Cột này chứa danh sách tên khác của một thành phố và gần như không có ảnh hưởng gì khi được sử dụng trong các mô hình Machine Learning.
 - modification date: Xoá cột này vì trong bài tập này không cần sử lý dữ liệu dạng thời gian như cột này.
- **Điền giá trị missing cho các biến định tính (string):** Sử dụng giá trị xuất hiện nhiều nhất (*mode*) cho các cột: admin1 code, feature class, feature code, country code, timezone.
- **Điền giá trị missing cho các biến số (float/int):** Sử dụng giá trị trung vị (*median*) cho các cột: latitude, longitude, population, dem.
- **Xóa các dòng còn missing ở cột quan trọng:** geonameid.

4. Chuẩn hóa và biến đổi dữ liệu số

Bảng 2.2 Bảng phân phối các biến số: latitude, longitude, population, dem

	latitude	longitude	population	dem
count	23,468	23,468	23,468	23,468
mean	27.771365	15.482522	115,649.0	292.932
std	22.924456	71.017615	481,382.4	553.311
min	-54.800000	-176.174530	0	-9,999
25%	14.970010	-45.416393	22,018	31
50%	34.366500	15.056190	35,748	125
75%	44.632280	75.029775	75,224	339
max	78.223340	179.364510	22,315,470	5,022

Bảng phân phối cho thấy latitude và longitude nằm trong phạm vi thực tế của vĩ độ và kinh độ, phân phối tương đối đồng đều; population lệch phai mạnh với một số giá trị cực lớn, cần áp dụng Log Transform; dem có giá trị phân bố rộng quanh trung bình, thích hợp để chuẩn hóa bằng Standard Scaler.

Min–Max Scaling

Công thức:

$$X_{\text{scaled}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}$$

Đối tượng áp dụng:

- Các biến có phạm vi giá trị rộng, có giá trị lớn nhất và nhỏ nhất rõ ràng.
- Thích hợp khi cần đưa dữ liệu về một khoảng cố định, ví dụ từ 0 đến 1.

Ưu điểm:

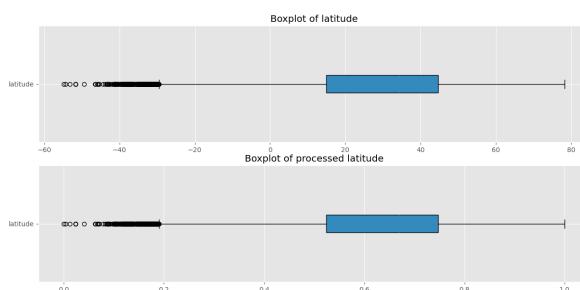
- Chuẩn hóa dữ liệu về một khoảng xác định.
- Giữ nguyên hình dạng phân phối gốc, phù hợp khi dữ liệu nằm trong phạm vi xác định và không có giá trị ngoại lai lớn.

Nhược điểm:

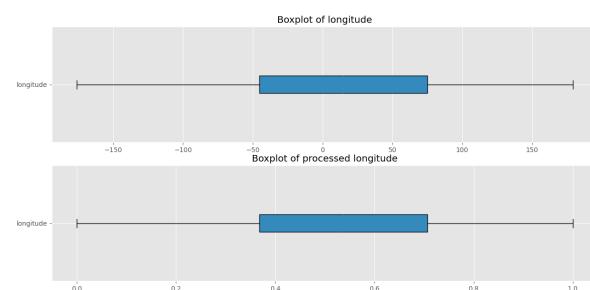
- Nhạy cảm với giá trị ngoại lai: khi có các giá trị quá lớn hoặc quá nhỏ, các giá trị khác có thể bị nén lại, giảm khả năng phân biệt.

Áp dụng cho latitude và longitude:

- **Giá trị có giới hạn:** latitude nằm trong $[-90, 90]$, longitude nằm trong $[-180, 180]$.
- **Phân phối tương đồng đều:** dữ liệu không bị lệch mạnh, có giãn tuyến tính vẫn giữ nguyên mối quan hệ không gian.



Hình 2.2 Boxplot thể hiện phân phối của latitude trước và sau khi áp dụng MinMaxScaler



Hình 2.3 Boxplot thể hiện phân phối của longitude trước và sau khi áp dụng MinMaxScaler

Có thể thấy rằng sau khi áp dụng MinMaxScaler, phân phối của dữ liệu latitude và longitude vẫn giữ nguyên hình dạng, nhưng các giá trị đã được giản về khoảng $[0, 1]$.

Log Transform**Công thức:**

$$X_{\text{scaled}} = \log(1 + X)$$

Đối tượng áp dụng:

- Dữ liệu lệch phải (right-skewed), có giá trị lớn (outlier), như dân số.

Ưu điểm:

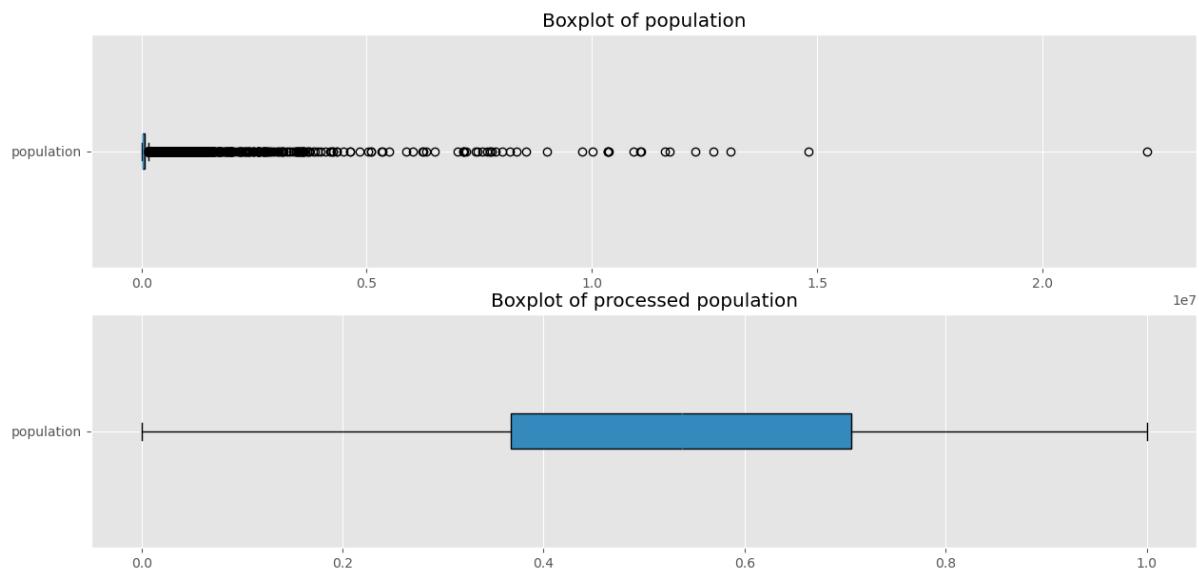
- Giảm độ lệch của phân phối, làm dữ liệu gần phân phối chuẩn hơn.
- Giúp các mô hình tuyến tính hoặc dựa trên khoảng cách hoạt động ổn định hơn.

Nhược điểm:

- Chỉ áp dụng cho dữ liệu dương (≥ 0).
- Mất tính diễn giải trực tiếp của dữ liệu gốc.

Lý do áp dụng cho population:

- Phân phối lệch phải mạnh: một vài thành phố có dân số cực lớn, trong khi nhiều thành phố chỉ có vài nghìn người.
- Giảm ảnh hưởng của giá trị outlier.
- Giúp phân phối gần chuẩn hơn, phù hợp cho thuật toán học máy.
- Giữ được thứ tự tương đối giữa các giá trị.



Hình 2.4 Boxplot thể hiện phân phối của population trước và sau khi áp dụng Log Transform

Có thể thấy rằng dữ liệu population trước khi áp dụng Log Transform bị lệch phải mạnh, nhưng sau khi biến đổi, hầu hết các giá trị nằm trong khoảng từ phân vị, giúp việc sử dụng population trong các mô hình Machine Learning trở nên dễ dàng và ổn định hơn.

Standard Scaler

Công thức:

$$X_{\text{scaled}} = \frac{X - \mu}{\sigma}$$

Đối tượng áp dụng:

- Dữ liệu có phân phối gần chuẩn hoặc khi mô hình nhạy với độ lớn giá trị (Linear Regression, SVM, PCA).

Ưu điểm:

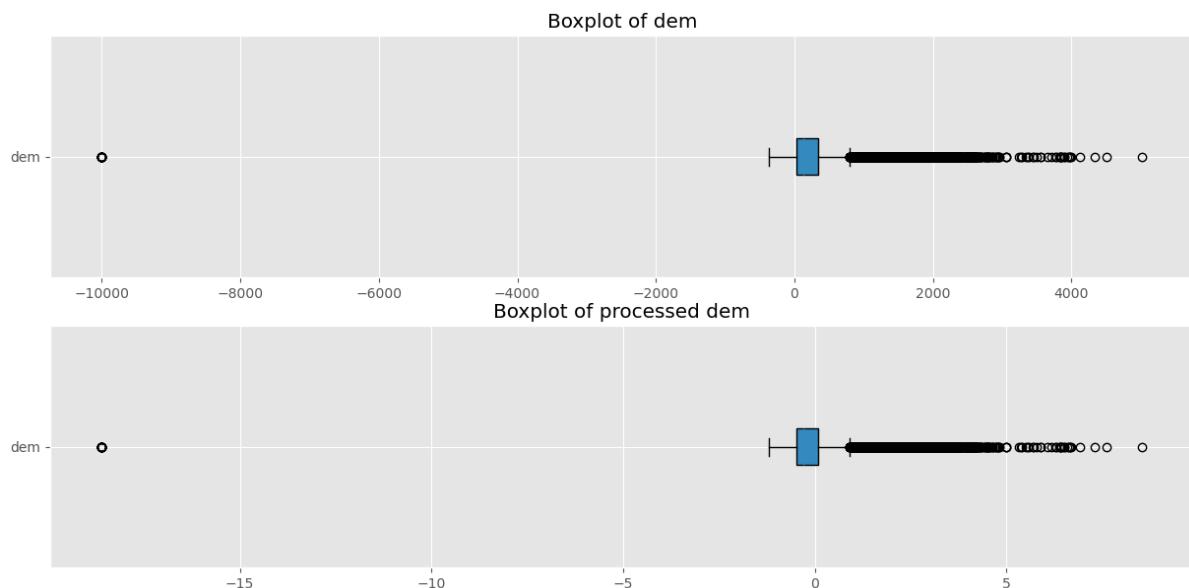
- Đưa dữ liệu về trung tâm 0 và độ lệch chuẩn 1 → giúp mô hình hội tụ nhanh hơn.
- Giảm ảnh hưởng của thang đo khác nhau giữa các đặc trưng.

Nhược điểm:

- Nhạy cảm với outlier vì mean và std bị ảnh hưởng mạnh.

Lý do áp dụng cho dem:

- dem có cả giá trị âm và dương → Standard Scaler xử lý tốt.
- Đưa dữ liệu về trung tâm 0 và độ lệch chuẩn 1 → giúp mô hình học ổn định hơn.
- Phân bố độ cao quanh giá trị trung bình, không cần giới hạn trong [0, 1].



Hình 2.5 Boxplot thể hiện phân phối của dem trước và sau khi áp dụng StandardScaler

Mặc dù vẫn còn một số giá trị outlier, sau khi chuẩn hóa bằng StandardScaler, khoảng giá trị của dem đã giảm từ $[-9999, 5022]$ xuống khoảng $[-20, 10]$. Điều này giúp dữ liệu dem trở nên ổn định hơn, giảm ảnh hưởng của các giá trị cực đoan khi huấn luyện mô hình Machine

Learning, đặc biệt với các thuật toán nhạy cảm với thang đo như Linear Regression, SVM hay KNN.

5. Xử lý biến phân loại (Encoding)

Các kỹ thuật Encoding phổ biến cho dữ liệu

• One-Hot Encoding

- Biến mỗi giá trị danh mục (nominal) thành một cột nhị phân riêng biệt, trong đó giá trị hiện diện được gán 1, các giá trị khác là 0.
- Thích hợp cho các biến **không có thứ bậc** hoặc quan hệ giữa các giá trị không có ý nghĩa số học, ví dụ: màu sắc, quốc gia, loại sản phẩm.
- Ưu điểm: tránh tạo ra thứ tự giả, giúp mô hình học các quan hệ giữa các giá trị một cách chính xác.
- Nhược điểm: làm tăng số chiều dữ liệu (curse of dimensionality) nếu biến có nhiều giá trị duy nhất.

• Ordinal Encoding

- Gán mỗi giá trị hạng mục một số nguyên theo thứ tự, phản ánh mức độ hoặc thứ bậc của biến.
- Thích hợp cho các biến **có thứ bậc**, ví dụ: trình độ học vấn (tiểu học, trung học, đại học), mức độ hài lòng (thấp, trung bình, cao).
- Ưu điểm: giữ được thông tin thứ bậc và giảm số chiều dữ liệu.
- Nhược điểm: mô hình có thể hiểu nhầm khoảng cách giữa các mức là tương đương, trong khi thực tế có thể không đều nhau.

• Label Encoding

- Gán mỗi giá trị hạng mục một số nguyên duy nhất mà không quan tâm đến thứ bậc.
- Thích hợp khi muốn **giảm số chiều dữ liệu** hoặc khi mô hình có thể xử lý dữ liệu dạng số trực tiếp (ví dụ: cây quyết định, random forest).
- Ưu điểm: đơn giản, tiết kiệm bộ nhớ.
- Nhược điểm: có thể gây hiểu nhầm thứ bậc nếu áp dụng cho biến không có thứ bậc, đặc biệt với các mô hình tuyến tính nhạy với thứ tự giá trị.

Quyết định phương pháp Encoding dựa trên số lượng giá trị unique

Số lượng giá trị unique của các cột trong dataset:

Bảng 2.3 Số lượng giá trị unique của các cột trong dataset

Cột dữ liệu	Số giá trị unique
geonameid	23,468
name	22,309
asciiname	22,247
feature class	1
feature code	16
country code	244
admin1 code	339
timezone	357

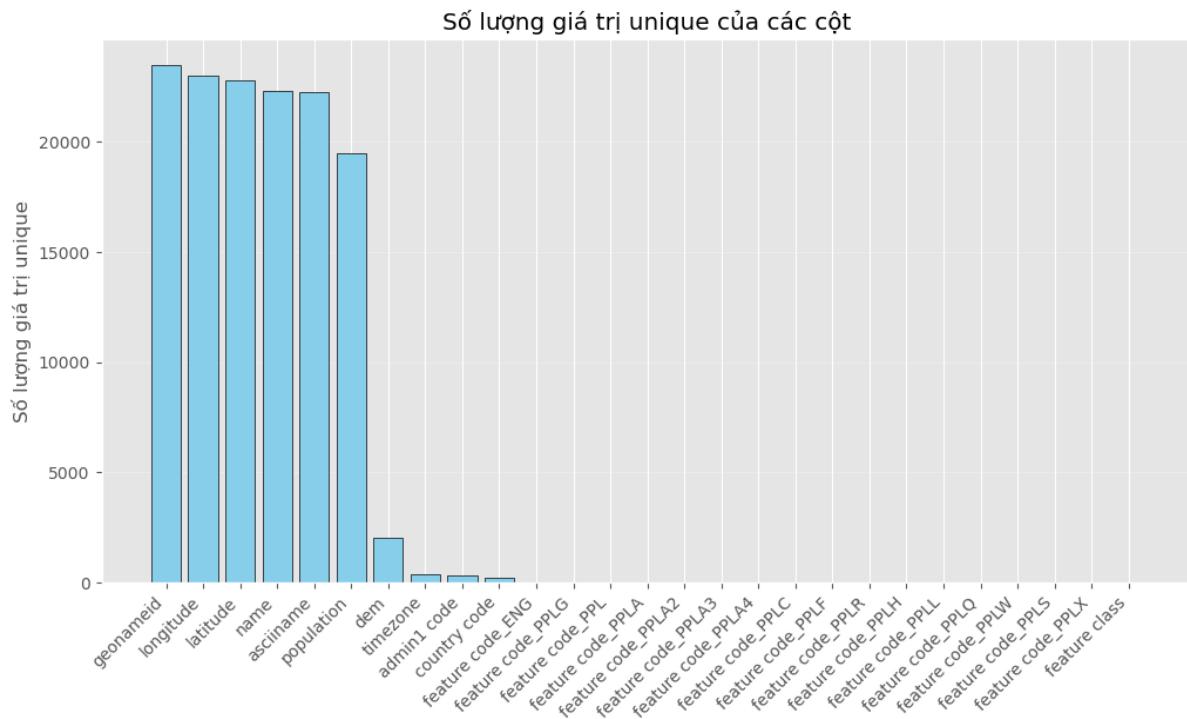
Nhận xét và phương pháp xử lý:

- geonameid, name, asciiname, alternatenames
→ Đây là ID hoặc tên riêng duy nhất, không mang thông tin phân loại → **không cần áp dụng encoding**.
- feature class
→ Chỉ có 1 giá trị, không mang thông tin phân biệt → **loại bỏ** cột này.
- feature code
→ Có số lượng giá trị ít và là danh mục → **One-Hot Encoding**.
- country code
→ Mỗi quốc gia là một danh mục riêng nhưng có nhiều giá trị → **Label Encoding** để giảm số chiều dữ liệu.
- admin1 code
→ Chứa mã cấp bang hoặc tỉnh → **Label Encoding** giúp giảm số chiều dữ liệu.
- timezone
→ Nhiều giá trị và không có thứ bậc → **Label Encoding** giúp tránh tăng chiều dữ liệu quá nhiều.

Sau khi áp dụng One-Hot Encoding và Label Encoding, các cột trong dataset có kiểu dữ liệu như sau:

- **ID và tên:** geonameid, name, asciiname – object
- **Kinh độ, vĩ độ và số liệu khác:** latitude, longitude, population, dem – float64
- **Các cột mã danh mục sau Label Encoding:** country code, admin1 code, timezone – int64

- Các cột **One-Hot Encoding** cho feature code: feature code_ENG, feature code_PPL, ..., feature code_PPLX – float64



Hình 2.6 Số lượng giá trị unique của mỗi cột

Tóm lại, dữ liệu sau tiền xử lý đã được chuẩn hóa về kiểu số (float/int) cho các mô hình Machine Learning, các cột danh mục được chuyển thành các cột nhị phân hoặc số nguyên.

6. Lựa chọn đặc trưng (Feature Selection)

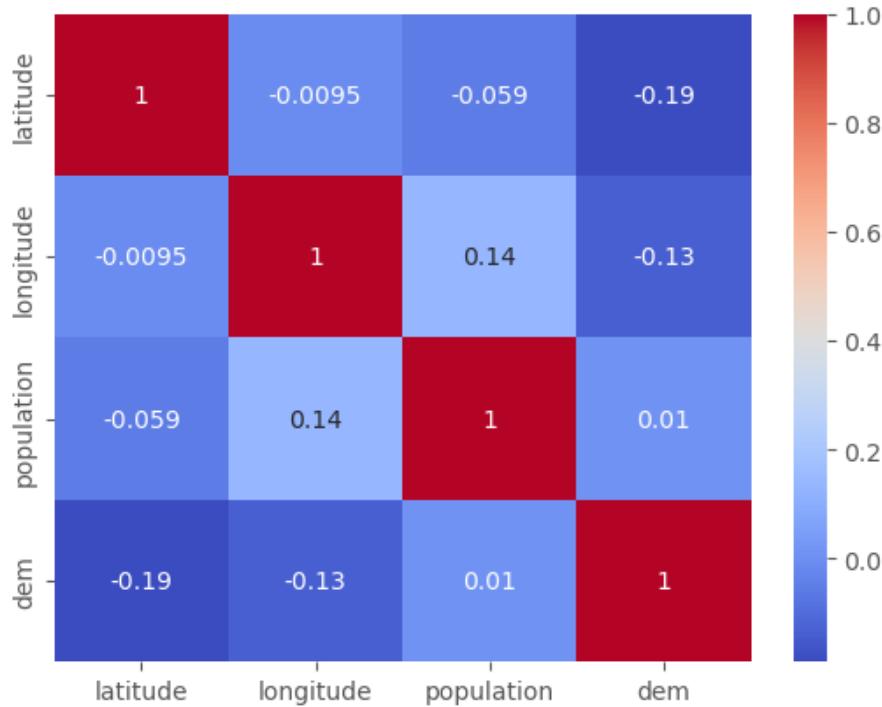
Loại bỏ các cột dữ liệu không cần thiết:

Có thể thấy rằng các cột geonameid, name, asciiname chứa các giá trị duy nhất và không mang thông tin phân loại hữu ích cho các thuật toán Machine Learning, do đó có thể bỏ qua.

Tương tự, cột feature_class chỉ có một giá trị duy nhất là P, không cung cấp thông tin phân biệt, nên cũng được loại bỏ khỏi dataset.

Phân tích mối tương quan giữa các cột số (numeric columns):

Các cột số được xét bao gồm: latitude, longitude, population, dem. Quan sát hệ số tương quan giữa các cặp cột:



Hình 2.7 Hệ số tương quan của các cột dữ liệu

- latitude vs longitude: $r \approx -0.01 \rightarrow$ gần như không có mối quan hệ tuyến tính.
- latitude vs population: $r \approx -0.06 \rightarrow$ tương quan rất yếu, không đáng kể.
- latitude vs dem: $r \approx -0.19 \rightarrow$ tương quan âm yếu, nghĩa là các khu vực có vĩ độ cao có xu hướng có độ cao thấp hơn một chút, nhưng không mạnh.
- longitude vs population: $r \approx 0.14 \rightarrow$ tương quan dương yếu, dân số hơi tăng theo kinh độ nhưng vẫn rất yếu.
- longitude vs dem: $r \approx -0.13 \rightarrow$ tương quan âm yếu, không đáng kể.
- population vs dem: $r \approx 0.01 \rightarrow$ gần như không có mối quan hệ.

Kết luận: Các biến số latitude, longitude, population, dem **hầu như không có tương quan tuyến tính mạnh** với nhau. Do đó, các mô hình dựa trên tuyến tính có thể không dự đoán tốt nếu chỉ sử dụng từng biến riêng lẻ.

7. Kết quả

Sau khi thực hiện các bước tiền xử lý:

- Dữ liệu sạch, không còn missing value quan trọng.
- Biến số chuẩn hóa, phân phối hợp lý.
- Biến phân loại được mã hóa phù hợp.

- Các cột dư thừa và unique đã được loại bỏ.
- Dữ liệu sẵn sàng cho các bước huấn luyện mô hình học máy.

2.4 Phân tích & đánh giá kết quả

2.4.1 Phân tích kết quả định lượng

Sau khi thực hiện các bước tiền xử lý, bao gồm:

- Loại bỏ các cột không cần thiết (geonameid, name, asciiname, feature class) vì chúng chứa các giá trị duy nhất hoặc không mang thông tin hữu ích cho mô hình Machine Learning.
- Điền giá trị missing: sử dụng giá trị xuất hiện nhiều nhất (*mode*) cho các biến định tính và giá trị trung vị (*median*) cho các biến số, nhằm đảm bảo rằng dữ liệu không bị thiếu và tránh ảnh hưởng đến quá trình huấn luyện mô hình.
- Thực hiện biến đổi log cho population để giảm độ lệch phai mạnh, làm cho phân phối dữ liệu trở nên gần chuẩn hơn và giảm ảnh hưởng của các giá trị ngoại lai (*outlier*).
- Chuẩn hóa dữ liệu: áp dụng *MinMaxScaler* cho latitude và longitude để đưa các giá trị về khoảng $[0, 1]$, đồng thời sử dụng *StandardScaler* cho dem nhằm đưa dữ liệu về trung tâm 0 với độ lệch chuẩn 1, giúp mô hình hội tụ nhanh và ổn định hơn.
- Áp dụng mã hóa dữ liệu danh mục: One-Hot Encoding cho cột feature code để chuyển các giá trị danh nghĩa thành các cột nhị phân, và Label Encoding cho các cột danh mục còn lại (country code, admin1 code, timezone) nhằm giảm số chiều dữ liệu mà vẫn giữ được thông tin phân loại.

Sau khi hoàn tất các bước này, có thể nhận thấy rằng các biến số (latitude, longitude, population, dem) đã có độ lệch chuẩn giảm đáng kể, giúp phân phối dữ liệu trở nên cân đối hơn. Các thống kê mô tả như *mean*, *std*, *min*, *max* cho thấy giá trị của các biến đã nằm trong phạm vi hợp lý, điều này không chỉ giúp giảm sự ảnh hưởng của các giá trị ngoại lai mà còn tạo điều kiện thuận lợi để các mô hình học máy hội tụ ổn định và đưa ra các dự đoán chính xác hơn trong các bước huấn luyện tiếp theo.

2.4.2 So sánh trực quan và diễn giải kết quả

Nhìn vào kết quả trực quan sau các bước tiền xử lý, có thể nhận thấy những thay đổi rõ rệt về phân phối của các biến số chính:

- latitude và longitude: sau khi áp dụng Min-Max Scaling, các giá trị vẫn giữ được phân phối gốc, nghĩa là hình dạng phân bố dữ liệu không thay đổi. Tuy nhiên, tất cả các giá trị đã được chuyển về khoảng $[0, 1]$, giúp các thuật toán học máy hoạt động ổn định hơn và tránh các vấn đề liên quan đến thang đo khác nhau giữa các biến.

- **population:** trước khi thực hiện log-transform, dữ liệu này bị lệch phai rất mạnh, với một số thành phố có dân số rất lớn so với phần còn lại. Sau khi biến đổi log, phân phối trở nên gần chuẩn hơn, các giá trị ngoại lai (*outlier*) được giảm ảnh hưởng, đồng thời thứ tự tương đối giữa các thành phố theo dân số vẫn được giữ nguyên, điều này giúp các mô hình học máy dựa trên khoảng cách hoặc phân phối chuẩn có thể học dữ liệu tốt hơn.
- **dem** (độ cao so với mực nước biển): mặc dù vẫn tồn tại một số outlier, nhưng sau khi chuẩn hóa với StandardScaler, khoảng giá trị đã giảm đáng kể từ $[-9999, 5022]$ xuống khoảng $[-20, 10]$. Việc này làm cho dữ liệu trở nên ổn định hơn, giảm sự ảnh hưởng của các giá trị cực đoan và giúp mô hình hội tụ nhanh hơn.

Nhìn chung, các bước tiền xử lý đã cải thiện đáng kể tính đồng nhất và cân đối của dữ liệu, đồng thời chuẩn hóa các biến về phạm vi hợp lý, tạo điều kiện thuận lợi cho việc huấn luyện các mô hình Machine Learning trong các bước tiếp theo.

2.4.3 Phân tích tác động về chất lượng dữ liệu

Việc áp dụng các bước tiền xử lý, bao gồm xử lý dữ liệu thiếu, loại bỏ các cột không cần thiết, chuẩn hóa các biến số và biến đổi log cho các giá trị lệch mạnh, đã đem lại những tác động tích cực rõ rệt đối với chất lượng dữ liệu:

- **Giảm ảnh hưởng của các giá trị cực đại (*outlier*):** Các giá trị bất thường, đặc biệt trong cột population và dem, sau khi biến đổi log và chuẩn hóa đã được làm mềm, giúp dữ liệu trở nên đồng đều hơn. Điều này tránh việc các outlier chi phối mô hình học máy và cải thiện khả năng học của các thuật toán nhạy cảm với khoảng cách hoặc độ lớn giá trị.
- **Hỗ trợ mô hình hội tụ ổn định và nâng cao hiệu suất:** Các thuật toán dựa trên gradient (như hồi quy tuyến tính, mạng neural) hoặc dựa trên khoảng cách (như KNN) đều yêu cầu dữ liệu có thang đo đồng nhất. Sau khi chuẩn hóa và scaling, các biến numeric nằm trong phạm vi hợp lý, giúp mô hình học máy hội tụ nhanh hơn và đưa ra dự đoán chính xác hơn.
- **Giữ lại thông tin quan trọng:** Mặc dù dữ liệu đã được biến đổi và chuẩn hóa, các bước này vẫn đảm bảo giữ được mối quan hệ tương đối và thứ tự của các quan sát, tránh mất thông tin giá trị quan trọng trong quá trình huấn luyện.

Kết luận: Nhìn chung, sau quá trình tiền xử lý, bộ dữ liệu đã đạt trạng thái cân bằng, phân phối hợp lý và ổn định hơn, sẵn sàng để được đưa vào các bước huấn luyện mô hình Machine Learning ở các chương tiếp theo. Các phương pháp này không chỉ cải thiện chất lượng dữ liệu mà còn tăng khả năng dự đoán và độ tin cậy của mô hình.

CHƯƠNG 03

TIỀN XỬ LÝ DỮ LIỆU VĂN BẢN

Trong chương này, nhóm chúng em sẽ trình bày về các kỹ thuật tiền xử lý dữ liệu dạng văn bản cho bộ dữ liệu Twitter 15 và 16, với mục tiêu phục vụ cho các bài toán phân loại bản tin thật hay tin giả, phục vụ việc tiền xử lý dữ liệu văn bản và áp dụng cho các mô hình học sâu (deep learning) trong việc phân loại và dự đoán văn bản.

Mục tiêu của chương này là xây dựng một quy trình hoàn chỉnh cho việc tiền xử lý văn bản một cách thống nhất và có khả năng tổng quát hoá cao. Đồng thời, việc tiền xử lý và phân tích các văn bản này sẽ giúp chúng ta học được nhiều kỹ thuật tiền xử lý hay.

Các kết quả sẽ được phân tích cụ thể, trực quan hoá và được phân tích định lượng và định tính, nhằm đánh giá tác động của từng kỹ thuật đối với chất lượng của văn bản.

3.1 Mô tả dữ liệu

Bộ dữ liệu được chọn sẽ là bộ dữ liệu về các tweet gốc trên nền tảng Twitter 15 và Twitter 16. Nhìn chung, Twitter 15 và Twitter 16 là hai bộ dữ liệu bao gồm hàng nghìn "cây lan truyền" (propagation trees) của các tweet. Mỗi cây đại diện cho một thông tin (một tweet gốc) và toàn bộ cấu trúc các tweet phản hồi (reply) và tweet lại (retweet) liên quan đến nó. Chúng được thu thập và giới thiệu bởi các nhà nghiên cứu như Jing Ma, Wei Gao, Kam-Fai Wong và cộng sự.

3.1.1 Giới thiệu về tập dữ liệu

Bộ dữ liệu Rumor Detection Dataset (Twitter15 and Twitter16) là một phiên bản đơn giản hoá của bộ dữ liệu gốc Twitter15 and Twitter16, trong đó bộ dữ liệu gốc được chủ yếu dùng trong việc phát hiện và phân loại các tin giả và tin kém thông tin.

Bộ dữ này chủ yếu tập trung vào các tweet gốc (source tweets). Mục đích chính của bộ dữ liệu này nhằm cung cấp dữ liệu cho các nhiệm vụ phân loại nhị phân (binary classification). Cụ thể, mỗi tweet gốc được phân loại đơn giản là "tin đồn" (rumor) hoặc "không phải tin đồn" (non-rumor). Để tạo điều kiện cho "cách tiếp cận đơn giản" này, phiên bản Kaggle đã cố ý bỏ qua cấu trúc chuỗi hội thoại phức tạp và các siêu dữ liệu bổ sung (additional metadata).

- Bộ dữ liệu này gồm 2 tập chính là Twitter 15 và Twitter 16. Mỗi tập gồm:
 - Source Tweets:
 - * Nội dung văn bản của tweet gốc (source_tweets).
 - * Phù hợp với các mô hình phân loại văn bản.
 - Labels
 - * Mỗi event (một source tweet cùng cascade replies/retweets) được gán một trong bốn nhãn:
 - True (T) — tin đúng (true rumor / verified true)

- False (F) — tin sai (false rumor)
- Unverified (U) — chưa xác thực / không rõ
- Non-rumor (NR) — không phải tin đồn (ví dụ: ordinary news / factual non-rumor)
- * Đây là kiểu nhãn “finer-grained” (4-way) thường dùng để làm bài toán phân lớp đa nhãn (multi-class)

Bộ dữ liệu này đóng vai trò quan trọng trong việc phân loại văn bản, loại bỏ sự phức tạp của cấu trúc cây và vấn đề ID tweet bị mất, cung cấp một tập hợp các tweet nguồn sạch, đã được gán nhãn có thể được sử dụng để huấn luyện các mô hình dựa trên văn bản (như BERT hoặc các bộ phân loại truyền thống) cho một tác vụ nhị phân đơn giản.

3.1.2 Hướng dẫn cài đặt dữ liệu

Đối với bộ dữ liệu này, chúng em chỉ thực hiện tiền xử lý dữ liệu trên bộ Twitter 15 mà thôi, bộ Twitter16 tương tự và số records của Twitter 16 nhỏ hơn 1000 nên nhóm chúng em sẽ không làm bộ này.

Tải đường dẫn đến bộ twitter15: data + label. Bạn cần đảm bảo phải vào đúng thư mục ‘Group_05‘ để có chạy được code. Mỗi file trong ‘data/text/twitter15‘ gồm các câu tweet gốc và ID lẩn nhãn tương ứng. Bạn có thể tải bộ dữ liệu này bằng cách sử dụng dataframe pandas.

Bạn có thể tải bộ dữ liệu này ở đây:

- Kaggle: [tại đây](#)
- Source gốc: Chunyuan Yuan, Qianwen Ma, Wei Zhou, Jizhong Han, Songlin Hu. Jointly embedding the local and global relations of heterogeneous graph for rumor detection. In 19th IEEE International Conference on Data Mining, IEEE ICDM 2019.

3.1.3 Phân tích tổng quan dữ liệu

Bộ dữ liệu này bao gồm: Twitter 15 với 1490 văn bản và Twitter 16 với 818 văn bản, mỗi văn bản sẽ được gán một trong 4 loại nhãn ‘non-rumor’, ‘unverified’, ‘true’, ‘false’.

3.1.4 Lý do chọn tập dữ liệu này ?

Bộ dữ liệu này là một trong những bộ dữ liệu văn bản đầy đủ và phổ biến nhất, cung cấp dữ liệu thật trên mạng Twitter 15 và Twitter 16.

Ưu điểm

- Bộ dữ liệu này có cấu trúc, mỗi event là một source tweet kèm cascade replies/retweets (tree/graph). Điều này cho phép nghiên cứu ảnh hưởng của ngữ cảnh reply/stance, luồng thông tin, và biểu hiện lan truyền — khác với tập chỉ chứa tweet rời rạc.
- Đồng thời, bộ dữ liệu này có nhãn chi tiếp 4 lớp (True / False / Unverified / Non-rumor), điều này phù hợp cho các bài toán phân lớp tinh vi (không chỉ binary), giúp đánh giá độ nhạy mô hình về nhiều dạng “tin”.

- Bộ dữ liệu này đã được dùng rộng, có baseline và splits sẵn. Điều này được minh chứng rằng nhiều công trình so sánh trên Twitter15/16 nên dễ benchmark, có các train/dev/test chuẩn để so sánh công bằng.

Nhược điểm

- Bộ dữ liệu này có kích thước nhỏ (Twitter15 \approx 1,490 event; Twitter16 \approx 818 event). Điều này dễ gây biến động kết quả, overfitting, và giới hạn khả năng tổng quát hóa. Hậu quả là cần chạy nhiều seed / cross-validation; kết quả so sánh có thể không ổn định nếu chỉ một vài seed.
- Bộ dữ liệu này nhẫn có thể chứa noise / định nghĩa không đồng nhất

3.2 Cơ sở tiền xử lý

Chúng ta cần phải xử lý bộ dữ liệu này vì:

- Các dấu câu xuất hiện rất nhiều trong từng tweet. Điều này ảnh hưởng tới bước tiền xử lý vì phải quyết định giữ hay loại bỏ, và làm thế nào để chuẩn hoá.
- Các hình ảnh emoji xuất hiện rất nhiều trong các tweet trên mà không có ý nghĩa về mặt hình ảnh.
- Một số tweets bao gồm nhiều URLs, điều này không đóng góp vào việc phân loại tin giả hay tin thật vì nội dung của các URLs này sẽ bị che mất khi nhấp vào.
 - Chúng ta cần phải loại bỏ những URLs vô nghĩa này, giảm được số lượng token và giúp model tập trung vào những chi tiết quan trọng.
- Các tài khoản được gắn thẻ trong tweet (mentions), điều này là bình thường. Tuy nhiên, một quan sát bất thường là ký hiệu '@' ở một số trường hợp được theo sau bởi một dấu cách trước khi tên tài khoản, định dạng không chuẩn này cần được xử lý trong bước làm sạch dữ liệu.
 - Ví dụ: thay vì '@username' có khi dữ liệu chứa '@ username' — cần chuẩn hoá thành '@username' hoặc loại bỏ toàn bộ mention.
- Các dấu hashtags () có xuất hiện nhưng được viết với một dấu cách giữa dấu “ ” và từ tag.
 - Ví dụ: thay vì là 'topic' thì dữ liệu có dạng ' topic' → làm mất tính nhất quán của hashtag → cần phải được sửa lại trong tiền xử lý.

Các kỹ thuật được áp dụng là:

- Làm sạch dữ liệu
 - Chuyển đổi sang chữ cái thường

- Loại bỏ Hashtags
- Loại bỏ đường dẫn URLs
- Loại bỏ tất cả mentions
- Xoá các hàng trống
- Mở rộng chữ viết tắt
- Loại bỏ hết tất cả chữ chửi tục và dấu *
- Loại bỏ và thay thế các từ tiếng lóng (slang)
- Chuyển đổi icon emoji thành chữ tương ứng
- Loại bỏ các chữ số
- Loại bỏ đi các ký tự đặc biệt
- Loại bỏ đi các câu / các từ không phải là tiếng Anh
- Loại bỏ hết tất cả dấu câu
- Loại bỏ đi các khoảng trắng
- Tokenization
 - Word Tokenization
 - Sentence Tokenization
 - Subword Tokenization
- Loại bỏ đi các Stop Words và Non Words
- Stemming
- Lemmatization
- Vector Tokenization
 - Bag-Of-Words
 - TF-IDF
 - Word2Vec
- Trực quan hóa
 - Word Cloud
 - Word Frequency

Các kỹ thuật trên đều được trình bày chi tiết sau đây.

3.3 Triển khai chi tiết

3.3.1 Khai phá dữ liệu

Sau khi khai phá dữ liệu, bộ dữ liệu không bị thiếu dữ liệu, và không bị trùng lặp dữ liệu.

Bộ dữ liệu sẽ gồm có 3 cột chính là: ID, text, targets

Sau khi chúng ta khám phá dữ liệu trên, chúng ta rút ra được các kết luận như sau:

- Các dấu câu xuất hiện rất nhiều trong từng tweet. Điều này ảnh hưởng tới bước tiền xử lý vì phải quyết định giữ hay loại bỏ, và làm thế nào để chuẩn hoá.
- Các hình ảnh emoji xuất hiện rất nhiều trong các tweet trên mà không có ý nghĩa về mặt hình ảnh.
- Một số tweets bao gồm nhiều URLs, điều này không đóng góp vào việc phân loại tin giả hay tin thật vì nội dung của các URLs này sẽ bị che mất khi nhấp vào.
 - Chúng ta cần phải loại bỏ những URLs vô nghĩa này, giảm được số lượng token và giúp model tập trung vào những chi tiết quan trọng.
- Các tài khoản được gắn thẻ trong tweet (mentions), điều này là bình thường. Tuy nhiên, một quan sát bất thường là ký hiệu '@' ở một số trường hợp được theo sau bởi một dấu cách trước khi tên tài khoản, định dạng không chuẩn này cần được xử lý trong bước làm sạch dữ liệu.
 - Ví dụ: thay vì '@username' có khi dữ liệu chứa '@ username' — cần chuẩn hoá thành '@username' hoặc loại bỏ toàn bộ mention.
- Các dấu hashtags (#) có xuất hiện nhưng được viết với một dấu cách giữa dấu '#' và từ tag.
 - Ví dụ: thay vì là '#topic' thì dữ liệu có dạng '# topic' → làm mất tính nhất quán của hashtag → cần phải được sửa lại trong tiền xử lý.

3.3.2 Làm sạch dữ liệu

Lưu bản sao của dataframe gốc và tweets

Để đảm bảo việc lưu trữ dữ liệu dataframe gốc vẫn được toàn vẹn mà không cần phải load lại dữ liệu gốc nhiều lần, ta cần phải tạo một bản sao của dataframe gốc. Điều này giúp ngăn chặn việc thực hiện những thay đổi vô ý đối với dataframe gốc.

Chuyển đổi tweet sang chữ viết thường

Bằng cách chuyển đổi các tweet gốc sang chữ viết thường nhằm đảm bảo tính đồng nhất về mặt con chữ, vì lúc này chữ hoa và chữ thường của cùng một từ là giống hệt nhau. Việc chuẩn hóa này làm giảm độ phức tạp của việc phân tích văn bản, giúp việc so sánh và phân tích dễ dàng hơn.

Loại bỏ các hashtags (#)

Dựa vào một số quan sát trên, mỗi tweet có thể đều có hashtag có thể có hoặc không một hay nhiều khoảng trắng giữa ký hiệu và tên của tag. Do đó chúng ta cần phải xoá khoảng trắng này trước khi trích xuất hashtags (vì riêng lẻ thì khó có thể loại bỏ hoàn toàn hashtags).

Sau khi xóa khoảng trắng, ta sử dụng biểu thức chính quy (regex) để xác định chính xác các hashtag.

Biểu thức ‘([a-zA-Z0-9_]1,50)‘ được thiết kế để khớp với định dạng hashtags:

- ‘#‘ dùng để định vị các tìm kiếm cho các điểm bắt đầu của một hashtags.
- ‘[a-zA-Z0-9_]‘: dùng để xác định bất kỳ sự kết hợp nào của số, chữ cái (viết hoa hoặc viết thường) hoặc dấu gạch dưới.
- ‘1,50‘: dùng để xác định độ dài của một hashtags (có kích thước từ 1 đến 50 ký tự)
- Các ràng buộc về nội dung và độ dài được đặt trong dấu ngoặc đơn để chỉ ra rằng chỉ nội dung hashtag mới phải tuân thủ các quy tắc này.

Loại bỏ các đường dẫn URLs

Dựa vào dữ liệu thô ban đầu, ta có thể thấy các đường dẫn URL xuất hiện khá nhiều ở mỗi câu. Chúng ta có thể sử dụng biểu thức regex để tìm kiếm các đường dẫn URL, tương tự như cách chúng ta tìm kiếm hashtags.

Loại bỏ các mentions ‘@user‘

Một lần nữa, ta có thể thấy các mentions có thể có hoặc không có khoảng trắng giữa ký hiệu ‘@‘ và tài khoản được truy cập. Ngoài ra, ký hiệu ‘@‘ có thể được sử dụng để thay thế từ ‘at‘, thường dùng để chỉ thời gian trong ngày (ví dụ: ‘@9pm‘). Thông tin này cũng không liên quan đến việc phân loại tin giả hay thật, nên ta có thể loại bỏ đi nhiều này. Thêm vào đó, dấu ‘@‘ có thể xuất hiện ở trong email, điều này có thể ảnh hưởng đến ý nghĩa phân loại đầu ra nên chúng ta có thể không loại bỏ đặc trưng này.

Xoá đi các hàng trống

Ta sẽ cần lưu ý rằng một số tweet có thể chỉ chứa URL, nội dung đề cập hoặc hashtag. Do đó, chúng ta sẽ xóa bất kỳ hàng nào chứa tweet trống bằng cách dùng ‘pd.drop‘ hoặc phép trừ.

Mở rộng các từ viết tắt

Các từ viết tắt trong tiếng Anh (ví dụ, “isn’t”, “wouldn’t”, hay “I’ve”) là các dạng rút gọn của các từ tiếng Anh. Để đảm bảo tính nhất quán và tính rõ ràng về mặt ý nghĩa câu từ, em sẽ xử lý việc mở rộng các chữ viết tắt này thành dạng đầy đủ. Để thực hiện việc này, em sẽ sử dụng thư viện ‘contractions‘, thư viện chứa danh sách đầy đủ các chữ viết tắt.

Loại bỏ hết tất cả dấu hoa thị () (bao gồm từ viết tắt thô lỗ các ký tự độc lập)

Kết quả là có dấu ở rất nhiều chỗ trong bộ dữ liệu. Theo quan sát, dấu thường nằm giữa các từ chửi thề (swearwords) và có khoảng trắng giữa các từ này. Đồng thời, các dấu cũng nằm riêng

lẻ giữa các từ khác nhau nhầm lẫn mạnh từ đó. Việc đầu tiên chúng ta cần làm là làm sao thay thế các từ chửi thề này. Sau đó, là loại bỏ hoàn toàn các dấu riêng lẻ giữa các từ.

Để làm được chuyện này, đầu tiên, ta cần tách chuỗi thành 3 phần, phần đầu không phải chữ / số / '*' (nếu có), phần giữa (nội dung thực sự), và phần cuối không phải chữ / số / '*' (nếu có). Mục đích là dùng để lấy các 'chữ / số / wordy' ở giữa, đồng thời giữ lại (hoặc xử lý) phần punctuation ở đầu và cuối (không coi '*' là phần dấu câu).

Sau khi ta tìm được các substring tương ứng với từng root, ta tiến hành dùng hàm 'remove_swearwords' để xoá các từ thô lỗ.

Sau khi loại bỏ các từ chửi thề, ta sẽ loại bỏ đi các dấu '*' đơn lẻ tương ứng.

Loại bỏ và thay thế các từ tiếng lóng (Slang words)

Đầu tiên, ta sẽ tải một tập từ điển về các từ tiếng lóng để làm chuẩn ('slang.csv')

Sau đó, ta sẽ chỉnh sửa và tiến xử lý bộ từ điển này

Tiếp theo, ta sẽ dùng hàm 'replace_slang' để thay thế các từ tiếng lóng này

Chuyển đổi các icon emoji sang chữ tương ứng

Để chuyển đổi các emoji sang chữ tương ứng, ta sẽ dùng thư viện 'emoji' để phát hiện và chuyển đổi các emoji tương ứng thành chữ. Đồng thời, sau khi chuyển đổi, ta có thể lưu lại metadata (vị trí, ký tự, tên) để phục vụ highlight, analytics, hoặc tái xây dựng. Cuối cùng, ta đã chuyển tất cả các icon emoji thành chữ tương ứng.

Loại bỏ hết tất cả chữ số

Sau đó, ta sẽ loại bỏ đi tất cả các chữ số có dính liền với từ. Ví dụ như: '1990s', '7up', '9am', ... bởi vì những thông tin này không có ý nghĩa cho việc phát hiện tin thật hay giả.

Dựa vào bảng thống kê, ta thấy các từ có lẩn số đã được loại bỏ. Trong đó '1970s' bị xoá nhiều nhất (12 lần).

Loại bỏ đi các ký tự đặc biệt

Ta cần phải loại bỏ đi các ký tự đặc biệt mà không có ý nghĩa cho việc phân loại, nhưng giữ lại mọi ký tự là chữ (letters), số (numbers) và khoảng trắng (spaces). Ghi lại danh sách các ký tự bị loại bỏ (unique, giữ thứ tự xuất hiện lần đầu) cho mỗi hàng, và lưu kết quả vào hai cột mới của DataFrame: 'text' (chuỗi đã sạch) và 'removed_chars' (list các ký tự bị loại bỏ).

Loại bỏ các câu / các từ không phải là tiếng Anh

Để loại bỏ được các câu / các từ không phải tiếng Anh, ta sẽ sử dụng thư viện 'langdetect' để xác định xem các ngôn ngữ ở các câu là ngôn ngữ gì. Nếu câu đó không phải là tiếng Anh ('en'), chúng ta sẽ loại bỏ nó khỏi dữ liệu này và lưu lại trong bảng 'removed_df'. Sau khi loại bỏ, từ 1490 câu nay đã thành 1439 câu tiếng Anh.

Loại bỏ hết tất cả dấu câu

Bây giờ, chúng ta sẽ loại bỏ hết tất cả dấu câu và một số ký tự đặc biệt ra khỏi tất cả các tweet.

Sau đó, trả về danh sách các ký tự đã được tìm thấy trong chuỗi để ghi logs.

Loại bỏ hết tất cả khoảng trắng

Trong bước này, chúng ta sẽ loại bỏ bất kỳ ký tự khoảng trắng nào có trong tweet. Việc loại bỏ khoảng trắng đảm bảo tính nhất quán và thống nhất của văn bản trong toàn bộ tập dữ liệu.

Lưu bảng dữ liệu được làm sạch

Ở giai đoạn này, chúng ta sẽ tạo một phiên bản tweet sạch, loại bỏ mọi khoảng trắng. Bước này sẽ trở nên có ý nghĩa hơn trong bài viết tiếp theo, nơi chúng ta sẽ tiến hành phân tích khám phá các tweet. Phiên bản tweet đã được làm sạch, không có khoảng trắng, sẽ làm cơ sở cho các bước xử lý tiếp theo như lemmatization và remove stop words. Đây là bước cuối cùng mà tweet sẽ giữ được tính mạch lạc trong tiếng Anh.

3.3.3 Tokenization

Sự token hoá (Tokenization) là bước cơ bản, nền tảng trong việc phân tích và xử lý các ngôn ngữ tự nhiên sâu hơn. Nó là quá trình phân chia đầu vào có dạng văn bản thành các đơn vị nhỏ hơn, mỗi đơn vị đó được gọi là tokens. Các tokens này có thể là từ, câu, subwords, hoặc các yếu tố có ý nghĩa khác, tùy thuộc vào từng nhiệm vụ và yêu cầu cụ thể. Tokenization giúp trích xuất các khối xây dựng cơ bản của văn bản, giúp phân tích và hiểu thông tin được dễ dàng hơn.

Sự token hoá ở cấp độ từ (Word Tokenization)

Token hoá ở cấp độ từ (Word Tokenization) là phương pháp được sử dụng phổ biến nhất, trong đó việc chia các từ thành những token riêng lẻ. Phương pháp này hiệu quả đối với những ngôn ngữ có mức độ ranh giới rõ ràng như tiếng Anh.

Sự token hoá ở cấp độ câu (Sentence Tokenization)

Token hoá ở cấp độ câu (Sentence Tokenization) cũng là một kỹ thuật phổ biến được sử dụng để chia các đoạn văn hoặc tập hợp thành các câu riêng biệt dưới dạng các tokens. Kỹ thuật này hữu ích cho các tác vụ yêu cầu phân tích hoặc xử lý từng câu riêng lẻ.

Sự token hoá ở cấp độ subword (Subword Tokenization)

Điều này tạo ra sự cân bằng giữa việc phân chia từ và ký tự bằng cách chia nhỏ văn bản thành các đơn vị lớn hơn một ký tự nhưng nhỏ hơn một từ đầy đủ. Điều này hữu ích khi xử lý các ngôn ngữ có hình thái phong phú hoặc các từ hiếm.

So sánh và kết luận

Sau khi áp dụng tách câu (sentence), tách từ (word) và tách subword (BPE) cho tập dữ liệu đã làm sạch, ta có các quan sát và so sánh sau:

- Sentence Tokenization: Chia văn bản thành các câu dựa trên dấu câu và quy tắc ngôn ngữ.
 - Ưu điểm

- * Bắt được ranh giới cú pháp — hữu ích cho các tác vụ cần ngữ cảnh câu (ví dụ tóm tắt, phân tích cảm xúc theo câu)
- * Cho phép phân tích chi tiết khi đầu vào có nhiều câu.
- Nhược điểm
 - * Tweet và văn bản ngắn thường dùng dấu câu không chính quy, emoji, viết tắt -> có thể gây chia sai câu.
 - * Ít có lợi nếu dữ liệu hầu hết là câu đơn/ngắn (như tweet).
 - * Ngay bước tiền xử lý, em đã loại bỏ hết dấu câu nên việc sentence tokenization ko có ý nghĩa đối với dữ liệu tweet.
- Word Tokenization: Tách văn bản thành từ/token và dấu chấm câu.
 - Ưu điểm
 - * Đơn giản, dễ hiểu và hỗ trợ rộng rãi.
 - * Phù hợp cho các phương pháp truyền thống (bag-of-words, TF-IDF).
 - * Nắm bắt được các đơn vị ngôn ngữ có ý nghĩa (danh từ, động từ...).
 - * Bộ dữ liệu này toàn câu ngắn nên việc dùng word tokenization là có ý nghĩa.
 - * Do em đã xử lý các nhiễu, slang, từ sai chính tả trước đó nên việc dùng này là hợp lý.
 - Nhược điểm
 - * Không xử lý tốt từ out of vocabulary -> ví dụ tên riêng hiếm, lỗi chính tả.
 - * Nhạy với các biến thể hình thái (run, running, ran được coi là khác nhau).
 - * Kích thước vocabulary lớn -> tính thừa thót cao.
- Subword Tokenization: Tách từ thành các đơn vị nhỏ hơn (subwords) dựa trên tần suất các cặp ký tự liền kề.
 - Ưu điểm
 - * Giảm đáng kể kích thước vocabulary → hiệu quả lưu trữ và mô hình.
 - * Xử lý tốt từ hiếm hoặc chưa gặp (ví dụ: biến thể chính tả, từ mới).
 - * Phù hợp cho các mô hình neural (Transformers, LLMs) — subword giúp tổng quát hoá tốt hơn.
 - * Cân bằng giữa character-level và word-level.
 - Nhược điểm
 - * Khó diễn giải trực giác — subword có thể không tương ứng với từ có ý nghĩa.
 - * Cần huấn luyện (tốn thời gian) để tìm các merge rules.
 - * Với dataset quá nhỏ, các merge có thể kém tổng quát.

Cuối cùng, vì phạm vi của bài tập tiền xử lý này không dùng các mô hình deep learning hoặc transfer learning (BERT/GPT-style) nên em sẽ không dùng ‘subword tokenization’ và ‘sentence tokenization’. Em sẽ chọn ‘word tokenization’ để tối ưu hoá cho các bước phân tích tiếp theo

3.3.4 Loại bỏ các Stop-Words và Non-Words

Loại bỏ các Stop-Words

Stop Words là các từ thường được sử dụng trong một ngôn ngữ nhất định, chẳng hạn như "a", "an", "the", "is", "are", "am", ...

- Những từ này thường không có giá trị về mặt ý nghĩa và phân tích cho kết quả đầu ra.
- Bằng cách loại bỏ các stop words, ta có thể loại bỏ đi những thông tin không cần thiết khỏi các tweet.
- Ví dụ: Cho câu "The sky is blue", các từ "the" và "is" là các từ đóng góp không đáng kể vào kết quả phân tích tin thật hay giả được thể hiện.
 - Do đó, việc loại bỏ những từ này giúp đơn giản hoá quá trình phân tích và nâng cao độ chính xác của việc phân loại tin thật hay tin giả.

Các stop words xuất hiện trong clean_df: ['a', 'about', 'above', 'after', 'again', 'against', 'all', 'am', 'an', 'and', 'any', 'are', 'aren', 'as', 'at', 'be', 'because', 'been', 'before', 'being', 'between', 'both', 'but', 'by', 'can', 'd', 'did', 'do', 'does', 'doesn', 'doing', 'don', 'down', 'during', 'for', 'from', 'further', 'had', 'has', 'hasn', 'have', 'having', 'he', 'her', 'here', 'herself', 'him', 'himself', 'his', 'how', 'i', 'if', 'in', 'into', 'is', 'it', 'its', 'itself', 'just', 'm', 'me', 'more', 'most', 'my', 'no', 'not', 'now', 'o', 'of', 'off', 'on', 'once', 'only', 'or', 'other', 'our', 'out', 'over', 'own', 're', 's', 'same', 'she', 'should', 'so', 'some', 'such', 't', 'than', 'that', 'the', 'their', 'them', 'themselves', 'then', 'there', 'these', 'they', 'this', 'those', 'through', 'to', 'too', 'under', 'until', 'up', 've', 'very', 'was', 'wasn', 'we', 'were', 'what', 'when', 'where', 'which', 'while', 'who', 'why', 'will', 'with', 'won', 'y', 'you', 'your', 'yourself', 'yourselves']

Tổng số stop words xuất hiện trong dữ liệu: 127

Loại bỏ các Non-Words

Trong bước này, ta muốn loại bỏ đi bất kỳ từ đơn lẻ nào có thể vẫn còn ở trong văn bản hiện tại.

- Ví dụ: ký tự "J" trong "Donald J.Trump".

Để làm được điều này, chúng ta cần chỉ định một loạt các chữ cái để loại bỏ nếu chúng xuất hiện riêng lẻ, ngoại trừ các chữ cái như "I", hay "a", vốn có ý nghĩa riêng lẻ. Bằng cách loại bỏ những chữ cái đơn lẻ này, ta có thể tinh chỉnh văn bản hơn nữa và đảm bảo rằng chỉ còn lại những từ có nghĩa để phân tích sau.

Phân tích và kết luận

Đầu tiên ta có 4.418 từ khác nhau trước khi loại stopwords, và 127 stopwords khác nhau xuất hiện trong dữ liệu. Nếu loại tất cả các stopwords này, số lượng từ duy nhất giảm khoảng $127/44182.87\%$ — tức là giảm rất nhỏ về số loại token. Mức giảm về số token tổng (tức tổng các token xuất hiện trong toàn bộ corpus) thường lớn hơn nhiều so với giảm vocabulary. Kết luận:

- Thông tin mất ít về chủ đề/từ khoá quan trọng (topic words) — vì stopwords thường không chứa nội dung chủ đề.
- Có thể mất thông tin quan trọng cho một số tác vụ: phân tích cảm xúc, hay phân loại tin giả / tin thật.
- Xóa stopwords ít giảm kích thước vocab, nhưng rất hữu dụng để giảm nhiễu, giảm sparsity, và thường giúp các mô hình cổ điển (TF-IDF + Logistic, LDA) hoạt động tốt hơn.

3.3.5 Stemming và Lemmatization

Trong xử lý ngôn ngữ tự nhiên (NLP), stemming và lemmatization là các kỹ thuật tiền xử lý văn bản nhằm rút gọn các dạng biến thể của từ trong một tập dữ liệu văn bản về một từ gốc chung hoặc dạng từ trong từ điển, còn được gọi là “lemma” trong ngôn ngữ học tính toán.

Stemming

Các thuật toán stemming khác nhau rất nhiều, mặc dù chúng có một số cách hoạt động chung. Các bộ stemmer loại bỏ hậu tố của từ bằng cách so sánh các token từ đầu vào với một danh sách đã định trước các hậu tố phổ biến. Sau đó stemmer sẽ xóa bất kỳ chuỗi ký tự hậu tố nào tìm thấy khỏi từ, với điều kiện từ vừa được xóa không vi phạm bất kỳ quy tắc hoặc điều kiện nào áp dụng cho hậu tố đó.

Porter Stemming

Porter stemming là một thuật toán rule-based để rút gọn các dạng biến thể của từ về một stem (từ gốc) nhằm giảm số lượng dạng từ khác nhau trong văn bản.

Thực hiện thuật toán Porter Stemming bằng thư viện ‘nltk’. Sau đó lưu vào ‘tokens_porter_list’.

Ưu điểm

- Nhanh, nhẹ, dễ implement.
- Đã là chuẩn trong IR và các tasks truyền thống. Giảm tốt số loại token → giảm sparsity.

Nhược điểm

- Có thể tạo stem không phải từ thật, gây gộp nghĩa khác nhau.
- Dựa trên quy tắc tĩnh nên không hiểu ngữ nghĩa; đôi khi loại sai thông tin (ví dụ phủ định, tên riêng).

Snowball Stemming

Snowball Stemming là một dạng nâng cấp biến thể của Porter Stemming. Thuật toán này cải thiện độ chính xác so với Porter gốc, làm code rõ ràng hơn, dễ mở rộng cho nhiều ngôn ngữ và giảm lỗi over-/under-stemming.

Thực hiện thuật toán snowball stemming cho các tokens và trả kết quả về ‘tokens_snowball_list’.

Ưu điểm

- Ổn định hơn Porter gốc: ít lỗi lạ, áp dụng thứ tự rules hợp lý hơn.
- Hỗ trợ đa ngôn ngữ (cùng framework) — thuận tiện cho corpus đa ngôn ngữ.
- Nhanh và lightweight
- Dễ tích hợp trong pipeline preprocessing

Nhược điểm

- Vẫn là rule-based — không hiểu ngữ nghĩa, có thể gây over-/under-stemming.
- Kết quả không phải lemma — nhiều stem không phải từ hợp lệ
- Không xử lý hoàn hảo morphology của mọi ngôn ngữ phức tạp

Lemmatization

Lemmatization bao gồm việc phân tích một từ thành dạng nguyên khởi, cơ sở dưới dạng từ điển của nó. Ví dụ, động từ “walk” có thể xuất hiện dưới nhiều dạng khác nhau như “walking”, “walks” hoặc “walked”. Lemmatization chuẩn hóa các từ này, giúp chúng hoàn toàn có thể so sánh được với nhau. Khác với stemming, trong khi stemming chỉ loại bỏ các tiền tố, hậu tố phổ biến khởi phàn cuối của từ thì lemmatization đảm bảo từ đầu ra là một dạng chuẩn hóa hiện có của từ (ví dụ: lemma) có thể tìm thấy trong từ điển. Vì lemmatization hướng đến việc xuất ra các dạng cơ sở từ điển, nó đòi hỏi phân tích hình thái mạnh mẽ hơn so với việc tách từ gốc. Part Of Speech (POS) tagging là một bước quan trọng của lemmatization.

So sánh và kết luận

Stemming

- **Ưu điểm**
 - Nhanh, phù hợp nếu chỉ cần so khớp từ hoặc thống kê tần suất.
 - Không yêu cầu tài nguyên lớn (không cần POS tagger, WordNet).
- **Nhược điểm**
 - Tạo ra nhiều từ giả không có nghĩa
 - Không phân biệt từ loại -> dễ mất ngữ nghĩa.

- Với dữ liệu tweet, có nhiều cấu trúc không chuẩn -> dễ bị cắt sai.
- Kết luận: Giảm chất lượng vector hóa (TF-IDF, embeddings) và làm giảm độ chính xác mô hình.

Lemmatization:

- Ưu điểm
 - Dựa trên WordNet, hiểu được từ loại và ngữ nghĩa.
 - Cho kết quả từ có nghĩa chuẩn, thống nhất dạng gốc thật.
 - Giúp cải thiện độ chính xác khi trích xuất đặc trưng (TF-IDF, Word2Vec, BERT).
 - Giảm nhiễu ngữ nghĩa.
- Nhược điểm
 - Tốc độ chậm hơn.
 - Cần POS tagger và WordNet (phải cài tài nguyên NLTK).

Tôi sẽ chọn sử dụng ‘Lemmatization’ vì độ chính xác và ngữ cảnh của văn bản rất quan trọng và các nguồn tài nguyên tính toán không bị hạn chế. Vì tôi không cần chạy nhanh thì không cần dùng ‘stemming’.

- Cho ra từ gốc có nghĩa thật, giúp biểu diễn vector chính xác hơn.
- Lemmatization có thể tận dụng POS tagging để phân biệt “run” (động từ) và “run” (danh từ), điều mà stemming không làm được.

3.3.6 Vector hóa văn bản (Text Vectorization)

Bag-Of-Words

Bag-of-Words (BoW) là kỹ thuật trích xuất đặc trưng từ dữ liệu văn bản cho phương pháp học máy ví dụ như phân loại văn bản (text classification) hay phân tích về mặt cảm xúc (sentiment analysis).

Điều này rất quan trọng vì thuật toán học máy không thể xử lý dữ liệu văn bản. Quá trình chuyển đổi văn bản thành số được gọi là trích xuất đặc trưng hoặc mã hóa đặc trưng.

Phương pháp này dựa trên sự xuất hiện của các từ trong một document. Quá trình này bắt đầu bằng việc tìm kiếm từ vựng trong văn bản và đo lường sự xuất hiện của chúng. Nó được gọi là bag-of-words vì thứ tự và cấu trúc của từ không được xem xét, mà chỉ là sự xuất hiện của chúng. Chúng ta sử dụng phương pháp này nhằm:

- Trích xuất đặc trưng: phương pháp này chuyển đổi dữ liệu không cấu trúc sang dữ liệu có cấu trúc.
- Đơn giản và hiệu quả: dễ cài đặt, hoạt động tốt với những bộ dữ liệu nhỏ

- Đo độ tương đồng giữa văn bản: có thể được sử dụng để tính độ tương đồng giữa các text documents bằng cách sử dụng cosine similarity.
- Phân loại văn bản: dễ áp dụng cho các bài toán nhận diện spam, phân tích cảm xúc.

Đầu tiên, tạo corpus bằng tất cả các câu đã được tiền xử lý. Sau đó, khởi tạo CountVectorizer

- Khởi tạo:
 - `CountVectorizer(min_df=2, max_df=0.95, ngram_range=(1,1))`
- Ý nghĩa các tham số:
 - `min_df=2`: loại bỏ các token (feature) xuất hiện ít hơn 2 document — giúp giảm nhiễu và kích thước vocabulary.
 - `max_df=0.95`: bỏ các token xuất hiện trong hơn 95% các document (rất phổ biến, thường là stopwords).
 - `ngram_range=(1,1)`: chỉ lấy unigrams (1-gram).
- Bước thực hiện: vectorizer sẽ xây dựng vocabulary gồm các từ (hoặc n-gram) thỏa điều kiện `min_df / max_df`, sau đó đếm tần suất xuất hiện của mỗi từ trong mỗi document (tạo ma trận document-term).

Sau đó, thực hiện học vocabulary từ corpus (xác định các feature theo tokenization, preprocessing mặc định) và biến mỗi document thành một vector tần suất dựa trên vocabulary vừa học.

- Ưu điểm**

- Rất đơn giản, dễ hiểu và dễ triển khai.
- Tạo ra vector thưa (sparse) giúp lưu trữ và xử lý hiệu quả.
- Hoạt động tốt với các thuật toán tuyến tính đơn giản như Logistic Regression.

- Nhược điểm**

- Tạo ra các vector có chiều rất cao, tốn tài nguyên.
- Kích thước từ vựng ảnh hưởng đến độ thưa thớt của biểu diễn: từ vựng càng lớn, biểu diễn càng thưa thớt và đa chiều. Độ thưa thớt này có thể khiến các mô hình khó học hiệu quả hơn và đòi hỏi phải điều chỉnh cẩn thận kích thước từ vựng để tránh chi phí tính toán quá mức.

- Tạo ra các ma trận thưa thớt tốn kém về mặt tính toán: vì mỗi tài liệu được biểu diễn bằng tần suất của từng từ trong một kho từ vựng có thể rất lớn, nên các ma trận kết quả thường chủ yếu là số không. Điều này có thể không hiệu quả khi lưu trữ và xử lý trong các quy trình học máy. Ma trận thưa thớt tiêu tốn đáng kể bộ nhớ và thường yêu cầu các công cụ và thư viện chuyên biệt để lưu trữ và tính toán hiệu quả, đặc biệt là với các tập dữ liệu lớn.
- Mất ý nghĩa và ngữ cảnh: BOW bỏ qua trật tự từ và cấu trúc câu, dẫn đến mất đi mối quan hệ ngữ pháp và ý nghĩa. Hạn chế này khiến nó kém phù hợp hơn cho các nhiệm vụ đòi hỏi ngữ cảnh, sắc thái và trật tự từ quan trọng, chẳng hạn như dịch thuật hoặc phát hiện cảm xúc trong các câu phức tạp.

TF-IDF Vectorization

Term Frequency (TF) thể hiện tần suất xuất hiện của một thuật ngữ trong một tài liệu. Inverse Document Frequency (IDF) làm giảm tác động của các từ thường gặp trên nhiều tài liệu. Điểm TF-IDF được tính bằng cách nhân hai chỉ số này.

‘TfidfVectorizer(...)' tạo một object của scikit-learn, là kết hợp ‘CountVectorizer‘ + ‘TfidfTransformer‘ 1) tách token và xây vocabulary 2) tính TF (term frequency) cho mỗi document 3) tính IDF (inverse document frequency) 4) nhân TF * IDF và (mặc định) chuẩn hoá vector (L2).

Sau đó, thực hiện học vocabulary và IDF từ corpus và chuyển mỗi document thành 1 vector TF-IDF.

TF-IDF cỗ gắng gán trọng số cho mỗi từ trong một document sao cho:

- từ xuất hiện nhiều trong một document (high TF) có trọng số lớn (quan trọng cho document đó)
- nhưng từ xuất hiện rất thường xuyên trên toàn bộ corpus (high DF) bị giảm trọng số (vì ít phân biệt document)
- **Ưu điểm**
 - Đơn giản, nhanh, hiệu quả cho nhiều tác vụ (text classification, retrieval).
 - Dễ hiểu, có thể giải thích (feature là từ/phrase).
 - Sparse: lưu trữ hiệu quả cho vocab lớn bằng sparse matrices.
 - Tốt cho mô hình tuyến tính (Naive Bayes, Logistic Regression, SVM).
- **Nhược điểm**
 - Không nắm bắt ngữ cảnh.
 - Không nắm bắt nghĩa sâu / polysemy / synonymy.

- Kích thước chiều cao.
- Nhạy cảm với corpus (thay đổi corpus có thể làm thay đổi features).
- Không tốt cho short text nếu document quá ngắn.

Word2Vec

Mục tiêu của Word2Vec là học các biểu diễn vector dày đặc (dense) và chiều thấp (300 chiều) cho các từ, nắm bắt ý nghĩa (ngữ nghĩa) của từ dựa trên ngữ cảnh chúng xuất hiện. Trong code thực hành, em đã dùng thư viện spacy với model ‘en_core_web_lg’

So sánh và kết luận

- Về Dimensionality
 - **BOW/TF-IDF**: chiều = kích thước vocab (ở đây 1867). Có thể tăng nhanh nếu thêm n-grams hoặc giảm `min_df`.
 - **Word2Vec**: chiều cố định do bạn chọn khi train hoặc dùng pretrained (ví dụ: 300). Ổn định, dễ kiểm soát.
- Về Sparsity & lưu trữ
 - **BOW/TF-IDF**: rất sparse (99.4%) — lợi cho lưu trữ bằng dạng sparse (CSR). Dạng sparse lưu trữ hiệu quả về bộ nhớ khi vocab lớn.
 - **Word2Vec (dense)**: chiếm bộ nhớ tuyển tính theo $n_{docs} \times \text{dim}$. Với dim = 300 và dataset nhỏ, chi phí lưu trữ thường thấp (vài MB). Tuy nhiên với rất nhiều document, biểu diễn dense có thể nặng hơn sparse.
- Về thông tin / semantic
 - **BOW**: chỉ dựa trên counts, không capture semantic.
 - **TF-IDF**: giảm trọng số từ phổ biến — tốt cho phân biệt document; cải thiện so với raw counts.
 - **Word2Vec (avg)**: nắm bắt semantic/relatedness giữa từ (ví dụ: synonymy). Việc lấy trung bình các vector làm mất cấu trúc thứ tự nhưng vẫn bắt được tương quan ngữ nghĩa hơn BOW.
- **Điễn giải (interpretability)**
 - **BOW/TF-IDF**: mỗi chiều mang ý nghĩa rõ ràng (mỗi dimension = 1 từ/ngram) — dễ giải thích.
 - **Word2Vec**: mỗi chiều không có ý nghĩa trực tiếp — khó giải thích hệ số mô hình (model explainability).

3.4 Phân tích & đánh giá kết quả

3.4.1 Phân tích kết quả định lượng

- Số lượng tài liệu: 1440

- BOW / TF-IDF:

- n_features = 1867 (kích thước từ vựng).
- nnz_total = 16,208 phần tử khác 0.
- Độ thưa (sparsity) 99.40% → rất thưa.
- Non-zero features per doc (mean, median) = 11.3, 11.0.
- Phân phối: phần lớn các tài liệu có khoảng 8–15 từ xuất hiện trong từ vựng (xem histogram BOW/TF-IDF).

- Word2Vec (document vectors, avg/agg → 300-dim):

- n_features = 300 (kích thước embedding).
- nnz_total = 432,000 = 1440 × 300 (mỗi giá trị đều khác 0).
- Sparsity = 0% → biểu diễn dense hoàn toàn.
- Non-zero features per doc = 300 (cố định cho mọi tài liệu).

Phân tích số liệu đặc trưng giữa các phương pháp biểu diễn

- Số lượng tài liệu: 1440 — đây là tổng số văn bản được sử dụng trong toàn bộ tập dữ liệu.

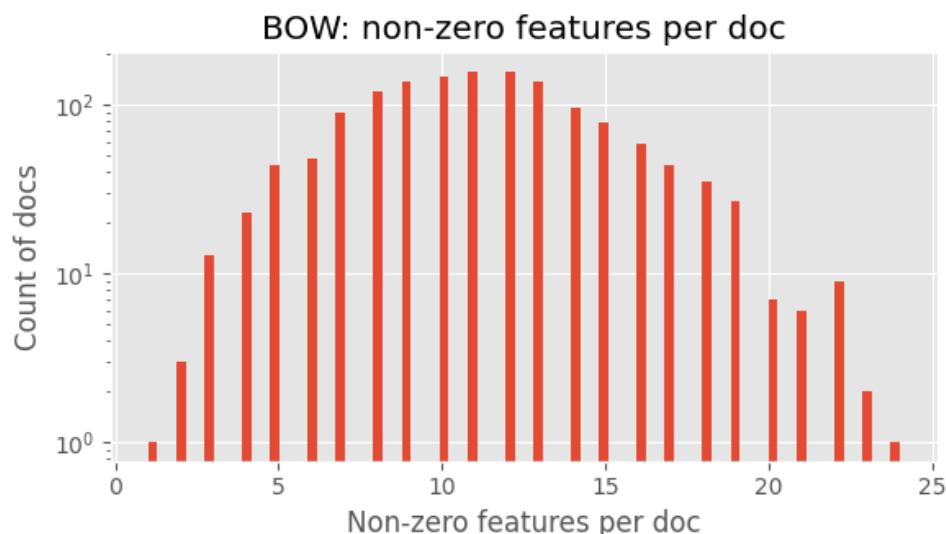
- BOW / TF-IDF:

- Có 1867 đặc trưng (từ trong vocab), thể hiện rằng từ vựng sau khi lọc (min_df, max_df) vẫn tương đối lớn so với số lượng tài liệu.
- Tổng số phần tử khác 0 là 16,208, nghĩa là trung bình mỗi tài liệu chỉ chứa một phần rất nhỏ trong toàn bộ không gian đặc trưng.
- Độ thưa (sparsity) đạt khoảng **99.40%**, cho thấy hầu hết các giá trị trong ma trận biểu diễn đều bằng 0 — đặc trưng cho BoW/TF-IDF.
- Số lượng đặc trưng khác 0 trung bình mỗi tài liệu là **11.3** (trung vị 11.0), nghĩa là một tài liệu trung bình chỉ chứa khoảng 11 từ khác nhau trong toàn bộ vocab.
- Phân phối tập trung trong khoảng **8–15 từ/tài liệu**, cho thấy các văn bản tương đối ngắn. Điều này đồng nghĩa với việc phần lớn tài liệu chỉ sử dụng một phần rất nhỏ trong vocab, dẫn đến ma trận biểu diễn rất thưa.

- Word2Vec (document vectors, trung bình/aggregate → 300 chiều):

- Kích thước đặc trưng cố định ở 300 — đây là số chiều embedding do mô hình định nghĩa.
- Tổng số phần tử khác 0 là $432,000 = 1440 \times 300$, nghĩa là toàn bộ các giá trị trong ma trận đều khác 0.
- **Sparsity = 0%**, thể hiện đây là biểu diễn **dense**, hoàn toàn khác so với BoW/TF-IDF.
- Mỗi tài liệu có đúng **300 đặc trưng**, không phụ thuộc vào độ dài hay số từ, giúp tạo ra không gian biểu diễn đồng nhất, thuận lợi cho các mô hình học sâu (Deep Learning).
- **Tổng kết:**
 - BoW/TF-IDF biểu diễn dữ liệu bằng ma trận thưa, có lợi cho các mô hình tuyến tính và tiết kiệm bộ nhớ khi dùng dạng lưu trữ sparse.
 - Word2Vec tạo ra ma trận dense kích thước nhỏ hơn, chứa thông tin ngữ nghĩa sâu hơn và phù hợp với các mô hình phi tuyến (Deep Neural Networks).
 - Sự khác biệt chính nằm ở tính **sparse** vs. **dense**, **chiều cố định** vs. **phụ thuộc vocab**, và khả năng **biểu diễn ngữ nghĩa**.

3.4.2 So sánh trực quan và diễn giải kết quả



Hình 3.1 BOW: non-zero features per docs

Nhận xét về Bag-of-Words (BOW)

- **Phân bố đặc trưng:**
 - Phân bố có dạng gần đối xứng (gần Gaussian), với đỉnh nằm ở khoảng 10–13 đặc trưng/tài liệu.

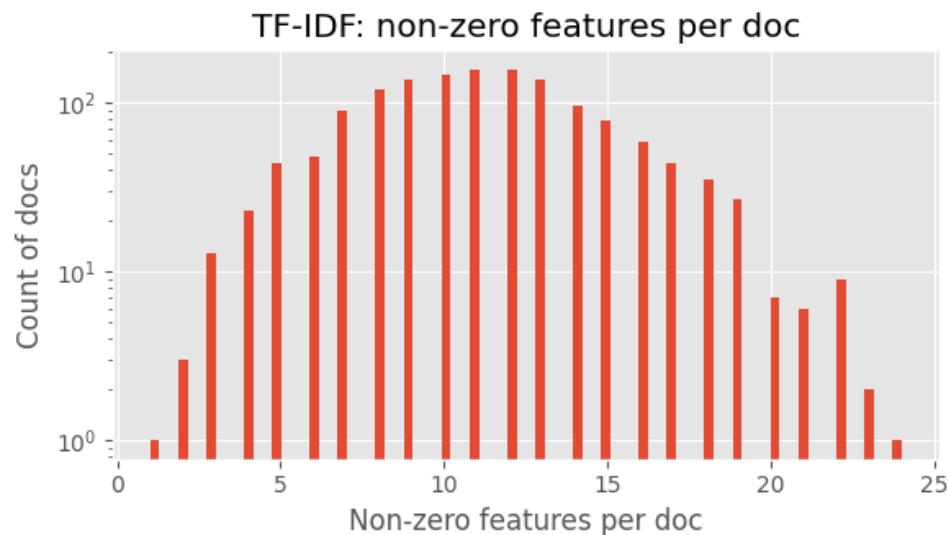
- Điều này cho thấy phần lớn các tài liệu chỉ chứa từ 10–13 từ duy nhất sau khi được vector hóa bằng BoW.

- **Hai đầu phân bố:**

- Rất ít tài liệu có dưới 3 đặc trưng hoặc trên 20 đặc trưng.
- Cho thấy độ dài trung bình của các văn bản tương đối ngắn (vì BoW chỉ đếm số từ khác 0, không xét tần suất).

- **Độ thưa (sparsity) của ma trận BoW:**

- Mỗi tài liệu chỉ kích hoạt một số nhỏ trong toàn bộ số đặc trưng của từ vựng.
- Đây là hiện tượng bình thường, do từ vựng tổng thể thường rất lớn (hàng nghìn từ hoặc hơn), trong khi mỗi tài liệu chỉ chứa một lượng từ hạn chế.



Hình 3.2 TF-IDF: non-zero features per docs

Nhận xét về TF-IDF

- **Phân bố đặc trưng:**

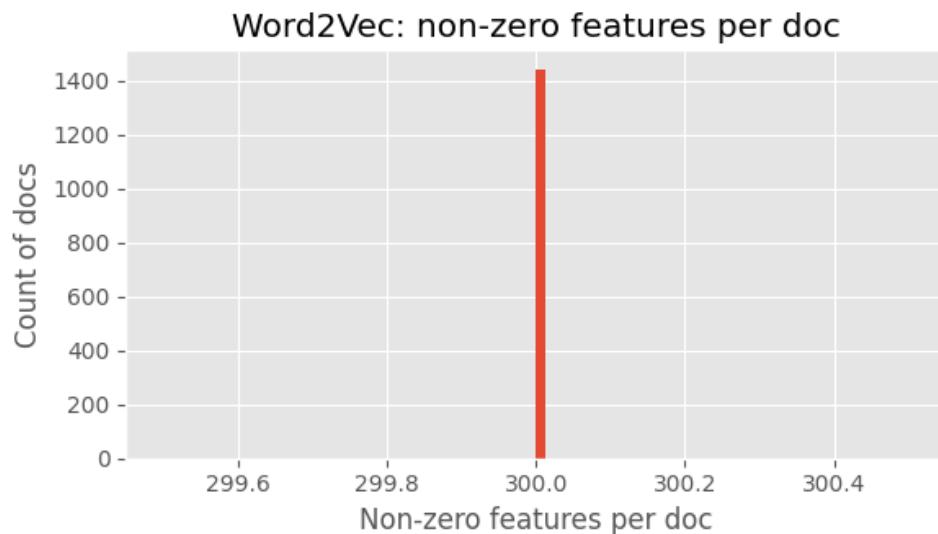
- Phân bố rất giống với BoW, với đỉnh nằm ở khoảng 10–13 đặc trưng/tài liệu.
- Điều này nghĩa là phần lớn tài liệu chỉ chứa khoảng 10–13 từ duy nhất sau khi được vector hóa bằng TF-IDF.

- **Hai đầu phân bố:**

- Rất ít tài liệu có dưới 3 hoặc trên 20 từ đặc trưng.
- Các tài liệu cực ngắn hoặc cực dài là hiếm gặp.

- **Kết luận:**

- TF-IDF không làm thay đổi số lượng đặc trưng khác 0, mà chỉ thay đổi giá trị của chúng (theo trọng số tầm quan trọng toàn cục).
- Do đó, phân bố của số “non-zero features per document” giữa BoW và TF-IDF gần như trùng khớp — thể hiện rõ qua hai biểu đồ.



Hình 3.3 Word2Vec: non-features per docs

Nhận xét về Word2Vec

- **Phân bố đặc trưng:**

- Chỉ có một thanh duy nhất tại giá trị 300.
- Mỗi tài liệu sau khi được biểu diễn bằng Word2Vec đều có 300 đặc trưng không bằng 0 (tức là vector đặc trưng đầy đủ, dense vector).

- **So sánh với BoW và TF-IDF:**

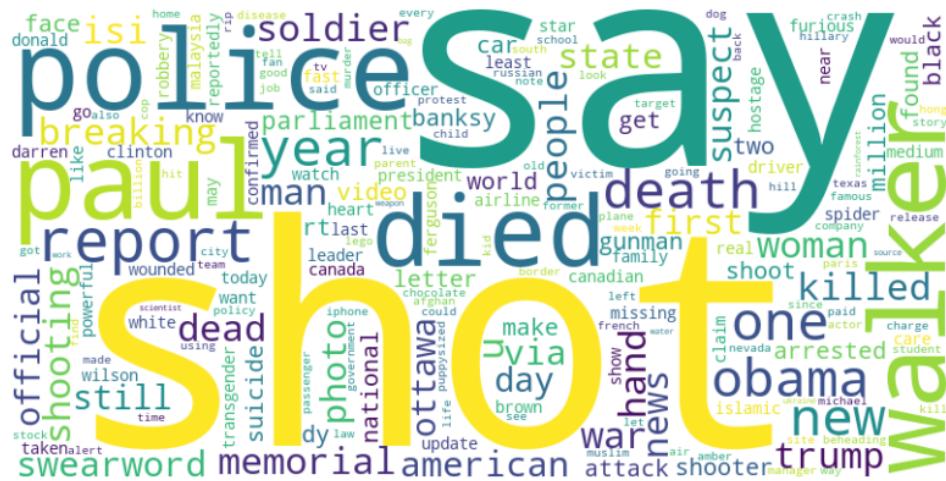
- Hoàn toàn khác biệt, vì ở BoW và TF-IDF, số đặc trưng khác 0 phụ thuộc vào độ dài và từ vựng của từng tài liệu.

- **Kết luận:**

- Tất cả các tài liệu đều có cùng số chiều vector (300).
- Không có sự thay đổi về độ thưa (sparsity) giữa các tài liệu — toàn bộ không gian biểu diễn là dense.

Word Cloud

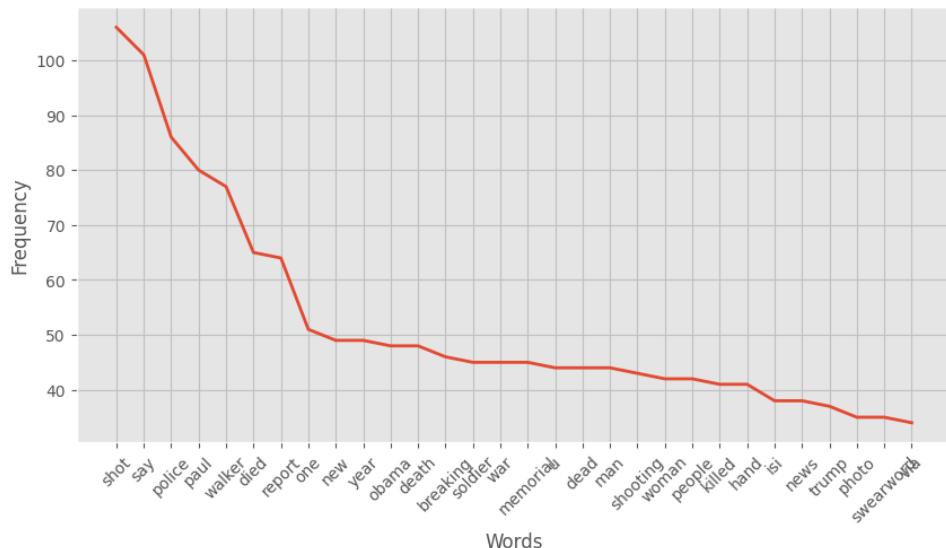
- Word Cloud sẽ cung cấp hình ảnh trực quan về các từ xuất hiện thường xuyên nhất trong văn bản, với các từ lớn hơn biểu thị tần suất cao hơn.
 - Bằng cách tạo word cloud, ta có thể nhanh chóng nắm bắt các chủ đề chính và các thuật ngữ nổi bật có trong tập dữ liệu hoặc tài liệu.



Hình 3.4 Word Cloud

Hình trên cho biết từ shot suất hiện lớn nhất sau đó là police. Điều này cho biết nói về các cuộc xả súng, tội phạm, cảnh sát, ...

Word Frequency



Hình 3.5 Word Frequency

Thông qua hình trên 3.5, ta có thể thấy tần suất xuất hiện của từ "Shot" là vô cùng cao. Dẫn đến ta biết được hầu hết các tin tức đều nói đến vấn đề xả súng, cảnh sát, tội phạm, chính trị, ... trên thế giới.

3.4.3 Phân tích tác động về chất lượng dữ liệu

Bộ dữ liệu này đã được làm sạch ở mức độ từng token, đồng thời áp dụng các phương pháp stemming và lemmatization khác nhau để so sánh hiệu quả. Cuối cùng, các biểu diễn văn bản như Bag-of-Words, TF-IDF và Word2Vec đã được tạo ra để phục vụ cho các tác vụ phân tích và mô hình hóa ngôn ngữ tự nhiên tiếp theo. Bộ dữ liệu này cũng đã được trực quan hóa và phân tích kết quả về độ sạch.