TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN - TIN HỌC

BÁO CÁO THỰC HÀNH CUỐI KỲ

Môn: Xử lí Ngôn ngữ Tự nhiên

Đề tài: Phát hiện Tin giả (Fake News Detection)

Thành viên nhóm

Võ Thành Đạt – 22280010
Vũ Ngọc Phương – 22280072
Phạm Minh Thái – 22280082

TP. Hồ Chí Minh, tháng 6 năm 2025

I. Giới thiệu về bộ dữ liệu

Bộ dữ liệu được sử dụng trong bài toán này bao gồm các bài báo thật và giả, phục vụ cho mục tiêu xây dựng một mô hình phát hiện và phân loại tin giả.

Bộ dữ liệu bao gồm hai phần:

- **MisinfoSuperset_TRUE.csv**: Chứa các bài báo thật được thu thập từ những nguồn tin uy tín như *Reuters*, *The New York Times*, *The Washington Post*, v.v.
- **MisinfoSuperset_FAKE.csv**: Gồm các bài viết mang nội dung giả mạo, tuyên truyền hoặc xuyên tạc, được tổng hợp từ các trang cực đoan cánh hữu của Mỹ (ví dụ: *Redflag Newsdesk*, *Breitbart*, *Truth Broadcast Network*) và từ một tập dữ liệu công khai của Ahmed et al. (2017).

Thông qua việc áp dụng các kỹ thuật học máy, học sâu và các mô hình hiện đại như Transformer, bài toán hướng đến xây dựng một hệ thống phát hiện tin giả một cách hiệu quả. Hệ thống này sẽ hỗ trợ các nền tảng truyền thông, cơ quan báo chí và người dùng cá nhân trong việc nhận diện và tiếp cận thông tin đáng tin cậy.

II. Model Development

1. Data Cleaning

a) Gộp, làm sạch và gán nhãn

- Bộ dữ liệu ban đầu được chia thành hai tệp riêng biệt và không chứa biến nhãn phân loại (`True/Fake`). Do đó, cần tạo thêm biến mục tiêu thể hiện nhãn và gộp hai bộ dữ liệu thành một tập hợp thống nhất.
- Biến `Unnamed: 0` chỉ đơn thuần là chỉ số dòng (index), không mang giá trị thông tin cho bài toán nên sẽ được loại bỏ.
- Trong bộ dữ liệu thật (TRUE), có 29 quan sát bị thiếu giá trị ở biến `text` — đây cũng là biến duy nhất trong tập dữ liệu và ở dạng chuỗi (object), nên không thể thay thế hợp lý. Do đó, các quan sát này sẽ bị loại bỏ.
- Sau khi hợp nhất hai tệp dữ liệu và loại bỏ biến `Unnamed: 0`, tổng số lượng quan sát bị trùng lặp là 10,012.

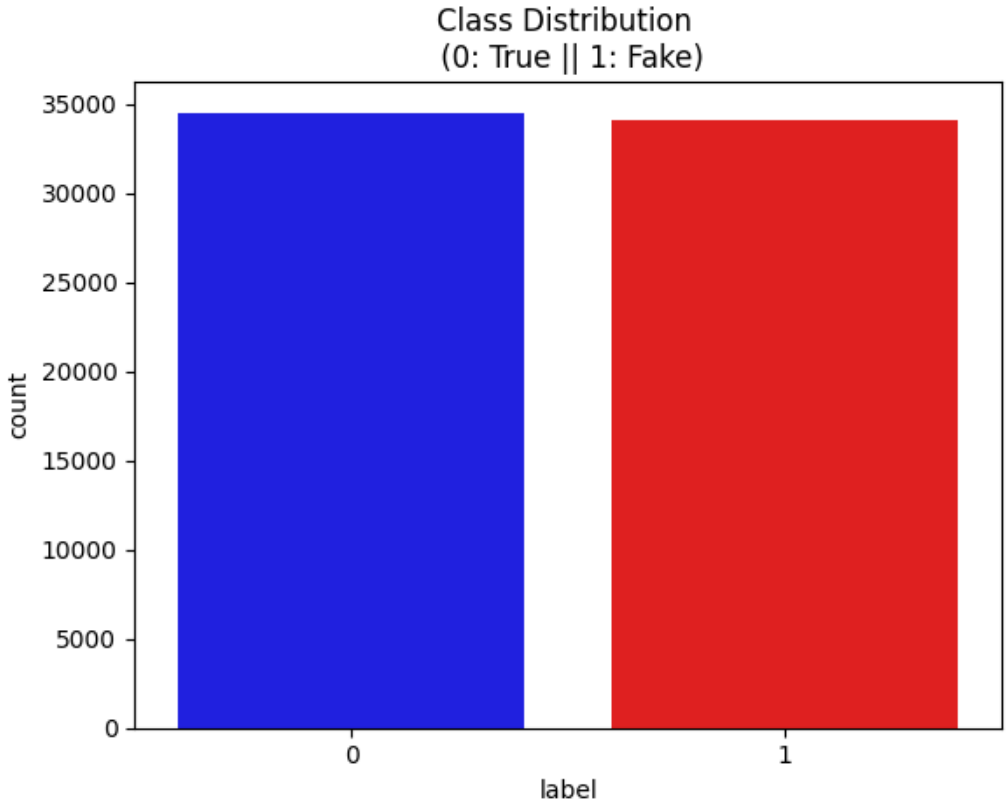
b) Loại bỏ các bản ghi trùng lặp

Sau khi loại bỏ các quan sát bị thiếu dữ liệu, số lượng bản ghi bị trùng lặp còn lại là 9,984.

Qua kiểm tra chi tiết, không có trường hợp nào mà một đoạn văn bản (`text`) giống nhau nhưng được gán nhãn khác nhau (vừa là tin thật, vừa là tin giả). Do đó, ta có thể loại bỏ toàn bộ các bản ghi trùng lặp này một cách an toàn, mà không cần xử lý thêm.

2. Exploratory Data Analysis

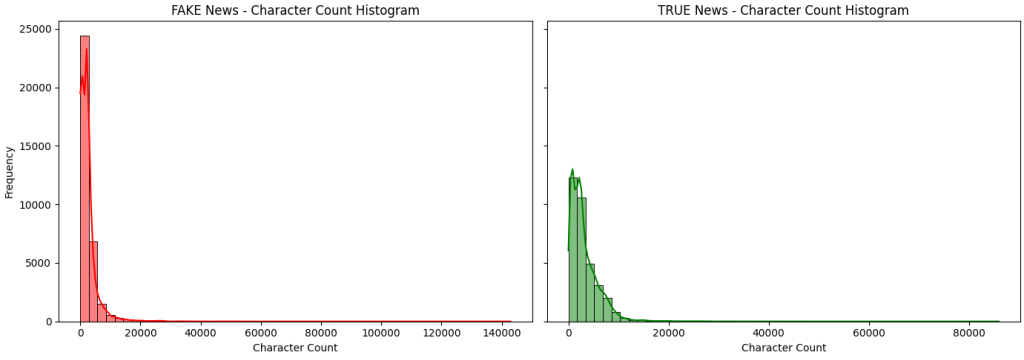
a) Phân phối nhãn



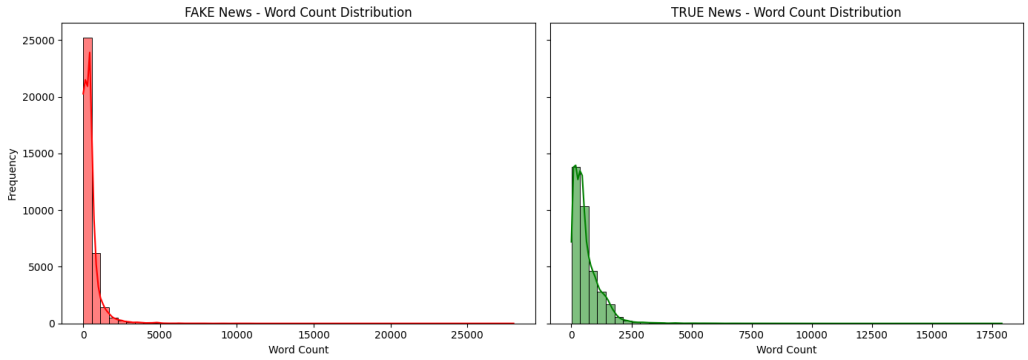
Hình 1: Phân bố nhãn trong tập dữ liệu

Dựa vào biểu đồ trên, ta nhận thấy hai lớp `True` (tin thật) và `Fake` (tin giả) gần như cân bằng về số lượng nên ta không cần xử lí gì thêm.

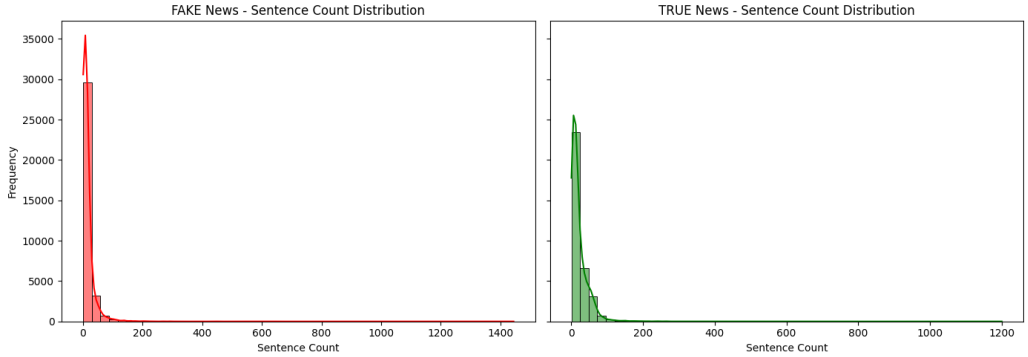
b) Phân tích độ dài văn bản



Hình 2: Số ký tự trong mỗi văn bản



Hình 3: Số từ trong mỗi văn bản



Hình 4: Số câu trong mỗi văn bản

Qua các biểu đồ trên, ta rút ra một số nhận xét quan trọng:

- Trung bình mỗi văn bản có độ dài lớn (khoảng 552 từ) nhưng phân phối không ổn định, với độ lệch chuẩn cao (xấp xỉ 700).
- Cả hai lớp `True` (tin thật) và `Fake` (tin giả) đều chứa những văn bản có số từ, câu hoặc ký tự rất nhỏ hoặc rất lớn — đây được xem là các **outlier**.
- Có sự khác biệt đáng kể về trung bình số từ, câu và ký tự giữa hai lớp. Điều này có thể là do ảnh hưởng của các outlier kéo lệch phân phối.
- Nếu không xử lý các outlier này, mô hình có thể học cách phân loại dựa trên đặc trưng hình thức như độ dài văn bản thay vì nội dung. Hệ quả là mặc dù mô hình có thể cho kết quả tốt khi đánh giá trên tập validation, nhưng sẽ không hoạt động ổn định trong môi trường triển khai thực tế.

Kết luận: Cần thực hiện các bước xử lý bao gồm loại bỏ các **outlier cực đoan** và **chuẩn hóa độ dài** các văn bản đầu vào trước khi huấn luyện mô hình.

c) Phân tích từ vựng

Qua các biểu đồ trong phần code, ta rút ra một số đặc điểm khác biệt giữa hai lớp `True` và `Fake`:

- **Emoji** xuất hiện rất ít nhưng đa dạng, tạo ra độ nhiễu cao và gây phức tạp khi xử lý.
- **Tin giả** có **TTR (độ đa dạng từ vựng)** cao hơn, có thể do xu hướng dùng nhiều từ thu hút, giật gân.

- **Stopwords** xuất hiện nhiều hơn trong tin thật (231.1 vs 188.4), phản ánh phong cách viết mô tả, tự nhiên hơn.
- **Dấu câu** cũng xuất hiện nhiều hơn trong tin thật (71.8 vs 60.3), thể hiện cấu trúc ngữ pháp phức tạp, chính quy hơn.
- **Từ viết IN HOA** và **dấu chấm than** xuất hiện nhiều hơn trong tin giả (8.34 vs 7.46 và 0.6526 vs 0.1398), cho thấy xu hướng nhấn mạnh, kích động cảm xúc.
- **URL/HTML** phổ biến hơn trong tin giả (0.109 vs 0.0014), nhiều khả năng dùng để đánh lừa hoặc dẫn tới nội dung độc hại.

Kết luận: Ngay cả khi chưa xét đến nội dung, hai lớp đã có nhiều đặc trưng hình thức khác nhau. Do đó, mô hình hoàn toàn có cơ sở để học và phân biệt hiệu quả.

d) Phân tích N-gram và Từ vựng đặc trưng

	top_real_unigrams	top_fake_unigrams	top_real_bigrams	top_fake_bigrams
0	(said, 166497)	(trump, 89391)	(united states, 20181)	(donald trump, 19284)
1	(trump, 96925)	(people, 38937)	(mr trump, 17402)	(hillary clinton, 12507)
2	(mr, 65801)	(said, 37327)	(donald trump, 16040)	(united states, 9241)
3	(president, 47573)	(clinton, 33690)	(white house, 12738)	(featured image, 7877)
4	(new, 38674)	(president, 29880)	(new york, 11294)	(white house, 7298)
5	(people, 38456)	(just, 27747)	(president donald, 6744)	(twitter com, 5817)
6	(state, 35455)	(like, 25037)	(hillary clinton, 5807)	(pic twitter, 5509)
7	(states, 28632)	(hillary, 23216)	(islamic state, 5774)	(new york, 5311)
8	(government, 26715)	(new, 21490)	(north korea, 5676)	(president obama, 4266)
9	(clinton, 26475)	(donald, 20914)	(trump said, 5319)	(getty images, 4135)

Hình 5: Top Unigram và Bigram trong Tin Thật và Tin Giả

```
display(pd.DataFrame(unique_stats))
```

	unique_real_words	unique_fake_words
0	yermolov	уйдёт
1	mcevoy	citys
2	fader	operam
3	nami	dmh
4	willowy	volo
5	settee	navales
6	bartman	bayrakları
7	fotini	acostumbrados
8	tarrell	igot
9	bluebay	caisson
10	aeolian	instancias
11	microsystems	tacheles
12	aguero	websiteread
13	تشویش	kritische
14	dramatizes	outhillaryhey
15	léger	coinvolgimento
16	carandiru	tbart
17	natrell	ouragan
18	ambroise	empirelast
19	freedman	отмечается

Hình 6: Các từ chỉ xuất hiện trong Tin Thật và Tin Giả

Nhận xét:

- Các **unigram phổ biến** trong cả hai lớp khá tương đồng, ngoại trừ từ `said` chiếm ưu thế trong tin thật, phản ánh văn phong trung lập và trích dẫn của báo chí truyền thống.
- Trong **bigrams**, sự khác biệt rõ rệt hơn:
  - Tin thật xuất hiện nhiều cụm danh từ chính thống như *mr trump*, *white house*, *president donald*, thể hiện tính chính xác và phong cách báo chí chuẩn mực.
  - Tin giả lại chứa nhiều cụm như *featured image*, *pic twitter*, *getty images*, liên quan đến hình ảnh, mạng xã hội, hoặc nội dung không kiểm chứng – dấu hiệu đặc trưng của tin lan truyền.
- Với **các từ chỉ xuất hiện riêng**, tin thật thường chứa tên riêng, thuật ngữ lịch sử hoặc từ quốc tế, thể hiện nội dung sâu và đa dạng. Trong khi đó, tin giả lại xuất hiện nhiều từ cảm thán, tiếng lóng mạng hoặc từ ngữ không chuẩn, đôi khi từ các ngôn ngữ khác không phổ biến – cho thấy tính không chính thống, giật gân hoặc dịch sai nguồn.

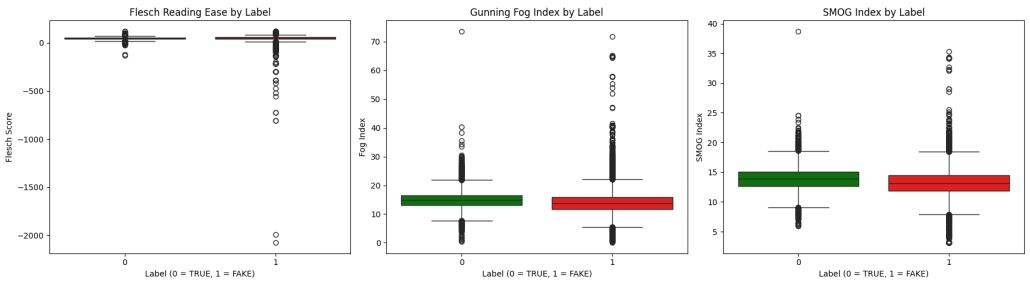
e) Phân tích độ dễ đọc

Bảng 1: Thống kê mô tả độ dễ đọc của tin thật (Class = True)

Thống kê	Flesch Reading	Gunning Fog	SMOG Index
Số lượng (count)	34,526	34,526	31,998
Trung bình (mean)	45.77	14.89	13.87
Độ lệch chuẩn (std)	11.44	2.89	1.87
Giá trị nhỏ nhất (min)	-132.58	0.40	5.99
Phân vị 25% (Q1)	38.83	13.05	12.66
Trung vị (median)	45.81	14.81	13.89
Phân vị 75% (Q3)	52.93	16.57	15.05
Giá trị lớn nhất (max)	121.22	73.65	38.73

Bảng 2: Thống kê mô tả độ dễ đọc của tin giả (Class = Fake)

Thống kê	Flesch Reading	Gunning Fog	SMOG Index
Số lượng (count)	34,078	34,078	27,970
Trung bình (mean)	48.58	13.98	13.19
Độ lệch chuẩn (std)	26.85	3.89	2.24
Giá trị nhỏ nhất (min)	-2078.38	0.00	3.13
Phân vị 25% (Q1)	40.79	11.74	11.86
Trung vị (median)	50.16	13.76	13.12
Phân vị 75% (Q3)	58.44	15.90	14.50
Giá trị lớn nhất (max)	121.22	71.83	35.28



Hình 7: Boxplot độ dễ đọc theo từng lớp

Nhận xét:

- Cả hai lớp `True` và `Fake` đều có trung bình các chỉ số `flesch_reading`, `gunning_fog` và `smog_index` tương đối gần nhau, cho thấy mức độ dễ hiểu trung bình là tương đương.
- Tuy nhiên, **tin thật** có phân phối tập trung hơn, ít outlier và ổn định hơn. Nếu loại bỏ các outlier (ví dụ: `gunning_fog` > 70 hoặc `smog_index` > 35), các chỉ số trung bình của lớp này sẽ còn gần hơn với phân phối chuẩn — phản ánh chất lượng ngôn ngữ tốt và nhất quán.
- Trong khi đó, **tin giả** xuất hiện nhiều outlier, thậm chí cực trị với điểm `flesch_reading` âm (thấp tới -2000), cho thấy một số văn bản cực kỳ khó đọc. Điều này có thể do lỗi ngắt câu, lỗi dịch thuật, font chữ, hay cấu trúc câu bất thường — phản ánh chất lượng ngôn ngữ kém, đúng với đặc điểm thường gặp của tin giả.

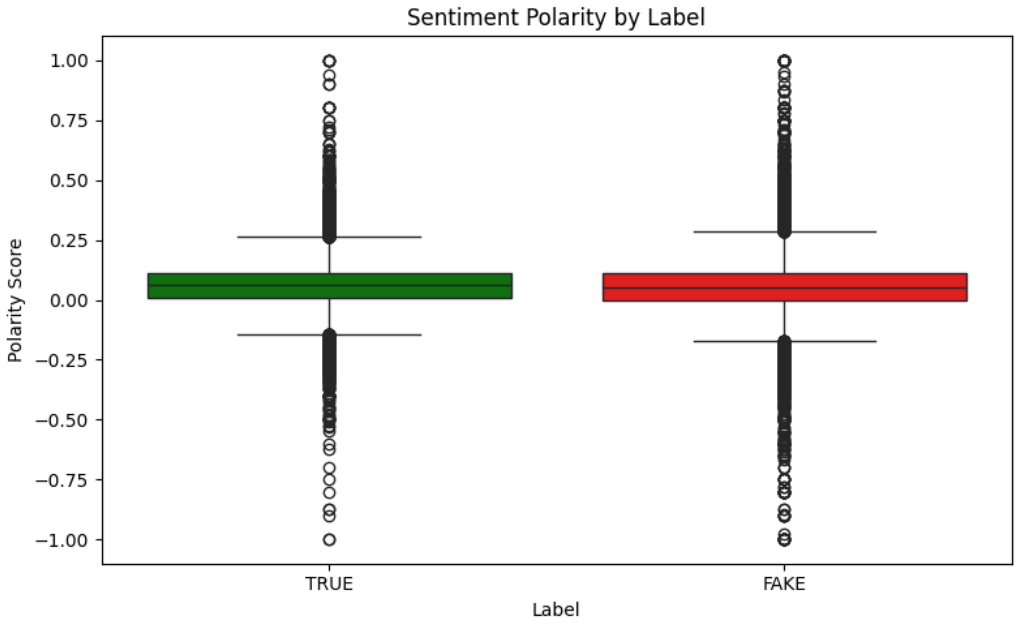
f) Phân tích cảm xúc (Sentiment Analysis)

Bảng 3: Thống kê mô tả chỉ số cảm xúc của tin thật (Class = True)

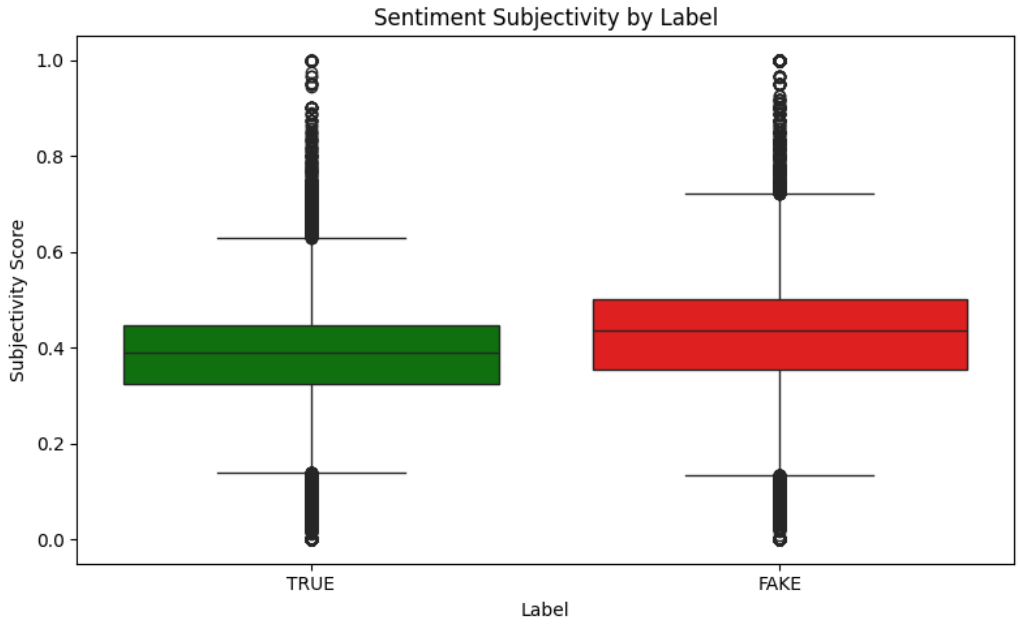
Thống kê	**Polarity**	**Subjectivity**
Số lượng (count)	34,526	34,526
Trung bình (mean)	0.0613	0.3803
Độ lệch chuẩn (std)	0.0942	0.1142
Giá trị nhỏ nhất (min)	-1.0000	0.0000
Phân vị 25% (Q1)	0.0096	0.3236
Trung vị (median)	0.0618	0.3887
Phân vị 75% (Q3)	0.1104	0.4462
Giá trị lớn nhất (max)	1.0000	1.0000

Bảng 4: Thống kê mô tả chỉ số cảm xúc của tin giả (Class = Fake)

Thống kê	**Polarity**	**Subjectivity**
Số lượng (count)	34,078	34,078
Trung bình (mean)	0.0544	0.4112
Độ lệch chuẩn (std)	0.1308	0.1690
Giá trị nhỏ nhất (min)	-1.0000	0.0000
Phân vị 25% (Q1)	0.0000	0.3538
Trung vị (median)	0.0507	0.4353
Phân vị 75% (Q3)	0.1142	0.5007
Giá trị lớn nhất (max)	1.0000	1.0000



Hình 8: Boxplot Polarity theo từng lớp



Hình 9: Boxplot Subjectivity theo từng lớp

Nhận xét:

- Cả hai lớp `Fake` và `True` đều có trung bình chỉ số `Polarity` (mức độ cảm xúc tích cực hoặc tiêu cực) gần bằng 0, phản ánh rằng phần lớn các bài viết đều có xu hướng trung tính.
- Tuy nhiên, lớp **tin thật (True)** có phân phối `Polarity` tập trung hơn, ít outlier ở hai cực -1 và 1, trong khi **tin giả (Fake)** lại có nhiều văn bản mang cảm xúc cực đoan hơn, thể hiện qua các outlier nằm ở cả hai phía.
- Về chỉ số `Subjectivity`, các tin thật có mức độ chủ quan thấp hơn so với tin giả và có xu hướng tập trung quanh trung bình. Điều này phù hợp với đặc trưng của báo chí chính thống, thường có xu hướng khách quan và trung lập. Ngược lại, tin giả thường mang tính chủ quan cao hơn nhằm kích thích cảm xúc người đọc.

3. Data Processing

Dựa trên các phân tích ở các mục trước, ta đề xuất các bước xử lý dữ liệu như sau:

- **Loại bỏ các extreme outlier** để cân bằng lại số chữ, từ và câu giữa hai lớp, tránh hiện tượng mô hình học theo đặc trưng hình thức thay vì nội dung.
- **Loại bỏ các văn bản có chỉ số Readability quá bất thường** nhằm đưa phân phối độ dễ đọc của cả hai lớp về gần phân phối chuẩn, đảm bảo tính ổn định trong huấn luyện.
- **Chuẩn hóa độ dài văn bản** bằng cách cắt/truncate hoặc padding, giúp mô hình học máy xử lý dữ liệu đồng nhất.
- **Giữ nguyên định dạng chữ hoa/thường** (không chuyển thành lowercase), vì các từ viết in hoa mang ý nghĩa nhấn mạnh đặc trưng trong tin giả.
- **Không loại bỏ stopwords**, vì tần suất xuất hiện của chúng mang đặc trưng ngữ pháp khác biệt giữa hai lớp.

- **Giữ lại dấu chấm than và dấu câu**, do chúng phản ánh rõ phong cách hành văn (giật gân, trung lập, mô tả...).
- **Không xóa URL/HTML**, thay vào đó tiến hành **mã hóa (encode)** vì đây là tín hiệu quan trọng phân biệt tin giả.

4. Model Development and Comparison

Bài toán đặt ra là một bài toán **phân loại nhị phân**, với hai nhãn: **Fake** (tin giả) và **True** (tin thật). Để giải quyết bài toán này, chúng em đã thử nghiệm với nhiều nhóm mô hình khác nhau, bao gồm:

- **Mô hình học máy truyền thống (Traditional ML):** Logistic Regression, Naive Bayes, SVM
- **Mô hình mạng nơ-ron hồi tiếp (RNN):** LSTM
- **Mô hình hiện đại dựa trên Transformer:** BERT, XLNet, DeBERTa-v3

Bảng 5: So sánh hiệu suất các mô hình trên tập kiểm tra (test set)

Mô hình	**Accuracy**	**Precision**	**Recall**	**F1-score**
Naive Bayes	0.8872	0.9028	0.8666	0.8843
Logistic Regression	0.9479	0.9470	0.9485	0.9477
LSTM	0.9447	0.9372	0.9524	0.9447
BERT	0.9910	0.9935	0.9883	0.9909
SVM	0.9177	0.9153	0.9196	0.9175
XLNet	0.9954	0.9940	0.9957	0.9949
DeBERTa	0.9976	0.9986	0.9960	0.9973

Tùy theo **mục tiêu tối ưu hóa** trong quá trình triển khai, ta có thể lựa chọn mô hình phù hợp:

a) Nếu mục tiêu chính là **rút ngắn thời gian huấn luyện** và **tiết kiệm tài nguyên tính toán**, thì mô hình **Naive Bayes** là lựa chọn hợp lý. Với thời gian train nhanh nhất trong tất cả các mô hình và yêu cầu cấu hình phần cứng thấp, Naive Bayes đặc biệt hiệu quả trong các hệ thống hạn chế về tài nguyên hoặc cần triển khai nhanh.

b) Nếu mục tiêu là đạt được **tính diễn giải cao**, kết quả huấn luyện **ổn định** và thời gian huấn luyện **khá nhanh**, thì **Logistic Regression** là lựa chọn phù hợp. Mô hình này cho phép giải thích được ảnh hưởng của từng đặc trưng đầu vào, rất hữu ích trong các ứng dụng yêu cầu tính minh bạch.

c) Nếu mục tiêu là đạt được **độ chính xác dự đoán cao nhất** và **không bị giới hạn bởi tài nguyên tính toán**, thì các mô hình **Transformer hiện đại** (như BERT, XLNet, DeBERTa) là lựa chọn tối ưu. Những mô hình này có khả năng học sâu ngữ cảnh và quan hệ giữa các từ, giúp cải thiện rõ rệt hiệu suất phân loại.

Nhận xét: Các mô hình Transformer (DistilBert, DeBERTa và XLNet) cho kết quả vượt trội trên tập test, nhưng đòi hỏi tài nguyên tính toán lớn và thời gian huấn luyện lâu. Trong khi đó, Logistic Regression vẫn đạt hiệu quả cao, dễ triển khai và dễ diễn giải, phù hợp với các hệ thống sản xuất yêu cầu tốc độ và sự minh bạch.

5. Analyze model

Naive Bayes

Điểm yếu

- Do mô hình Naive Bayes giả định tính độc lập giữa các đặc trưng, điều này không phản ánh đúng đặc trưng trong ngôn ngữ dẫn đến **hiệu suất thấp nhất** trong các model.
- Phải vector hóa dựa trên ma trận TF-IDF nên bị phụ thuộc nặng vào vocabulary của tập train. Nếu có từ mới, ngôn ngữ khác thì không thể tính toán tốt, sẽ bị bỏ qua hoặc gán trọng số 0, gây mất thông tin.
- Naive Bayes vượt trội ở tốc độ train và inference nhưng chắc chắn phải đánh đổi lại tính chính xác không bằng các phương pháp khác.

Cách cải thiện

- Kết hợp Naive Bayes với các feature bổ sung như bigram/trigram để giảm giả định độc lập hoàn toàn. Tuy giảm tính độc lập giả định, nhưng làm tăng chiều dữ liệu, phải đánh đổi tốc độ.
- Dùng kỹ thuật ensemble như VotingClassifier để kết hợp Naive Bayes với các mô hình khác.
- Tuy có thể train Naive Bayes với ma trận TF-IDF khi có từ mới, ngôn ngữ mới hoặc kết hợp với các model khác để tăng tính chính xác, nhưng đó lại làm mất đi điểm mạnh là tốc độ của Naive Bayes. Đồng thời, mỗi lần có dữ liệu mới thì lại phải train từ đầu, không linh hoạt. Ta nên cân nhắc đổi qua mô hình khác khi bài toán trở nên phức tạp hơn.

Logistic Regression

Điểm yếu

- Chỉ có thể học được các mối quan hệ tuyến tính, không học được các quan hệ phi tuyến phức tạp.
- Dù có thể học quan hệ phi tuyến gián tiếp qua biến đổi đặc trưng (polynomial), khả năng vẫn kém hơn mô hình non-linear (NN, SVM kernel).
- Phải vector hóa dựa trên ma trận TF-IDF nên bị phụ thuộc nặng vào vocabulary của tập train. Nếu có từ mới, ngôn ngữ khác thì không thể tính toán được.

Cách cải thiện

- Bổ sung **polynomial features** nhưng cần cẩn thận với overfitting do chiều dữ liệu tăng. Nên kết hợp regularization mạnh (L1/L2).
- Kết hợp với các **đặc trưng ngữ nghĩa** như sentiment score, entity recognition. Tuy nhiên, các đặc trưng này cần được trích xuất từ công cụ bên ngoài (spaCy, Sentiment Analyzer...) sẽ tăng độ phức tạp hệ thống.
- Dùng embedding thay vì TF-IDF để giảm chiều dữ liệu và xử lý từ mới tốt hơn.
- Tuy có thể train Logistic Regression với ma trận TF-IDF khi có từ mới, ngôn ngữ mới hoặc kết hợp với các đặc trưng ngữ nghĩa hoặc polynomial feature để tăng tính chính xác, nhưng nếu vocab quá lớn, số lượng biến của mô hình tăng theo cấp số nhân, dẫn tới quá nặng và hiệu quả chưa chắc bằng với việc sử dụng các mô hình khác. Vì vậy, ta nên cân nhắc đổi qua mô hình khác khi bài toán trở nên phức tạp hơn.

SVM

Điểm yếu

- LinearSVM chỉ phù hợp khi dữ liệu có thể phân tách tuyến tính tốt.
- Không tự động trích xuất đặc trưng như các mô hình deep learning nên phụ thuộc hoàn toàn vào chất lượng biểu diễn đầu vào (TF-IDF).
- Không hỗ trợ "early stopping" trong huấn luyện vì không phải là mô hình huấn luyện theo epoch như neural networks.
- Hiệu suất có thể kém hơn trên các văn bản ngắn, nhiễu hoặc có thông tin ngữ cảnh sâu mà TF-IDF không thể nắm bắt.

Cách cải thiện

- Sử dụng các kernel phi tuyến như RBF hoặc polynomial để mô hình hóa các quan hệ phức tạp hơn.
- Kết hợp TF-IDF với đặc trưng bổ sung như n-gram, topic modeling, sentiment scores để bổ sung thông tin ngữ nghĩa.
- Thử nghiệm với các cách giảm chiều dữ liệu khác như PCA, cũng như tăng lượng thông tin giữ lại để model có thể nắm bắt tốt hơn.

LSTM

Điểm yếu

- Tính diễn giải thấp.
- Thời gian train vẫn cao hơn mô hình truyền thống.
- Không hoạt động tốt với những văn bản quá dài (LSTM vẫn bị vanishing gradient với chuỗi rất dài (>100 tokens)) vì kiến trúc của LSTM cũng chỉ cải thiện hơn việc "nhớ" ngắn hạn của RNN truyền thống chứ không trực tiếp tính toán bằng thông tin của mọi vị trí như các mô hình Transformer-based với Attention.

Cách cải thiện

- Sử dụng **pretrained Word2Vec từ corpus lớn hơn** (như Google News) hoặc thử nghiệm **fastText** để biểu diễn từ hiệu quả hơn.
- Bổ sung **attention layer** để mô hình tập trung vào các từ quan trọng trong câu.
- Có thể dùng hierarchical LSTM cho văn bản dài (chia thành đoạn nhỏ).
- Với bài toán phức tạp, nên chuyển sang Transformer-based models (hiệu quả hơn với ngữ cảnh dài).

DistilBERT

Điểm yếu

- Dù nhẹ hơn BERT đầy đủ, DistilBERT vẫn rất lớn (66M param) và mất nhiều thời gian huấn luyện và dự đoán kết quả. DistilBERT vẫn cần GPU để inference nhanh trong production.
- Tính diễn giải của mô hình thấp.

Cách cải thiện

- Tối ưu inference bằng cách chuyển mô hình sang **ONNX** hoặc áp dụng **quantization** (nhưng giảm độ chính xác (float32 → float16/int8) → giảm 50–75% bộ nhớ).
- Dùng **LIME, SHAP hoặc explain_transformers** để giải thích quyết định mô hình.
- Thử nghiệm với Bert bé hơn như **Tiny Bert** để giảm thời gian huấn luyện, tuy nhiên điều này cũng có thể đánh đổi rằng kết quả đầu ra của mô hình có thể giảm.
- Chỉ fine-tune lớp top layers để tiết kiệm thời gian.

XLNet

Điểm yếu

- **Mô hình phức tạp và nặng hơn**: XLNet có kiến trúc transformer với cơ chế hoán vị ngữ cảnh nên cần nhiều tài nguyên tính toán (thời gian huấn luyện lâu hơn, sử dụng nhiều RAM/VRAM).
- **Thời gian huấn luyện và suy diễn chậm hơn**: Khi triển khai trên hệ thống thực tế hoặc cần huấn luyện lại, chi phí tính toán và thời gian tăng đáng kể so với các mô hình nhẹ hơn như Logistic Regression hay SVM.
- **Cần xử lý dữ liệu đầu vào cẩn thận**: Độ dài văn bản cần cắt gọn hợp lý (max_length=512) và đòi hỏi bộ tokenizer riêng để đảm bảo chất lượng đầu vào phù hợp.
- Tính diễn giải của mô hình thấp.

Cách cải thiện

- Không fine-tune toàn bộ XLNet mà dùng embedding vector rồi huấn luyện bằng mô hình nhẹ hơn (LR, SVM, LSTM). Tuy nhiên hiệu suất mô hình sẽ không được như ban đầu.
- Tối ưu bằng **ONNX** hoặc **quantization** (nhưng giảm độ chính xác (float32 → float16/int8) → giảm 50–75% bộ nhớ).
- Dùng **LIME, SHAP hoặc explain_transformers** để giải thích.

DeBERTa

Điểm yếu

- **Kích thước mô hình lớn**: nhiều tham số và kiến trúc phức tạp hơn BERT, XLNet → cần GPU.
- **Tốn thời gian huấn luyện hơn**: nếu không dùng GPU sẽ rất lâu.
- **Cần tiền xử lý tốt**: cần tokenizer chuyên biệt, giới hạn đầu vào (512 tokens).
- Tính diễn giải của mô hình thấp.

Cách cải thiện

- Không fine-tune toàn bộ mà trích embedding vector từ DeBERTa → huấn luyện mô hình nhẹ hơn (LR, SVM, LSTM). Tuy nhiên hiệu suất mô hình sẽ không được như ban đầu.
- Tối ưu bằng **ONNX** hoặc **quantization** (nhưng giảm độ chính xác (float32 → float16/int8) → giảm 50–75% bộ nhớ).
- Dùng **LIME, SHAP hoặc explain_transformers** để giải thích.