

ĐẠI HỌC UEH
TRƯỜNG CÔNG NGHỆ VÀ THIẾT KẾ
KHOA CÔNG NGHỆ THÔNG TIN KINH DOANH



BÁO CÁO CUỐI KỲ
PHÂN TÍCH CẢM XÚC VÀ XÂY DỰNG MÔ HÌNH
PHÂN LỚP VỚI DỮ LIỆU DISNEYLAND

Tên học phần	:	Dữ liệu lớn và ứng dụng
Mã học phần	:	24D1INF50907903
Giảng viên hướng dẫn	:	TS. Đặng Nhân Cách
Nhóm sinh viên	:	Nhóm 1
Võ Tuấn Cường	:	31211027631
Nguyễn Thị Thơm	:	31211027673
Lưu Trọng Tốt	:	31211027678

TP. Hồ Chí Minh, ngày 27 tháng 3 năm 2024

MỤC LỤC

LỜI MỞ ĐẦU	4
PHẦN 1: TỔNG QUAN.....	1
I. Giới thiệu vấn đề	1
1. <i>Tổng quan về Big Data</i>	<i>1</i>
2. <i>Phân tích cảm xúc</i>	<i>2</i>
3. <i>Phân loại dữ liệu văn bản</i>	<i>2</i>
II. Mục tiêu nghiên cứu.....	2
III. Phương pháp nghiên cứu	3
IV. Dữ liệu đề tài.....	3
PHẦN 2: XỬ LÝ VÀ PHÂN TÍCH DỮ LIỆU.....	4
I. Tiền xử lý dữ liệu	4
1. <i>Xử lý dữ liệu bị thiếu</i>	<i>4</i>
2. <i>Gán nhãn dữ liệu.....</i>	<i>4</i>
3. <i>Xử lý dữ liệu text</i>	<i>5</i>
II. Sentiment Analysis	6
1. <i>Word Cloud.....</i>	<i>7</i>
2. <i>TextBlob</i>	<i>7</i>
3. <i>VADER.....</i>	<i>10</i>
4. <i>NLTK.....</i>	<i>10</i>
III. Emotion Analysis	11
PHẦN 3: XÂY DỰNG MÔ HÌNH.....	14
I. Tạo tập dữ liệu huấn luyện và kiểm tra.....	14
II. Trích chọn đặc trưng	14
1. <i>Phương pháp TF_IDF</i>	<i>14</i>
2. <i>Phương pháp Bag – of - Words.....</i>	<i>15</i>

III. Mô hình hóa	16
1. <i>Mô hình Naive Bayes</i>	16
2. <i>Mô hình Logistic Regression (MaxEntClassifier)</i>	20
3. <i>Mô hình học sâu (Deep Learning)</i>	24
PHẦN 4: ĐÁNH GIÁ MÔ HÌNH VỚI DỮ LIỆU CRAWL	32
I. Tổng quan dữ liệu	32
II. Xử lý dữ liệu	32
III. Đánh giá mô hình với dữ liệu crawl	32
PHẦN 5: ĐÁNH GIÁ KẾT QUẢ VÀ KẾT LUẬN	34
I. So sánh kết quả của các mô hình	34
II. Tóm tắt kết quả và nhận xét	35
III. Những hạn chế và hướng phát triển	35
1. <i>Hạn chế</i>	35
2. <i>Hướng phát triển</i>	36
PHẦN 6: GIAO DIỆN DEMO	37
I. Thư viện Gradio và Interface	37
II. Xây dựng giao diện	37
PHỤ LỤC	39
TÀI LIỆU THAM KHẢO	43

LỜI MỞ ĐẦU

Disneyland là một trong những điểm đến giải trí nổi tiếng trên toàn cầu và biểu tượng của kỷ nguyên phim hoạt hình. Đây là địa điểm thu hút hàng triệu du khách mỗi năm.

Bên cạnh những trải nghiệm tuyệt vời, Disneyland cũng nhận được hàng ngàn đánh giá và bình luận từ phía khách hàng. "Disneyland Reviews.csv" là một tập dữ liệu thu thập những đánh giá đó. Nhóm sử dụng dữ liệu này và một số phương pháp trong môn học Big data để phân tích trải nghiệm khách hàng. Trong đó, chủ yếu là áp dụng xử lý ngôn ngữ tự nhiên (NLP), phân tích cảm xúc (Sentiment Analysis) và phân loại dữ liệu văn bản (Text Classification) kết hợp xây dựng giao diện để phân tích những đánh giá này.

Qua đồ án, không chỉ giúp hiểu rõ hơn về những cảm nhận và ý kiến cá nhân mà còn cung cấp thông tin quan trọng về trải nghiệm của khách hàng, xu hướng phổ biến và các điểm mạnh, yếu của công viên giải trí này. Bên cạnh đó, nhóm thực hiện học hỏi thêm kiến thức chuyên môn và ứng dụng các kỹ thuật phân tích vào các dự án khác.

PHẦN 1: TỔNG QUAN

I. Giới thiệu vấn đề

1. Tổng quan về Big Data

Dữ liệu lớn (Big Data) là thuật ngữ dùng để miêu tả tập hợp các tập dữ liệu cực kỳ lớn và phức tạp mà các công cụ truyền thống xử lý và quản lý dữ liệu không thể xử lý một cách hiệu quả.

Có rất nhiều định nghĩa khác nhau về dữ liệu lớn, từ định nghĩa truyền thống Big Data = 3Vs (Volume, Variety, Velocity). Theo thời gian đặc tính của Big Data cũng được bổ sung nhiều chữ V hơn mà hiện nay được hiểu chủ yếu theo 6Vs với đặc điểm như sau:

- Volume (khối lượng): khối lượng dữ liệu được tạo ra, lưu trữ và xử lý cực lớn (hàng trăm ngàn Terabyte, thậm chí hiện nay đã phải dùng đến zettabyte (ZB) để đo lường). Và sau này có thể tăng lên còn hơn thế nữa, thậm chí không thể tính bằng đơn vị. Điều này đặt ra thách thức về việc lưu trữ và quản lý dữ liệu một cách hiệu quả.
- Velocity (Tốc độ sinh): Dữ liệu lớn thường được tạo ra và lưu lại với tốc độ nhanh chóng. Điển hình hình đó chính là sự phát triển của Internet of Things (IoT) dẫn đến việc tốc độ truyền dữ liệu càng nhanh hơn.
- Variety (Độ đa dạng): Dữ liệu lớn không chỉ bao gồm dữ liệu có cấu trúc (structure) dữ liệu trong cơ sở dữ liệu truyền thống, mà còn bao gồm dữ liệu phi cấu trúc (unstructured) như văn bản (text), ảnh (pictures), video, audio, ... hay thậm chí cả dữ liệu bán cấu trúc (semi-structure) như file json hay file xml.
- Veracity (Độ tin cậy): Dữ liệu các nguồn khác nhau có độ tin cậy khác nhau.
- Value (Giá trị): Mục tiêu chính của việc xử lý dữ liệu lớn là tạo ra giá trị từ dữ liệu vì vậy cần xác định giá trị mà dữ liệu mang lại đến như thế nào từ đó mới xác định được có nên triển khai dữ liệu hay không.

- Variability (Tính biến đổi): Đề cập đến sự không nhất quán có thể được hiển thị bởi dữ liệu, do đó có thể cản trở quá trình có thể xử lý và quản lý dữ liệu một cách hiệu quả.

2. *Phân tích cảm xúc*

Phân tích cảm xúc (Sentiment Analysis) tập trung vào việc đánh giá và phân tích cảm xúc của khách hàng khi họ trải nghiệm sản phẩm và dịch vụ. Điều này có thể bao gồm việc phân tích các đánh giá, bình luận, hoặc phản hồi của khách hàng để xác định xem họ cảm thấy như thế nào về trải nghiệm của mình từ đó ta có thể đưa ra được các chiến lược và cách giải quyết phù hợp để phù hợp với nhu cầu của các khách hàng.

3. *Phân loại dữ liệu văn bản*

Phân loại văn bản (Text classification) là quá trình xác định và gán một hoặc nhiều nhãn (labels) cho các mẫu văn bản dựa trên nội dung của chúng. Mục tiêu của text classification là tự động phân loại các đoạn văn bản vào các danh mục cụ thể, giúp tổ chức và hiểu được thông tin từ các nguồn văn bản lớn. Text classification có ý nghĩa quan trọng trong nhiều lĩnh vực và ứng dụng khác nhau. Dưới đây là một số ý nghĩa chính của text classification:

- Tổ chức thông tin.
- Phân tích ý kiến và đánh giá.
- Phân loại nội dung.
- Phát hiện tin tức giả và thư rác.
- Hỗ trợ ra quyết định.

II. Mục tiêu nghiên cứu

Mục tiêu nghiên cứu của đề tài là ứng dụng Sentiment Analysis và các phương pháp máy học và xử lý ngôn ngữ vào bộ dữ liệu để tìm hiểu và phân tích ý kiến, cảm xúc của du khách đối với công viên. Cuối cùng dựa vào kết quả có được để sử dụng mô hình để phân lớp dữ liệu và đề ra hướng phát triển cho đề tài.

III. Phương pháp nghiên cứu

Trước hết, nhóm tập trung áp dụng các phương pháp nghiên cứu để khám phá và hiểu sâu hơn về bộ dữ liệu “Disneyland Reviews”. Sau đó tiến xử lý sao cho phù hợp và thuận lợi cho việc sử dụng các mô hình phân lớp và học sâu

Tiếp theo nhóm tiến hành Sentiment Analysis cho bộ dữ liệu bằng cách sử dụng những phương pháp như TextBlob, Vader, NLTK và Wordcloud để phân tích những thông tin chi tiết về những bình luận từ bộ dữ liệu.

Thực hiện xây dựng các mô hình phân lớp văn bản, mô hình học sâu dựa trên trích chọn đặc trưng và so sánh giữa các mô hình để tìm ra mô hình tốt nhất cho việc áp dụng vào bộ dữ liệu mà nhóm tự đào để phân tích dự đoán cảm xúc

Cuối cùng, nhóm sẽ tóm tắt lại những kết quả nghiên cứu để đánh giá và kết luận. Từ đó, chỉ ra những hạn chế mà nghiên cứu gặp phải và đề ra hướng phát triển để cải thiện đề tài

IV. Dữ liệu đề tài

Bộ dữ liệu DisneylandReview.csv bao gồm 42.000 đánh giá về 3 chi nhánh của Disneyland bao gồm Paris, California và Hồng Kông được khách truy cập đăng trên Trip Advisor. (Nguồn: <https://www.kaggle.com/datasets/arushchillar/disneyland-reviews>)

STT	Tên thuộc tính	Mô tả thuộc tính
1	Review_ID	ID được cung cấp cho mỗi đánh giá
2	Rating	Từ 1(Không hài lòng) đến 5(Thỏa mãn)
3	Year_Month	Thời gian du khách đến thăm công viên
4	Reviewer_Location	Đất nước xuất xứ của du khách
5	Review_Text	Bình luận của du khách
6	Disneyland_Branch	Vị trí của công viên Disneyland

Bảng 1: Thuộc tính trong dữ liệu

PHẦN 2: XỬ LÝ VÀ PHÂN TÍCH DỮ LIỆU

I. Tiền xử lý dữ liệu

1. Xử lý dữ liệu bị thiếu

```
1 data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 42656 entries, 0 to 42655
Data columns (total 6 columns):
 #   Column                Non-Null Count  Dtype  
---  --
 0   Review_ID             42656 non-null  int64  
 1   Rating                42656 non-null  int64  
 2   Year_Month            42656 non-null  object  
 3   Reviewer_Location     42656 non-null  object  
 4   Review_Text           42656 non-null  object  
 5   Branch                42656 non-null  object  
dtypes: int64(2), object(4)
memory usage: 2.0+ MB
```

Hình 1: Kết quả kiểm tra dữ liệu bị thiếu

Bộ dữ liệu không có giá trị bị thiếu nên nhóm không tiến hành xử lý dữ liệu bị thiếu.

2. Gán nhãn dữ liệu

Để thực hiện quá trình gán nhãn dữ liệu trước khi đưa vào huấn luyện, nghiên cứu áp dụng phương pháp phân loại cảm xúc theo điểm số đánh giá (Rating) của du khách, để phân chia tập dữ liệu đã thu thập được thành 3 bộ dữ liệu được gán nhãn theo quy tắc sau: rating ≤ 2 : bình luận nào đánh giá dưới 2 sao sẽ được dán nhãn là tiêu cực (Negative). Rating = 3: Neutral và Rating > 3 : bình luận nào đánh giá trên 3 sao sẽ được dán nhãn là tích cực (Positive).

```
rating = data["Rating"]
def label_func(rating):
    if rating >= 4:
        return "Positive"
    elif rating == 3:
        return "Neutral"
    else:
        return "Negative"

data["Sentiment"] = data["Rating"].apply(lambda x: label_func(x))
```

Hình 2: Gán nhãn cột Rating

3. Xử lý dữ liệu text

Để thuận lợi cho việc phân tích, hiểu và áp dụng các phương pháp xử lý ngôn ngữ tự nhiên hiệu quả hơn, nhóm thực hiện tiền xử lý dữ liệu để giảm nhiễu cho cột Review_Text. Các bước làm sạch và chuẩn hóa dữ liệu được thực hiện như sau:

```
def clean_reviews(text):  
    # Xóa html  
    regex = re.compile('<.*?>')  
    text = re.sub(regex, '', text)  
  
    # Xóa những ký tự đặc biệt  
    pattern = re.compile('[^a-zA-z0-9\s]')  
    text = re.sub(pattern, '', text)  
  
    # Xóa chữ số  
    pattern = re.compile('\d+')  
    text = re.sub(pattern, '', text)  
  
    # Chuyển đổi text sang chữ thường  
    text = text.lower()  
  
    # Tokenization of words  
    text = word_tokenize(text)  
  
    # Lemmatization of words  
    lemmatizer = WordNetLemmatizer()  
    text = [lemmatizer.lemmatize(word) for word in text]  
  
    # Stop words removal  
    text = [word for word in text if word not in stop_words]  
  
    # Nối các từ tạo thành một chuỗi  
    text = ' '.join(text)  
  
    return text  
data['Review_Text'] = data['Review_Text'].apply(clean_reviews)
```

Hình 3: Tiền xử lý dữ liệu cột Review_Text

a. Loại bỏ HTML: loại bỏ các thẻ HTML trong văn bản reviews. Trong việc xử lý dữ liệu, thông thường văn bản có thể chứa các thẻ HTML như "<p>", "<div>", "<a>",... Các thẻ này không mang ý nghĩa ngôn ngữ tự nhiên và cần được loại bỏ để tập trung vào nội dung văn bản review.

b. Chuyển đổi văn bản về lowercase: việc chuẩn hóa văn bản thông qua việc chuyển đổi về lowercase để đồng nhất dữ liệu. Điều này giúp giảm sự phức tạp trong việc xử lý văn bản, vì các từ viết hoa và viết thường sẽ được coi là giống nhau.

c. Loại bỏ kí tự đặc biệt và dấu câu: loại bỏ ký tự đặc biệt và dấu câu không cần thiết. Việc này giúp làm sạch dữ liệu, loại bỏ các ký tự như dấu chấm, dấu phẩy, dấu chấm hỏi, và các ký tự đặc biệt khác có thể làm nhiễu dữ liệu và không mang ý nghĩa ngữ nghĩa.

d. Loại bỏ stopwords: các từ phổ biến và không mang ý nghĩa quan trọng trong việc hiểu ý nghĩa của văn bản như "and", "or", "but", vv. Loại bỏ stop words giúp giảm kích thước của văn bản và tập trung vào các từ khóa quan trọng hơn.

e. Tokenization: phân chia văn bản thành các từ hoặc cụm từ riêng lẻ. Quá trình này tạo ra các đơn vị nhỏ hơn từ văn bản gốc, gồm các từ hoặc cụm từ để sử dụng trong các phân tích và mô hình hóa.

f. Lemmatization: là một quá trình trong xử lý ngôn ngữ tự nhiên tương tự như stemming, nhưng lemmatization không chỉ cắt bỏ các hậu tố để thu được "stem", mà nó còn chuyển đổi từ về dạng gốc thực sự, phù hợp với từ điển và ngữ cảnh ngữ pháp. Nên trong phần này nhóm chỉ sử dụng phương pháp lemmatization.

Review_Text	Rating		Review_Text	Sentiment
If you've ever been to Disneyland anywhere you...	4		youve ever disneyland anywhere youll find disn...	Positive
Its been a while since d last time we visit HK...	4		since last time visit hk disneyland yet time s...	Positive
Thanks God it wasn t too hot or too humid wh...	4		thanks god hot humid wa visiting park otherwis...	Positive
HK Disneyland is a great compact park. Unfortu...	4		hk disneyland great compact park unfortunately...	Positive
the location is not in the city, took around 1...	4		location city took around hour kowlon kid like...	Positive
...
i went to disneyland paris in july 03 and thou...	5		went disneyland paris july thought wa brillian...	Positive
2 adults and 1 child of 11 visited Disneyland ...	5		adult child visited disneyland paris beginning...	Positive
My eleven year old daughter and myself went to...	5		eleven year old daughter went visit son london...	Positive
This hotel, part of the Disneyland Paris compl...	4		hotel part disneyland paris complex wonderful ...	Positive
I went to the Disneyparis resort, in 1996, wit...	4		went disneyparis resort small child minute ent...	Positive

Dữ liệu trước khi tiền xử lý

Dữ liệu sau khi tiền xử lý

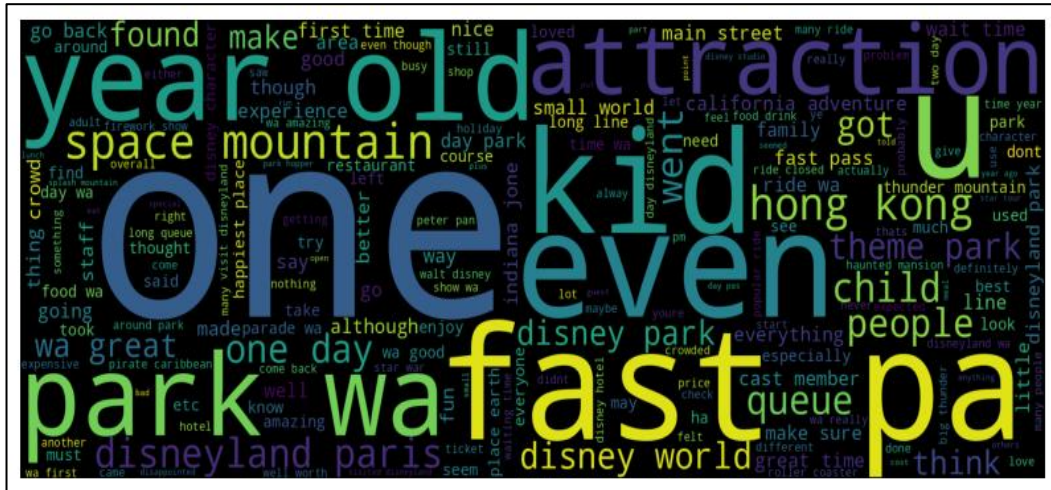
II. Sentiment Analysis

Sentiment Analysis về cơ bản đây là quá trình ứng dụng của trí tuệ nhân tạo, sử dụng các thuật toán phức tạp để xử lý ngôn ngữ tự nhiên của con người (NLP). Qua đó xác định được cảm xúc của du khách mà trong bài này nhóm sử dụng 3 loại là Positive, Negative và Neutral.

Để có thể có nhận định sâu hơn về reviews của du khách, nhóm đã dùng bốn phương pháp là Word Cloud, TextBlob, Vader và NLTK.

1. Word Cloud

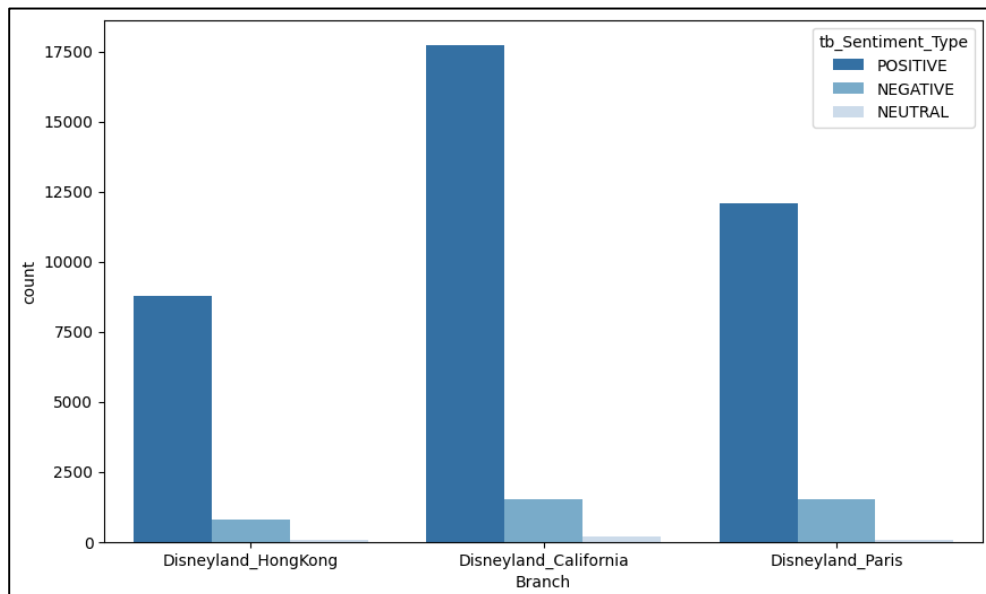
Nhóm đã dùng thư viện WordCloud để làm rõ những từ ngữ xuất hiện nhiều lần nhất trong bộ dữ liệu. Những từ có kích thước lớn hơn sẽ xuất hiện nhiều lần hơn và ngược lại.



Hình 4: Word Cloud của bộ dữ liệu

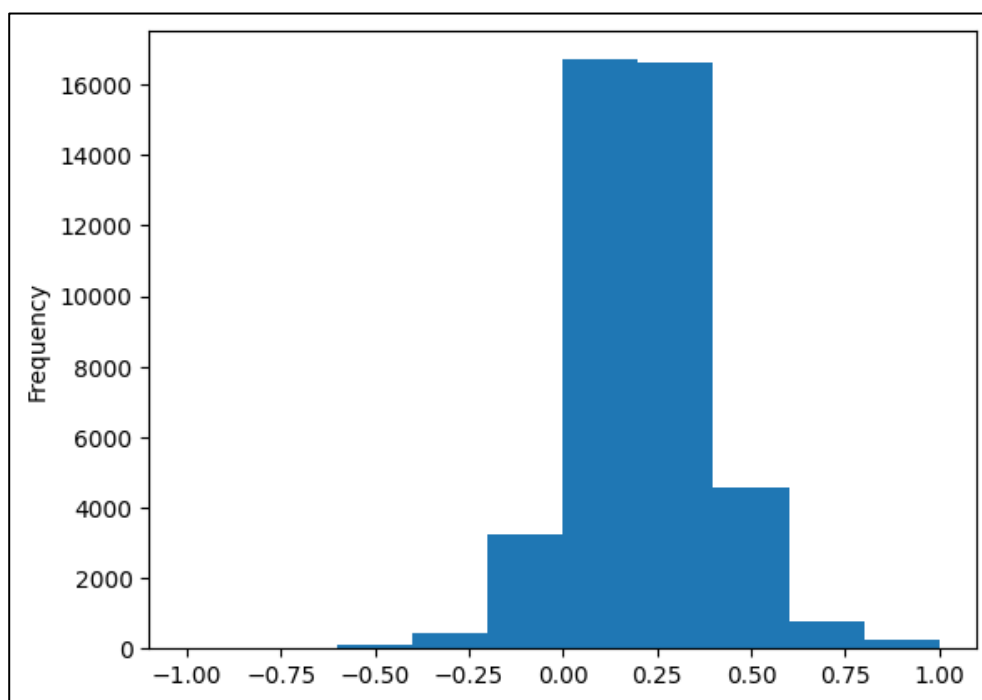
2. *TextBlob*

Nhóm đã dùng thư viện TextBlob để tính toán và trả về kết quả tính khách quan “polarity” và tính chủ quan “subjectivity”.



Hình 5: Biểu đồ thể hiện số lượng reviews theo sentiment type

- Tính khách quan “polarity”: Là các giá trị số nằm trong khoảng từ $[-1;1]$. Trong đó -1 đang thể hiện tâm lý tiêu cực của người viết và 1 là với tâm trạng tích cực.

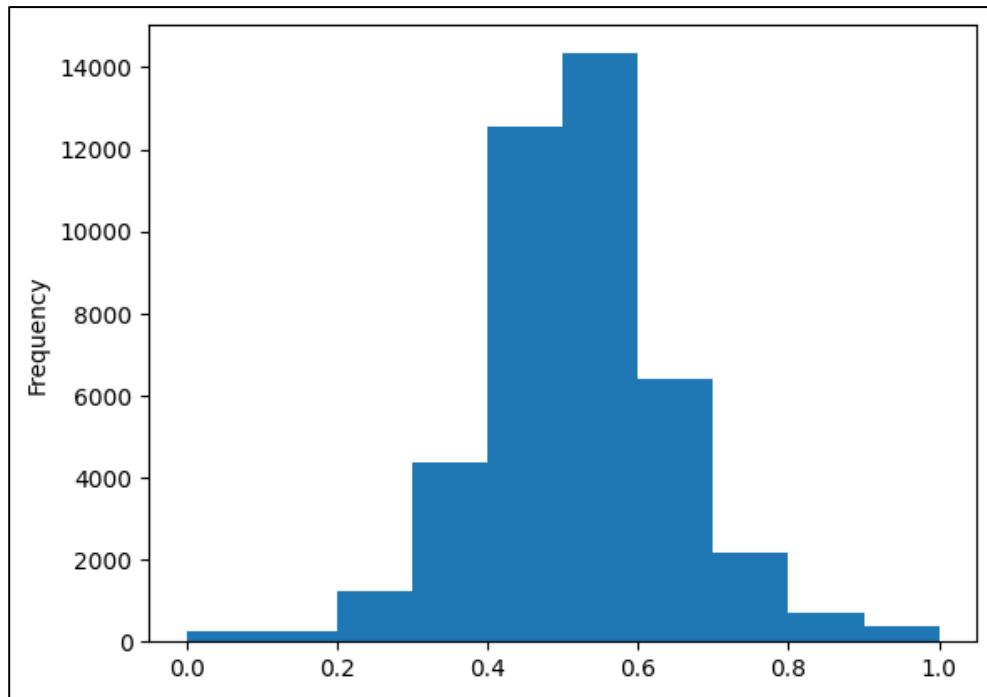


Hình 6: Biểu đồ thể hiện tính khách quan

Về tổng thể, có thể thấy được phần lớn các nhận xét được ghi nhận từ khách hàng đều có mức polarity ở mức trung bình. Bên cạnh đó, có nhiều lời review có độ polarity gần bằng 0, có nghĩa là chúng có cảm xúc không rõ ràng. Tuy nhiên, cũng có một số câu có mức polarity cao.

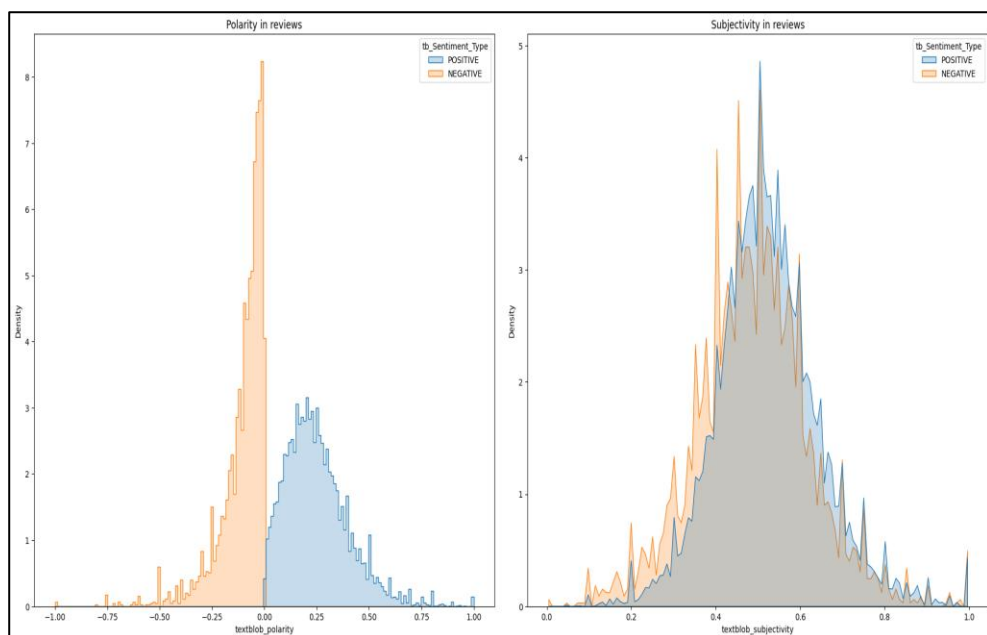
Biểu đồ cũng cho thấy những lời nhận xét tích cực nhiều hơn là tiêu cực.

- Tính chủ quan “subjectivity”: Đo lường lượng ý kiến cá nhân và thông tin thực tế có trong văn bản. Bao gồm các giá trị nằm trong đoạn từ $[0;1]$, trong đó 0 biểu thị văn bản mang tính khách quan và 1 biểu thị văn bản mang tính chủ quan. Tính chủ quan cao hơn là văn bản chứa đựng quan điểm cá nhân nhiều hơn là thông tin thực tế.



Hình 7: Biểu đồ thể hiện tính chủ quan

Về tổng thể, có thể thấy được phần lớn các nhận xét được ghi nhận từ khách hàng đều có mức subjectivity ở mức trung bình. Bên cạnh đó, có nhiều lời review có mức subjectivity ở mức từ $[0.5; 0.6]$, có nghĩa là chúng mang tính chủ quan hơn khách quan. Tuy nhiên, cũng có một số câu có mức subjectivity dưới 0.5.

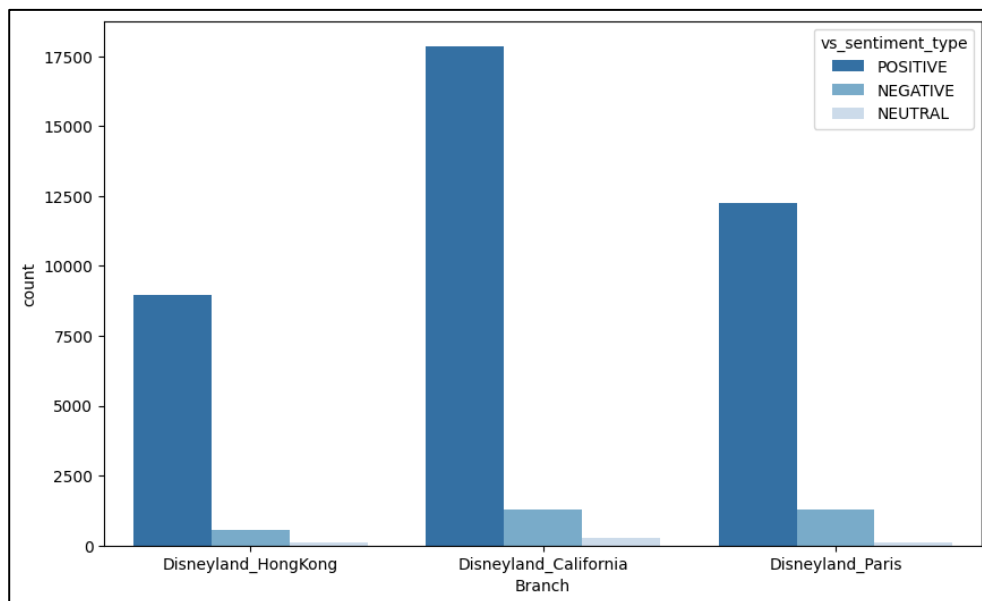


Hình 8: Biểu đồ Figure về 2 mức độ polarity và subjectivity

3. VADER

Valence Aware Dictionary và Sentiment Reasoner (VADER) là mô hình dựa trên quy tắc từ vựng để phân tích tình cảm. Kết quả do VADER tạo ra gồm một từ điển gồm 4 khóa: neg, neu, pos và compound. Trong đó, neg, neu và pos lần lượt có nghĩa là tiêu cực, trung tính và tích cực. Tổng của chúng phải bằng 1 hoặc gần bằng 1. Compound tương ứng với tổng điểm của mỗi từ vựng và nằm trong đoạn $[-1;1]$ với 1 là tích cực nhất và -1 là tiêu cực nhất.

- Positive sentiment: (compound score ≥ 0.65)
- Neutral sentiment: (compound score ≥ 0.4) and (compound score < 0.65)
- Negative sentiment: (compound score < 0.4)



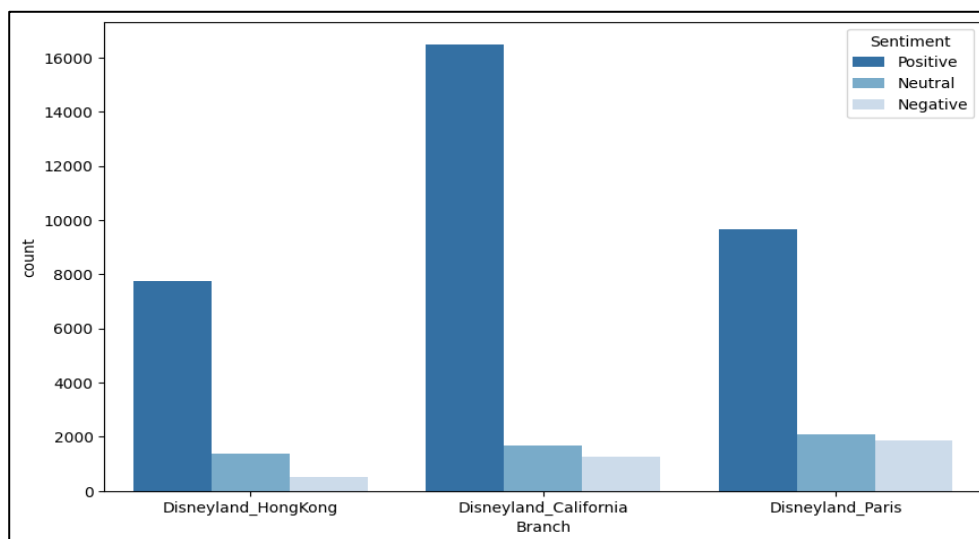
Hình 9: Biểu đồ phân phối VADER theo địa điểm

Qua biểu đồ, nhóm thấy được Disneyland ở California là địa điểm có số lượng du khách nhận xét và có tỷ lệ đánh giá tích cực cao nhất. Để giải thích cho việc này có thể là do phần lớn khách tham quan được khảo sát ở Mỹ nên số lượng người đến với California nhiều hơn so với HongKong và Paris. Nhưng nhìn chung, du khách khi đến với Disneyland đều có trải nghiệm tốt và có tỷ lệ đánh giá tích cực gấp nhiều lần so với tiêu cực.

4. NLTK

Thư viện NLTK-Natural Language Toolkit là một trong những thư viện open-source xử lý ngôn ngữ tự nhiên. NLTK đã được tích hợp sẵn một bộ phân tích cảm xúc có tên là VADER.

Vì VADER đã được đào tạo sẵn nên ta có thể nhận được kết quả nhanh hơn so với nhiều bộ phân tích khác. Tuy nhiên, VADER phù hợp nhất với ngôn ngữ được sử dụng trên mạng xã hội, như các câu ngắn có từ lóng và viết tắt. Nó kém chính xác khi đánh giá câu dài hơn, không có cấu trúc.



Hình 10: Biểu đồ phân phối NLTK theo địa điểm

III. Emotion Analysis

Để tìm hiểu được cảm xúc của du khách khi để lại những lời nhận xét sau khi có một ngày đến chơi tại Disneyland, nhóm đã dùng NRClex để đo lường. Từ điển chứa khoảng 27.000 từ và dựa trên từ điển tác động của Hội đồng Nghiên cứu Quốc gia Canada (NRC) và các bộ từ đồng nghĩa WordNet của thư viện NLTK.

```
emotion = []
for i in range(len(data)):
    emotions = NRClex(data['Review_Text'][i])
    emotion.append(emotions.top_emotions[0][0])
data['emotion'] = emotion
```

Hình 11: Thực hiện phân tích emotion

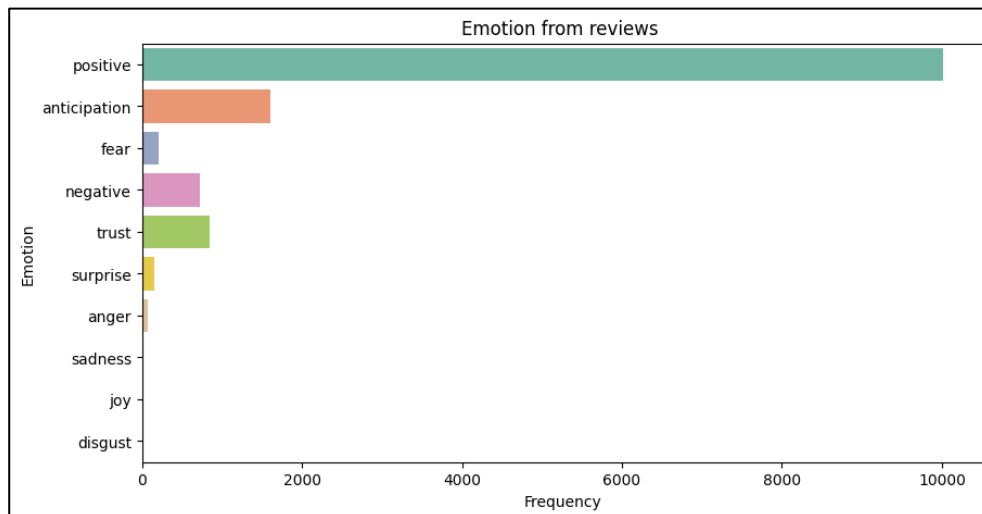
- Kết quả:

1	<code>data['emotion'].value_counts()</code>
positive	29814
anticipation	5849
trust	2820
negative	2083
fear	1001
surprise	677
anger	363
sadness	32
joy	13
disgust	4
Name: emotion, dtype: int64	

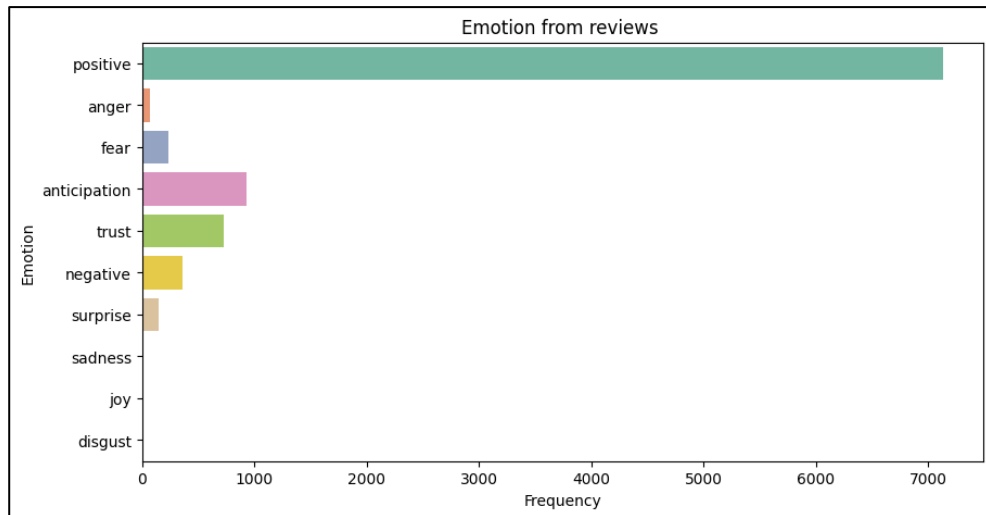
Hình 12: Kết quả phân tích emotion

Sau khi tiến hành đo lường bằng viện sử dụng từ điển NCRLex, nhóm nhận thấy được phần lớn cảm xúc của du khách đều mang ý nghĩa tích cực, trông đợi và tin tưởng vào chuyến tham quan vui chơi của mình. Nhưng bên cạnh đó, vẫn có một số lời nhận xét mang tính tiêu cực, sợ hãi và không hài lòng ở khía cạnh nào đó khi đến Disneyland.

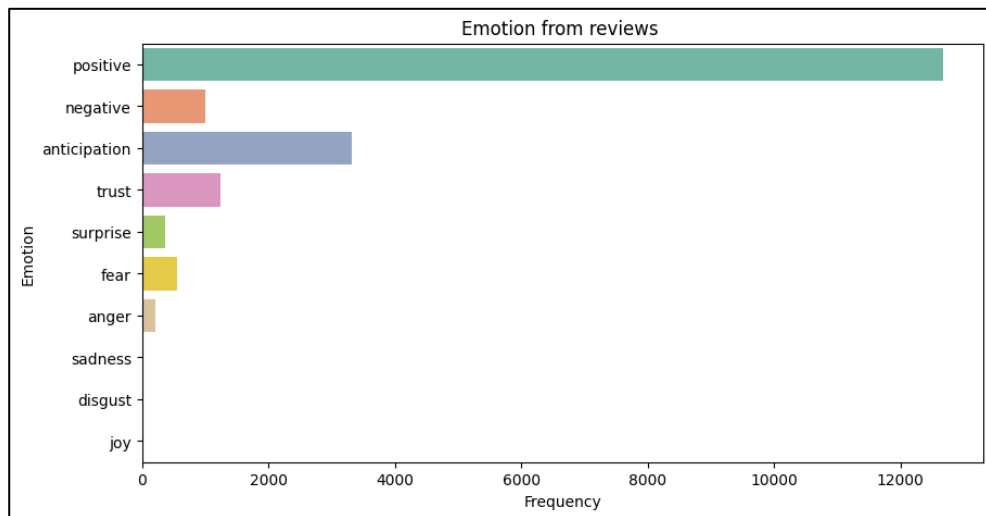
Để có thể tìm hiểu rõ hơn, nhóm đã tiến hành phân tích các cảm xúc của du khách khi đến với Disneyland ở Paris, Hongkong và California.



Hình 13: Biểu đồ trực quan emotion tại Disneyland ở Paris



Hình 14 Biểu đồ trực quan emotion tại Disneyland ở Hongkong



Hình 15: Biểu đồ trực quan emotion tại Disneyland ở California

Qua cả ba biểu đồ cho thấy, điểm chung của du khách khi đến tham quan đều có cái nhìn tích cực và hào hứng. Tuy nhiên, ở Disneyland tại California, tần số xuất hiện của những từ ngữ mang tính tiêu cực chiếm phần lớn nhất.

PHẦN 3: XÂY DỰNG MÔ HÌNH

I. Tạo tập dữ liệu huấn luyện và kiểm tra

Trước khi chạy các mô hình phân lớp, nhóm sẽ tiến hành phân bộ dữ liệu thành train và test set độc lập với nhau. Training set sẽ dùng để huấn luyện mô hình, và testing set sẽ dùng để kiểm thử, đánh giá độ chính xác của mô hình phân lớp đã được huấn luyện.

Vì mô hình được xây dựng nhằm phân tích Review của du khách nên nhóm chỉ sử dụng biến đầu vào là Review_text và biến mục tiêu là Sentiment. Sử dụng thư viện sklearn để tự động hóa chia tập dữ liệu huấn luyện và kiểm thử. Sử dụng stratify bằng biến mục tiêu để đảm bảo rằng phân phối của các nhãn trong tập dữ liệu ban đầu được duy trì trong cả tập huấn luyện và tập kiểm tra. Kết quả sau khi thực hiện việc tách tập dữ liệu data sẽ thu được tập huấn luyện chứa 34124 reviews (chiếm 80%), tập kiểm thử chứa 8532 reviews (chiếm 20%).

```
1 X = df_main_class['Review_Text']
2 y = df_main_class['Sentiment']
3
4 # Chia dữ liệu thành tập huấn luyện và tập kiểm tra với tỷ lệ 80-20
5 train_x, test_x, train_y, test_y = train_test_split(X, y, test_size=0.2, stratify=y, random_state=42)
6
7 print('Tập Train (80%): ', train_x.shape)
8 print('Tập Test (20%): ', test_x.shape)
```

Tập Train (80%): (34124,)
Tập Test (20%): (8532,)

Hình 16: Phân tách tập dữ liệu thành tập Train – Test

II. Trích chọn đặc trưng

1. Phương pháp TF_IDF

Phương pháp TF-IDF (Term Frequency-Inverse Document Frequency) là một kỹ thuật trích chọn đặc trưng để biểu diễn văn bản. TF-IDF kết hợp cả tần suất xuất hiện của một từ trong một văn bản (TF) và tần suất nghịch đảo của từ đó.

Thực hiện trích chọn đặc trưng theo phương pháp TF-IDF, sử dụng module TfidfVectorizer trong thư viện Sklearn. Kết quả sau quá trình này sẽ thực hiện chuyển đổi dữ liệu văn bản sang vector số. Dữ liệu số này sẽ được sử dụng là đầu vào cho mô hình học máy Naive Bayes và Logistic Regression để thực hiện phân lớp các Review_Text.

```

1  #Trích chọn đặc trưng theo phương pháp TF_IDF
2  vector =TfidfVectorizer(analyzer = 'word',
3                          max_features=35000,
4                          stop_words = 'english')
5  vector.fit(df_main_class['Review_Text'])
6  xtrain_tfidf = vector.transform(train_x)
7  xtest_tfidf = vector.transform(test_x)
8  print('Kết quả vector hóa tập Train sang dạng số:')
9  print(xtrain_tfidf.data)
10 print(xtrain_tfidf.shape)
11 print('Kết quả vector hóa tập Test sang dạng số:')
12 print(xtest_tfidf.data)
13 print(xtest_tfidf.shape)

```

Kết quả vector hóa tập Train sang dạng số:
 [0.10443389 0.06894211 0.07245709 ... 0.24082332 0.44093295 0.30857837]
 (34124, 35000)
 Kết quả vector hóa tập Test sang dạng số:
 [0.06356125 0.29141832 0.12852096 ... 0.08389852 0.14622652 0.11414897]
 (8532, 35000)

Hình 17: Kết quả vector hóa Review_Text sang dạng số sử dụng TF_IDF

2. Phương pháp Bag – of - Words

Phương pháp trích chọn đặc trưng Bag-of-Words (BoW) có nghĩa là bỏ túi các từ, là một kỹ thuật quan trọng trong xử lý ngôn ngữ tự nhiên và học máy để biểu diễn văn bản dưới dạng các vector đặc trưng. Trong BoW, mỗi văn bản được biểu diễn dưới dạng một vector, trong đó mỗi chiều của vector tương ứng với một từ trong từ điển, và giá trị của mỗi chiều thể hiện số lần xuất hiện của từ đó trong văn bản.

Thực hiện trích chọn đặc trưng theo phương pháp Bag-of-Words, sử dụng module CountVectorizer trong thư viện Sklearn. Kết quả sau quá trình này sẽ thực hiện chuyển đổi dữ liệu văn bản sang vector số. Dữ liệu số này sẽ được sử dụng là đầu vào cho mô hình học máy Naive Bayes và Logistic Regression để thực hiện phân lớp các Review_Text.

```
1 from sklearn.feature_extraction.text import CountVectorizer
2
3 # Tạo một CountVectorizer
4 vectorizer = CountVectorizer(analyzer='word', max_features=20000, stop_words='english')
5
6 # Khớp dữ liệu văn bản vào vectorizer để tạo BoW
7 vectorizer.fit(df_main_class['Review_Text'])
8
9 # Chuyển đổi dữ liệu huấn luyện và kiểm tra thành dạng ma trận BoW
10 xtrain_bow = vectorizer.transform(train_x)
11 xtest_bow = vectorizer.transform(test_x)
12
13 # In ra thông tin về ma trận BoW
14 print('Kết quả vector hóa tập Train sang dạng số:')
15 print(xtrain_bow.toarray())
16 print(xtrain_bow.shape)
17 print('Kết quả vector hóa tập Test sang dạng số:')
18 print(xtest_bow.toarray())
19 print(xtest_bow.shape)
20
```

Kết quả vector hóa tập Train sang dạng số:

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
(34124, 20000)
```

Kết quả vector hóa tập Test sang dạng số:

```
[[0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 ...
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]
 [0 0 0 ... 0 0 0]]
(8532, 20000)
```

Hình kết quả trích chọn đặc trưng bằng BoW

III. Mô hình hóa

1. Mô hình Naive Bayes

1.1. Giới thiệu mô hình Naive Bayes

Naive Bayes Classifiers là một trong những thuật toán tiêu biểu cho bài toán phân lớp dựa trên lý thuyết xác suất áp dụng định lý Bayes. Kỹ thuật này dựa vào xác suất có điều kiện giữa các đặc trưng và nhãn phân loại để dự đoán văn bản cần phân loại thuộc lớp nào. Điểm quan trọng của phương pháp này là giả định tất cả các đặc trưng trong văn bản đều độc lập với nhau.

NBC có thể hoạt động với các vector đặc trưng mà một phần là liên tục (sử dụng Gaussian Naive Bayes), phần còn lại ở dạng rời rạc (sử dụng Multinomial hoặc Bernoulli). Trong phần thực nghiệm, nhóm sử dụng MultinomialNB để xây dựng mô hình. Mỗi văn bản được biểu diễn bởi một vector có độ dài d chính là số từ trong từ điển. Giá trị của thành phần thứ i trong mỗi vector chính là số lần từ thứ i xuất hiện trong văn bản đó.

1.2. Áp dụng mô hình Naive Bayes

1.2.1. Xây dựng mô hình

Để sử dụng thuật toán phân lớp Naive Bayes, nhóm sử dụng thư viện học máy Sklearn sử dụng giải thuật của Naive Bayes là Multinomial Naive Bayes có 2 tham số đầu vào chính bao gồm:

- `alpha`: Tham số dùng để làm mịn (Laplace smoothing). Thông thường giá trị thiết lập là một số thực nằm trong đoạn $[0,1]$. Nhóm sử dụng Grid Search tìm ra `alpha` có chỉ số accuracy cao nhất và tìm được `alpha = 0.1`.

```
1 # Sử dụng GridSearchCV để tìm alpha tốt nhất
2 MultiNB = nb.MultinomialNB()
3 alphas = {'alpha': [0.1, 0.25, 0.5, 0.75, 1.0]}
4 grid = GridSearchCV(estimator=MultiNB, param_grid=alphas, cv=5, scoring='accuracy')
5 grid.fit(xtrain_tfidf, train_y)
6 best_alpha = grid.best_params_['alpha']
7 print("Alpha tốt nhất:", best_alpha )

Alpha tốt nhất: 0.1
```

Hình 18: Sử dụng Grid Search tìm ra `alpha` có chỉ số accuracy cao nhất

- `fit_prior`: Tham số dùng để xác định có sử dụng xác suất các phần tử trước đó cho việc huấn luyện hay không. Giá trị thiết lập kiểu boolean (True|False), thiết lập mặc định bằng True.

Quá trình chạy mô hình với tham số `alpha = 0.1`, `fit_prior = True` cho kết quả tốt nhất. Hình 18 thể hiện việc xây dựng mô hình MultinomialNB, huấn luyện trên tập Train, và kiểm thử trên tập Test:

```
1 #Sử dụng mô hình Naive Bayes với TF-IDF
2 #Sử dụng thuật toán MultinomialNB
3 MultiNB = nb.MultinomialNB(alpha=best_alpha, fit_prior=True)
4 MultiNB.fit(xtrain_tfidf, train_y)
5
```

▼ MultinomialNB

MultinomialNB(alpha=0.1)

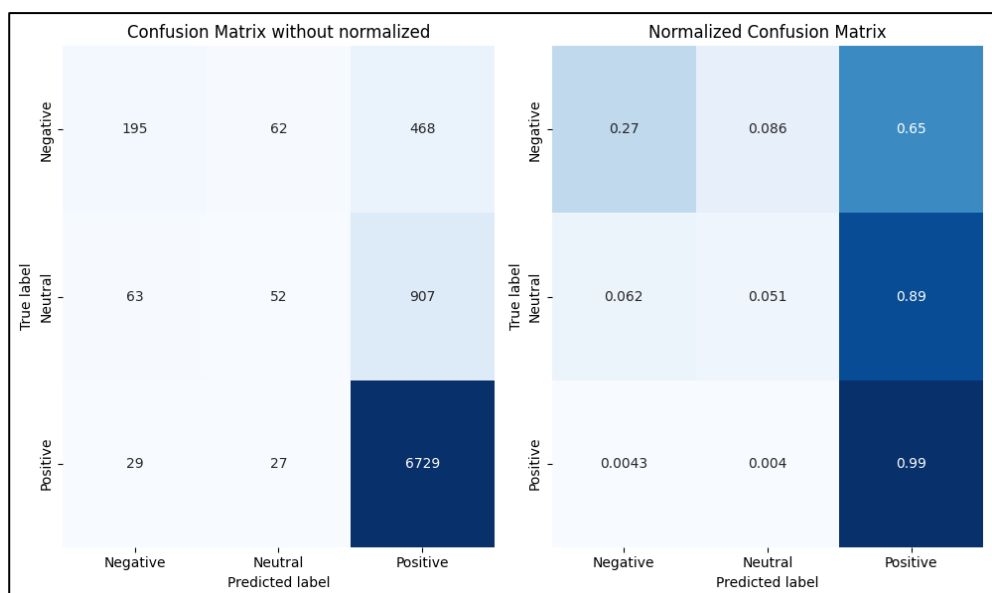
Hình 19: Xây dựng mô hình phân lớp văn bản sử dụng MultinomialNB

1.2.2. Đánh giá kết quả

Độ chính xác của mô hình trên tập Train: 86.27%				
Thời gian huấn luyện trên tập Train: 0.1009 giây				
classification_report:				
	precision	recall	f1-score	support
Negative	0.68	0.27	0.39	725
Neutral	0.37	0.05	0.09	1022
Positive	0.83	0.99	0.90	6785
accuracy			0.82	8532
macro avg	0.63	0.44	0.46	8532
weighted avg	0.76	0.82	0.76	8532
Thời gian dự đoán trên tập Test: 0.0037 giây				

Hình 20: Kết quả mô hình Naive Bayes sử dụng MultinomialNB

Hình 19 cho thấy độ chính xác của mô hình phân lớp MultinomialNB trên tập dữ liệu Train bao gồm 34124 reviews đạt 86.27%, thời gian để huấn luyện mô hình hết 0.1009 giây. Độ chính xác của mô hình khi chạy trên tập Test bao gồm 8532 reviews đạt 82%, thời gian để chạy mô hình trên tập dữ liệu Test hết 0.0037 giây. Để có cái nhìn tổng quan hơn về kết quả đánh giá độ chính xác của mô hình trên tập kiểm thử (Test), nhóm sử dụng Confusion matrix để trực quan hóa kết quả thể hiện trong Hình 20.



Hình 21: Kết quả dự đoán trên tập test của Naive Bayes với Confusion matrix

Có tất cả 6976 reviews trên tổng số 8532 reviews trong tập kiểm thử được dự đoán chính xác. Trong đó:

- 195 reviews lớp Negative và được dự đoán đúng vào lớp Negative (27%).
- 52 reviews lớp Neutral và được dự đoán đúng vào lớp Neutral (5.1%).
- 6729 reviews lớp Positive và được dự đoán đúng vào lớp Positive (99%).

Đa số các reviews thuộc lớp Negative và Neutral đều bị dự đoán sai sang lớp Positive:

- 468 reviews lớp Negative nhưng mô hình dự đoán sai sang lớp Positive (65%)
- 907 reviews lớp Neutral nhưng mô hình dự đoán sai sang lớp Positive (89%)

Bên cạnh đó, nhóm cũng thực hiện trích chọn đặc trưng theo phương pháp Bag of Words (BoW) và cài đặt bộ phân lớp với thuật toán học máy khác để đánh giá là Decision Tree.

Thuật toán phân lớp	MultinomialNB ($\alpha = 0.1$)		Decision Tree (entropy)	
Chỉ số đánh giá	TF-IDF	BoW	TF-IDF	BoW
Precision	0.63	0.60	0.46	0.47
Recall	0.44	0.62	0.46	0.47

<i>F_score</i>	0.46	0.61	0.46	0.47
<i>Accuracy</i>	0.82	0.82	0.74	0.75
<i>Thời gian (giây)</i>	0.0037	0.0055	0.0088	0.0082

Bảng 2: Kết quả thu được từ mô hình Naive Bayes

Dựa vào kết quả thu được cho thấy thuật toán phân lớp Naive Bayes với class MultinomialNB ($\alpha = 0.1$) thu được kết quả tốt hơn Decision Tree. Cả hai mô hình đều thu được kết quả tốt hơn khi sử dụng phương pháp trích chọn đặc trưng bằng Bag of Word.

2. Mô hình Logistic Regression (MaxEntClassifier)

2.1. Giới thiệu về mô hình Logistic Regression (MaxEnt Classifier)

Logistic Regression Classifier của scikit-learn cũng là một mô hình trong nhóm MaxEnt Classifier. Logistic Regression là một mô hình học máy được sử dụng rộng rãi trong bài toán phân loại. Nó được xây dựng dựa trên nguyên lý cực đại hóa thông tin (maximum entropy) và sử dụng hàm sigmoid để dự đoán xác suất xảy ra của biến mục tiêu.

2.2. Áp dụng mô hình Logistic Regression

Để sử dụng thuật toán phân lớp Logistic Regression, nhóm sử dụng thư viện học máy Sklearn sử dụng thuật toán Logistic Regression có 4 tham số đầu vào chính bao gồm:

- Penalty: Xác định loại độ lệch
- C: Điều chỉnh mức độ độ lệch
- Solver: Xác định thuật toán sử dụng cho quá trình tối ưu hóa
- Max iter: Số lần lặp để tìm ra giá trị tối ưu

Trước tiên tiến hành tìm những tham số tốt nhất cho mô hình sử dụng GridSearch:


```

1 C_values = [0.1, 0.6, 0.7, 0.8, 1.0, 5.0, 10.0]
2 solvers = ['newton-cg', 'lbfgs', 'liblinear', 'sag', 'saga']
3
4 # Create a logistic regression model
5 logistic_regression = LogisticRegression()
6
7 # Create a parameter grid
8 param_grid = {'C': C_values, 'solver': solvers}
9
10 # Perform grid search using cross-validation
11 grid_search = GridSearchCV(logistic_regression, param_grid, cv=5, scoring='accuracy')
12 grid_search.fit(xtrain_tfidf, train_y)
13
14 # Get the best parameters
15 best_parameters = grid_search.best_params_
16 best_C = best_parameters['C']
17 best_solver = best_parameters['solver']
18
19 print("Best C:", best_C)
20 print("Best solver:", best_solver)

```

Best C: 1.0
Best solver: lbfgs

Hình 22: Mã code tìm tham số tốt nhất cho Logistic Regression

Kết quả cho ra được tham số tốt nhất là $C = 1.0$ và solver = lbfgs. Sau khi nghiên cứu nhóm chọn ra được những tham số tốt nhất cho mô hình là:

Penalty	C	solver	max_iter
L2	1.0	lbfgs	100

Triển khai thuật toán Logistic Regression theo 2 trích chọn đặc trưng TF-IDF và Bow cho mô hình MaxEntClassifier.

- Áp dụng Logistic Regression với TF-IDF:

```

1 LogReg = LogisticRegression(penalty='l2', C=1.0, solver='lbfgs', max_iter=100)
2 LogReg.fit(xtrain_tfidf, train_y)

```

▼ LogisticRegression
LogisticRegression()

Hình 23: Áp dụng Logistic Regression với TF-IDF

- Áp dụng Logistic Regression với BoW:

```

1 LogReg = LogisticRegression(penalty='l2', C=1.0, solver='lbfgs', max_iter=100)
2 LogReg.fit(xtrain_bow, train_y)

```

▼ LogisticRegression
LogisticRegression()

Hình 24: Áp dụng Logistic Regression với BoW

2.3. Đánh giá kết quả

- Kết quả dựa trên TF-IDF:

Độ chính xác của mô hình trên tập Train: 88.41%				
Thời gian huấn luyện trên tập Train: 14.0400 giây				
classification_report:				
	precision	recall	f1-score	support
Negative	0.67	0.53	0.59	725
Neutral	0.49	0.23	0.31	1022
Positive	0.88	0.98	0.93	6785
accuracy			0.85	8532
macro avg	0.68	0.58	0.61	8532
weighted avg	0.82	0.85	0.83	8532
Thời gian dự đoán trên tập Test: 0.0054 giây				

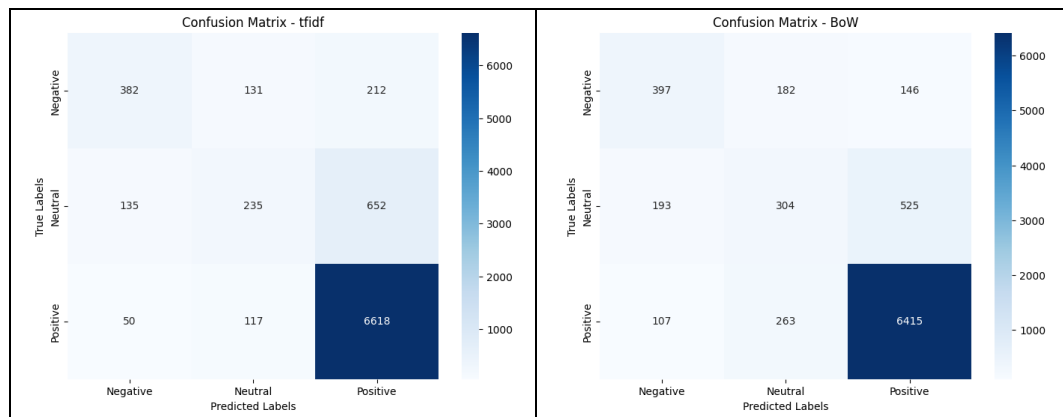
Hình 25: Kết quả mô hình dự đoán dựa trên TF-IDF

- Kết quả dựa trên BoW:

Độ chính xác của mô hình trên tập Train: 93.35%				
Thời gian huấn luyện trên tập Train: 10.7780 giây				
classification_report:				
	precision	recall	f1-score	support
Negative	0.57	0.55	0.56	725
Neutral	0.41	0.30	0.34	1022
Positive	0.91	0.95	0.92	6785
accuracy			0.83	8532
macro avg	0.63	0.60	0.61	8532
weighted avg	0.82	0.83	0.82	8532
Thời gian dự đoán trên tập Test: 0.0056 giây				

Hình 26: Kết quả mô hình dự đoán dựa trên BoW

- So sánh kết quả:



Hình 27: Confusion matrix của mô hình Logistic Regression

Kết quả Confusion matrix thu được như sau:

❖ 'TF-IDF

Có tất cả 7235 reviews trên tổng số 8532 reviews trong tập kiểm thử được dự đoán chính xác. Trong đó:

- 382 reviews lớp Negative và được dự đoán đúng vào lớp Negative.
- 235 reviews lớp Neutral và được dự đoán đúng vào lớp Neutral.
- 6618 reviews lớp Positive và được dự đoán đúng vào lớp Positive.

Đa số các reviews thuộc lớp Negative và Neutral đều bị dự đoán sai sang lớp Positive:

- 212 reviews lớp Negative nhưng mô hình dự đoán sai sang lớp Positive.
- 652 reviews lớp Neutral nhưng mô hình dự đoán sai sang lớp Positive.

❖ BoW

Có tất cả 7116 reviews trên tổng số 8532 reviews trong tập kiểm thử được dự đoán chính xác. Trong đó:

- 397 reviews lớp Negative và được dự đoán đúng vào lớp Negative.
- 304 reviews lớp Neutral và được dự đoán đúng vào lớp Neutral.
- 6415 reviews lớp Positive và được dự đoán đúng vào lớp Positive.

Đa số các reviews thuộc lớp Negative và Neutral đều bị dự đoán sai sang lớp Positive:

- 146 reviews lớp Negative nhưng mô hình dự đoán sai sang lớp Positive.

- 525 reviews lớp Neutral nhưng mô hình dự đoán sai sang lớp Positive.

<i>Logistic Regression</i>	<i>TF-IDF</i>	<i>BoW</i>
<i>Precision</i>	0.68	0.63
<i>Recall</i>	0.58	0.60
<i>F_score</i>	0.61	0.61
<i>Accuracy</i>	0.85	0.83
<i>Thời gian dự đoán trên tập test (giây)</i>	0.0052	0.0039

Bảng 3: Kết quả thu được từ mô hình Logistic Regression

Dựa vào kết quả đánh giá giữa 2 phương pháp trích chọn đặc trưng, có thể rút ra kết luận mô hình Logistic Regression có độ chính xác cao hơn khi thực hiện với Bag of Word.

3. Mô hình học sâu (Deep Learning)

3.1. Giới thiệu về mô hình học sâu (Deep Learning)

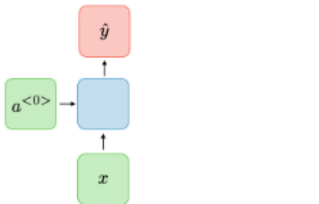
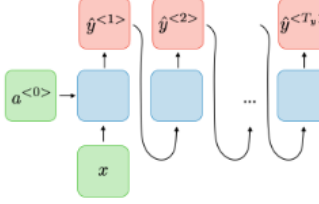
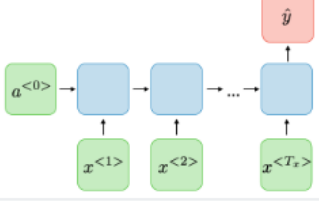
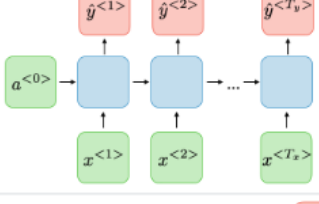
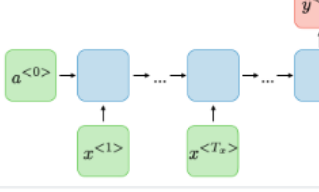
Học sâu trong lĩnh vực trí tuệ nhân tạo (AI) dạy máy tính xử lý dữ liệu theo cách được lấy cảm hứng từ bộ não con người. Mô hình học sâu có khả năng nhận diện các mẫu phức tạp trong ảnh, văn bản, âm thanh và dữ liệu khác để tạo ra thông tin sâu hơn và dự đoán chính xác. Mạng nơ-ron là một thành phần quan trọng trong học sâu.

Ý nghĩa của việc áp dụng Deep Learning vào phân lớp văn bản:

- Mang lại độ chính xác cao: Deep Learning đã được chứng minh là có thể đạt được độ chính xác cao trong phân loại văn bản, vượt xa các phương pháp truyền thống.
- Khả năng xử lý dữ liệu lớn, điều này có thể giúp cải thiện độ chính xác của phân loại văn bản.
- Deep Learning có thể học hỏi các đặc trưng phức tạp trong văn bản, điều này có thể giúp cải thiện độ chính xác của phân loại văn bản.

- Thời gian tính toán không nhanh bằng cách Neural Network khác.

3.3. Ứng dụng của RNN

Các loại RNN	Hình minh họa	Ví dụ
Một-Một $T_x = T_y = 1$		Mạng neural truyền thống
Một-nhiều $T_x = 1, T_y > 1$		Sinh nhạc
Nhiều-một $T_x > 1, T_y = 1$		Phân loại ý kiến
Nhiều-nhiều $T_x = T_y$		Ghi nhận thực thể tên
Nhiều-nhiều $T_x \neq T_y$		Dịch máy

Hình 29: Các ứng dụng của RNN trong xử lý ngôn ngữ tự nhiên²

3.4. Áp dụng mô hình RNN vào dữ liệu

3.4.1. Tiến hành chuẩn bị dữ liệu:

Nhóm trích xuất nội dung reviews từ cột "Review_Text" và chuyển thành dạng list. và sau đó sử dụng pad_sequences để đưa các chuỗi văn bản về cùng độ dài (max_words = 500). Với input X là ma trận chứa các chuỗi văn bản đã xử lý và output y là ma trận chứa các label cảm xúc tương ứng (Negative, Neutral, Positive)

² CS 230 - Mạng nơ-ron hồi quy cheatsheet (stanford.edu)

được biểu diễn dưới dạng one-hot encoding. Kết quả thu được trong tập dữ liệu có 42656 mẫu.

```
tokenizer = Tokenizer()
# chuyển đổi reviews thành list
reviews_to_list = df_main_class['Review_Text'].tolist()
tokenizer.fit_on_texts(reviews_to_list)
# Tạo chuỗi văn bản
text_sequences = np.array(tokenizer.texts_to_sequences(reviews_to_list))

# one hot encoding
df_main_class = pd.get_dummies(df_main_class, columns = ['Sentiment'])

# setting maximum words
max_words = 500

# X (input) to the model
# using pad_sequences and y (output)
X = pad_sequences(text_sequences, maxlen = max_words)
y = df_main_class[['Sentiment_Negative', 'Sentiment_Neutral', 'Sentiment_Positive']]
print(X.shape, y.shape)
```

Hình 30: Chuẩn bị dữ liệu cho huấn luyện RNN

3.4.2. Chuẩn bị mô hình RNN

```
# Creating a RNN model
rnn = Sequential(name="Simple_RNN")
rnn.add(Embedding(len(tokenizer.word_index)+1,
                  max_words,
                  input_length=max_words))

rnn.add(SimpleRNN(128, activation='relu', return_sequences=True))

rnn.add(SimpleRNN(64, activation='relu', return_sequences=False))

rnn.add(Dense(3, activation='softmax'))

print(rnn.summary())
```

Hình 31: Chuẩn bị mô hình RNN

Nhóm khởi tạo mô hình RNN đơn giản với hai lớp SimpleRNN được xếp chồng lên nhau với số nơ - ron ẩn lần lượt là 128 và 64 sử dụng hàm kích hoạt 'relu', kết hợp với lớp Embedding ở đầu và lớp Dense với hàm kích hoạt softmax ở cuối để dự đoán xác suất cho cho mỗi loại đánh giá.

- Mô hình thu được như sau:

Model: "Simple_RNN"		
Layer (type)	Output Shape	Param #
embedding_1 (Embedding)	(None, 500, 500)	32304000
simple_rnn_2 (SimpleRNN)	(None, 500, 128)	80512
simple_rnn_3 (SimpleRNN)	(None, 64)	12352
dense_1 (Dense)	(None, 3)	195
=====		
Total params: 32397059 (123.58 MB)		
Trainable params: 32397059 (123.58 MB)		
Non-trainable params: 0 (0.00 Byte)		
=====		
None		

Hình 32: Mô hình thu được sau khi chuẩn bị

- Chia tập dữ liệu thành tập huấn luyện (80%) và tập kiểm tra (20%):

```

1 # Chia dữ liệu thành tập huấn luyện và tập kiểm tra với tỷ lệ 80-20
2 train_x, test_x, train_y, test_y = train_test_split(X, y, test_size=0.2, stratify= y,
3                                                    random_state=42)
4
5 print('Tập Train (80%): ', train_x.shape)
6 print('Tập Test (20%): ', test_x.shape)

```

Tập Train (80%): (34124, 500)
Tập Test (20%): (8532, 500)

Hình 33: Chia tập dữ liệu huấn luyện và kiểm tra

3.4.3. Huấn luyện mô hình

Trong quá trình huấn luyện một mô hình RNN, nhóm sử dụng hàm mất mát (loss function) là 'categorical_crossentropy', 'adam' làm tối ưu hóa, và sử dụng Early Stopping để ngừng quá trình huấn luyện nếu không có cải thiện đáng kể trong việc giảm độ lỗi trên tập xác thực sau số epochs quy định để tránh overfitting.

Cuối cùng, tiến hành huấn luyện mô hình.


```

# Compiling model
rnn.compile(
    loss="categorical_crossentropy",
    optimizer='adam',
    metrics=['accuracy']
)

#Early Stopping
early_stopping = EarlyStopping(monitor='val_loss', patience=3, restore_best_weights=True)
# Training the model
history = rnn.fit(train_x, train_y,
                  batch_size=64,
                  epochs=50,
                  verbose=1,
                  validation_data = (test_x, test_y),
                  callbacks=[early_stopping])

print("Simple_RNN Score--> ", rnn.evaluate(test_x, test_y, verbose=1))

```

Hình 34: Set cấu hình cho mô hình RNN

- Kết quả sau khi huấn luyện mô hình:

```

Epoch 1/50
534/534 [=====] - 1060s 2s/step - loss: 0.5050 - accuracy: 0.8108 - val_loss: 0.5340 - val_accuracy: 0.8138
Epoch 2/50
534/534 [=====] - 1031s 2s/step - loss: 0.3819 - accuracy: 0.8457 - val_loss: 0.4571 - val_accuracy: 0.8312
Epoch 3/50
534/534 [=====] - 1025s 2s/step - loss: 0.2977 - accuracy: 0.8834 - val_loss: 0.6097 - val_accuracy: 0.8156
Epoch 4/50
534/534 [=====] - 1032s 2s/step - loss: 0.2129 - accuracy: 0.9167 - val_loss: 0.6134 - val_accuracy: 0.7958
Epoch 5/50
534/534 [=====] - 1031s 2s/step - loss: 0.1540 - accuracy: 0.9429 - val_loss: 0.6867 - val_accuracy: 0.8100
267/267 [=====] - 42s 158ms/step - loss: 0.4571 - accuracy: 0.8312
Simple_RNN Score--> [0.45712682604789734, 0.8312236070632935]

```

Hình 35: Kết quả sau khi huấn luyện mô hình

	Epoch 1	Epoch 2	Epoch 3	Epoch 4	Epoch 5	Kết quả cuối cùng của RNN
loss (Giá trị mất mát trung bình trên tập train)	0,5050	0.3819	0,2977	0,2129	0,1540	0,4571
accuaracy (Độ chính xác trung bình trên tập train)	0,8108	0,8457	0,8834	0,9167	0,9429	0,8312
val_loss (Giá trị mất mát trung bình trên tập validation)	0,5340	0,4475	0,6097	0,6134	0,6867	
val_accuaracy (Độ chính xác trung bình trên tập validation)	0,8138	0,8312	0,8156	0,7958	0,8100	

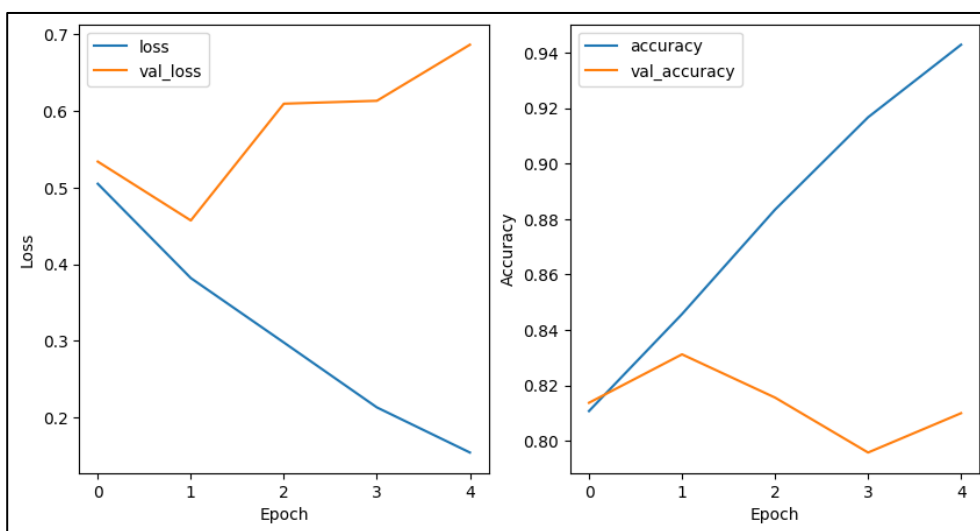
Bảng 4: So sánh giá trị loss và accuracy cho từng epoch

Có thể thấy, độ chính xác (accuracy) trên tập train tăng từ 81,08% trong epoch 1 lên 94,29% trong epoch 5. Nhưng bên cạnh đó, độ chính xác trên tập validation tăng lên ở giai đoạn đầu và giảm sau epoch 3 và giá trị loss cuối cùng trên tập test (0.4571)

cao hơn giá trị loss trên tập train (0.1540) cho thấy rõ ràng overfitting. Vì nhóm sử dụng early stopping nên mô hình đã dừng sau khi huấn luyện qua 5 epochs.

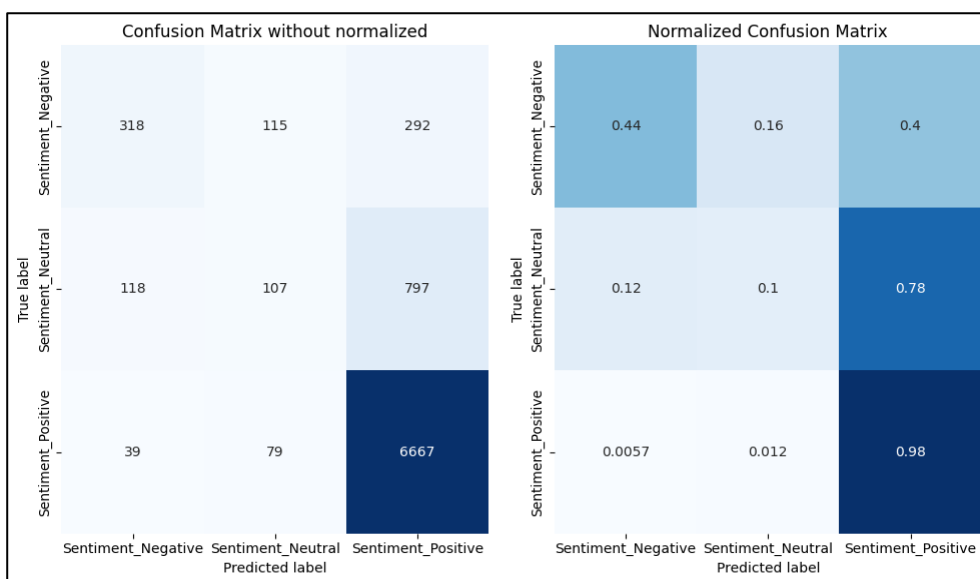
Nhóm rút ra được kết quả cuối cùng cho mô hình RNN với kết quả tương tự như khi huấn luyện trên tập test là giá trị mất mát trung bình: 0.4571 và độ chính xác: 83.12%

3.4.4. Trực quan kết quả sau khi huấn luyện mô hình:



Hình 36: Trực quan kết quả sau khi huấn luyện mô hình.

Để có cái nhìn tổng quan hơn về kết quả đánh giá độ chính xác của mô hình trên tập kiểm thử (Test), nhóm sử dụng Confusion matrix để trực quan hóa kết quả thể hiện trong Hình 34.



Hình 37: Trực quan kết quả dự đoán trên tập kiểm thử với Confusion matrix.

Có tất cả 7092 reviews trong tập kiểm thử được dự đoán chính xác. Trong đó:

- 318 reviews lớp Negative được dự đoán đúng vào lớp Negative (44%).
- 107 reviews lớp Neutral được dự đoán đúng vào lớp Neutral (10%).
- 6667 reviews lớp Positive và được dự đoán đúng vào lớp Positive (98%).

Đa số các reviews thuộc lớp Negative và Neutral đều bị dự đoán sai sang lớp Positive:

- 292 reviews lớp Negative nhưng mô hình dự đoán sai sang lớp Positive (40%).
- 797 reviews lớp Neutral nhưng mô hình dự đoán sai sang lớp Positive (78%).

PHẦN 4: ĐÁNH GIÁ MÔ HÌNH VỚI DỮ LIỆU CRAWL

I. Tổng quan dữ liệu

Bộ dữ liệu `amazon_reviews.csv` được nhóm đào trực tiếp từ dữ liệu đánh giá sản phẩm trên sàn thương mại điện tử Amazon. Nhóm sử dụng thư viện BeautifulSoup để trích xuất dữ liệu ngẫu nhiên từ trang web.

Bên cạnh đó, nhóm sử dụng ba thư viện nữa là: ‘requests’ để thực hiện các yêu cầu HTTP đến trang web và truy xuất nội dung của chúng, ‘pandas’ sử dụng để thao tác và phân tích dữ liệu, tạo DataFrames và ‘datetime’ cung cấp các công cụ để làm việc với ngày tháng và thời gian.

Bộ dữ liệu thu được sau khi đào được bao gồm 5 biến và 1400 quan sát và không có dữ liệu bị thiếu.

```
1 data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1400 entries, 0 to 1399
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype
---  ---
0   Name             1400 non-null   object
1   Rating           1400 non-null   float64
2   Title            1400 non-null   object
3   Date             1400 non-null   object
4   Review_Text      1400 non-null   object
dtypes: float64(1), object(4)
memory usage: 54.8+ KB
```

Hình 38: Kết quả kiểm tra dữ liệu bị thiếu

II. Xử lý dữ liệu

Với dữ liệu `amazon_reviews` vừa thu thập, nhóm thực hiện các bước tiền xử lý tương tự bộ dữ liệu DisneyLand Reviews được lấy từ Kaggle để đảm bảo kết quả thực nghiệm. Tương tự như trên, với mục tiêu đề án, nhóm cũng sử dụng biến ‘Reviews_Text’ và biến mục tiêu ‘Rating’ để đưa vào mô hình phân lớp

III. Đánh giá mô hình với dữ liệu crawl

Để đảm bảo tính chính xác khi so sánh kết quả mô hình với 2 bộ dữ liệu, với bộ dữ liệu `amazon_reviews` khi đưa vào các mô hình phân lớp cũng được giữ nguyên các thang đo như phần 3.

Kết quả sau khi thực hiện việc tách tập dữ liệu data sẽ thu được tập huấn luyện chứa 11120 reviews (chiếm 80%), tập kiểm thử chứa 280 reviews (chiếm 20%).

```
[ ] 1 X = df_main_class['Review_Text']
    2 y = df_main_class['Sentiment']
    3
    4 # Chia dữ liệu thành tập huấn luyện và tập kiểm tra với tỷ lệ 80-20
    5 train_x, test_x, train_y, test_y = train_test_split(X, y, test_size=0.2, stratify=y, random_state=42)
    6
    7 print('Tập Train (80%): ', train_x.shape)
    8 print('Tập Test (20%): ', test_x.shape)
```

Tập Train (80%): (1120,)
Tập Test (20%): (280,)

Hình 39: Phân tách tập dữ liệu thành tập Train – Test

⇒ Kết quả thu được tổng hợp như sau:

Thuật toán phân lớp	MultinomialNB (alpha = 0.1)		Logistic Regression		RNN
	BoW	TF-IDF	BoW	TF-IDF	
Precision	1.00	1.00	1.00	1.00	1.00
Recall	1.00	1.00	1.00	1.00	1.00
F_score	1.00	1.00	1.00	1.00	1.00
Accuracy	1.00	1.00	1.00	1.00	1.00
Thời gian (giây)	0.0007	0.0008	0.0004	0.0153	1.16

Bảng 5: Kết quả so sánh kết quả các mô hình phân lớp với amazon_reviews.csv

Với bộ dữ liệu này kết quả mô hình thu về khá tốt, kết quả các chỉ số đo độ chính xác thu về đều đạt 100%. Tuy nhiên, thời gian chạy của từng mô hình khác nhau - thang đo duy nhất có thể sử dụng để so sánh kết quả mô hình trong phần này. Thời gian chạy dài nhất vẫn là mô hình học sâu RNN (1.16 giây). Phương pháp chuẩn hóa dữ liệu bằng Bag of Word vẫn đạt kết quả tốt hơn TF-IDF. Dựa vào đó, nhóm nhận xét mô hình Logistic Regression chuẩn hóa bằng Bag of Word là mô hình tốt nhất (0.0004 giây) đạt được phần này, tương tự như kết quả thu được ở phần 3.

PHẦN 5: ĐÁNH GIÁ KẾT QUẢ VÀ KẾT LUẬN

I. So sánh kết quả của các mô hình

Thuật toán phân lớp	MultinomialNB ($\alpha = 0.1$) (BoW)	Logistic Regression (BoW)	RNN
<i>Precision</i>	0.60	0.63	0.63
<i>Recall</i>	0.62	0.60	0.51
<i>F_score</i>	0.61	0.61	0.54
<i>Accuracy</i>	0.82	0.83	0.83
<i>Thời gian (giây)</i>	0.0055	0.0039	42

Bảng 6: So sánh kết quả các mô hình trên tập test của bộ dữ liệu Disneyland

Dựa vào bảng so sánh kết quả của 3 mô hình phân lớp trên tập test thu được kết quả:

- Mô hình Logistic Regression thực hiện chuẩn hóa dữ liệu bằng phương pháp Bag of Word tốt nhất. Với độ chính xác 83% với thời gian nhanh nhất và các chỉ số cũng có kết quả cao nhất.
- Mô hình học sâu RNN cũng có độ chính xác tương đương Logistic Regression. Tuy nhiên, thời gian thực hiện thuật toán này chậm nhất.

Thuật toán phân lớp	MultinomialNB ($\alpha = 0.1$) (BoW)	Logistic Regression (BoW)	RNN
<i>Precision</i>	1.00	1.00	1.00
<i>Recall</i>	1.00	1.00	1.00
<i>F_score</i>	1.00	1.00	1.00
<i>Accuracy</i>	1.00	1.00	1.00
<i>Thời gian (giây)</i>	0.0007	0.0004	1.16

Bảng 7: So sánh kết quả các mô hình trên tập test của bộ dữ liệu crawl Amazon

Với kết quả tương tự bảng 6 từ dữ liệu Kaggle trên, kết quả mô hình của bộ dữ liệu Crawl cũng cho ra được mô hình Logistic Regression thực hiện chuẩn hóa dữ liệu bằng phương pháp Bag of Word tốt nhất.

II. Tóm tắt kết quả và nhận xét

Thứ nhất, nhóm đã thực hiện phân phân tích cảm xúc (Sentiment Analysis) cho bộ dữ liệu Disneyland ở phần 2. Từ đó, nhóm đã phân tích được cảm xúc của khách hàng cũng như một số sự khác biệt giữa các chi nhánh của Disneyland. Qua quá trình thực hiện phân tích, nhóm cũng đã tìm hiểu và biết thêm nhiều kiến thức và phương pháp phân tích dữ liệu trong phần này.

Thứ hai, nhóm thực hiện 3 mô hình học máy có giám sát: Naive Bayes (Multinomial Naive Bayes), Logistic Regression từ scikit-learn và mô hình học sâu RNN. 3 mô hình này được sử dụng để phân loại sắc thái cảm xúc cho bài toán phân tích đánh giá và bình luận của du khách tới Disneyland và các reviews sản phẩm trên sàn thương mại điện tử Amazon, theo 3 mức độ cảm xúc: Negative, Neutral và Positive. Dữ liệu thực nghiệm là các review về mức độ hài lòng về chất lượng dịch vụ tại Disneyland và nhận xét chất lượng sản phẩm amazon của khách hàng bằng văn bản tiếng Anh.

Đồng thời nhóm đã ứng dụng những kiến thức đã học về Xử lý ngôn ngữ tự nhiên để thực hiện Text Mining (khai thác và xử lý dữ liệu văn bản)

III. Những hạn chế và hướng phát triển

1. Hạn chế

- Kết quả mô hình sau mỗi lần chạy sẽ khác nhau, nhóm chỉ thực hiện bài đánh giá trên một lần chạy.
- Nhóm chỉ chia dữ liệu thành 3 mức độ cảm xúc, trên thực tế thì nhiều hơn.
- Nhóm chưa khai thác và sử dụng được hết các biến trong bộ dữ liệu như trong việc phân loại phản hồi của du khách theo từng địa điểm, từng nhóm du khách khác nhau để có thể phân tích nguyên nhân của các đánh giá tiêu cực đồng thời đưa ra hướng cải thiện chất lượng dịch vụ.
- Vì không có quá nhiều thời gian nghiên cứu nên bộ dữ liệu đầu vào mà nhóm sử dụng là bộ dữ liệu có sẵn trên Kaggle mà chưa thể khai thác và thu thập dữ liệu đầu vào theo thời gian thực.

- Mô hình học sâu chạy mất thời gian khá lâu và có thể chưa mang lại tính chính xác hoàn toàn vì mỗi lần chạy mô hình cho ra mỗi kết quả khác nhau.
- Bộ dữ liệu crawl từ sản Amazon đang tương đối ít quan sát, có thể dẫn đến kết quả không chính xác khi thực hiện so sánh mô hình.

2. Hướng phát triển

Từ việc phân tích phản hồi của du khách theo biến phân loại Negative, Neutral và Positive, nhóm có thể phát triển xây dựng mô hình học máy phân loại đa lớp (với biến phân loại Rating 1-5 sao hoặc theo nhiều loại cảm xúc khác nhau: giận dữ, hạnh phúc, buồn bã, bất ngờ,...). Ngoài ra có thể bổ sung thêm thông tin về khách hàng để xây dựng mô hình phân lớp đánh giá phản hồi của khách hàng theo từng nhóm khách hàng, hoặc theo địa điểm khác nhau,...

Vấn đề gia tăng độ chính xác có thể giải quyết bằng việc kết hợp thuật toán phân lớp để gia tăng độ chính xác đồng thời không ảnh hưởng quá nhiều tốc độ xử lý thông tin

Việc xử lý ngôn ngữ tự nhiên hiện tại nhóm chỉ phân tích trên ngôn ngữ là tiếng Anh. Trong tương lai có thể phân tích nhận diện cảm xúc của khách hàng bằng tiếng Việt (tham khảo và áp dụng các mô hình trong các dự án của nhóm Vietnamese NLP Research Group - Under The Sea). Trong quá trình thu thập phản hồi khách hàng có thể lọc ra những phản hồi không phù hợp (toxic comment classification) để dự đoán liệu phản hồi của khách hàng có thật sự phù hợp hay không.

PHẦN 6: GIAO DIỆN DEMO

I. Thư viện Gradio và Interface

Gradio là một thư viện Python mạnh mẽ được sử dụng để xây dựng giao diện người dùng (UI) cho các mô hình máy học và các ứng dụng liên quan đến dữ liệu. Nó cho phép tạo ra các giao diện người dùng tương tác cho các mô hình máy học mà không cần kiến thức vững về phát triển giao diện người dùng.

Thư viện Gradio có thể sử dụng linh hoạt trên nhiều dạng tập lệnh Python, notebook Jupyter hoặc Colaboratory.

```
!pip install -q comet_ml gradio
import gradio as gr
```

Hình 40: *install và import Gradio trên colaboratory*

Interface: một API cấp cao cho phép bạn tạo một bản demo học máy đầy đủ chỉ đơn giản bằng cách cung cấp danh sách các đầu vào và đầu ra.

II. Xây dựng giao diện

Nhóm sử dụng Gradio để tạo giao diện đơn giản để dự đoán cảm xúc của văn bản đầu vào. Khi chạy giao diện này sẽ có một ô văn bản để nhập văn bản cần phân loại cảm xúc. Kết quả sẽ là cảm xúc dự đoán của văn bản đó.

Giao diện được xây dựng dựa theo mô hình tốt nhất sau khi thực hiện xây dựng và đánh giá các mô hình phân lớp văn bản ở các phần trên đã thực hiện ở phần trên. Đó là mô hình Logistic Regression chuẩn hóa văn bản bằng phương pháp Bag of Words và thiết lập các chỉ số như trước.

Dữ liệu huấn luyện được sử dụng từ kết quả sau khi tiền xử lý dữ liệu của bộ dữ liệu DisneylandReviews.csv được thực hiện ở phần trên.

```

# Hàm huấn luyện mô hình
def train_model(train_text, train_labels):
    # Tạo một vectorizer để chuyển đổi văn bản thành biểu diễn Bow
    vectorizer = CountVectorizer(analyzer='word', max_features=20000, stop_words='english')
    # Huấn luyện vectorizer trên dữ liệu huấn luyện
    vectorizer.fit(df_main_class['Review_Text'])
    # Chuyển đổi dữ liệu huấn luyện và kiểm tra thành biểu diễn Bow
    xtrain_bow = vectorizer.transform(train_x)
    xtest_bow = vectorizer.transform(test_x)
    # Khởi tạo và huấn luyện mô hình logistic regression
    LogReg = LogisticRegression(penalty='l2', C=1.0, solver='lbfgs', max_iter=100)
    LogReg.fit(xtrain_bow, train_y)
    # Trả về mô hình đã được huấn luyện và vectorizer
    return LogReg, vectorizer

# Hàm dự đoán cảm xúc
def predict_sentiment(text, model, vectorizer):
    # Chuyển đổi văn bản thành biểu diễn Bow bằng cách sử dụng vectorizer
    text_vectorized = vectorizer.transform([text])
    # Dự đoán cảm xúc bằng cách sử dụng mô hình đã được huấn luyện
    prediction = model.predict(text_vectorized)[0]
    # Trả về dự đoán
    return prediction

# Huấn luyện mô hình
train_text = df_main_class['Review_Text']
train_labels = df_main_class['Sentiment']
model, vectorizer = train_model(train_text, train_labels)

```

Hình 41: Code xây dựng giao diện demo

The image shows a web-based user interface for a sentiment prediction demo. It consists of two main input/output areas. On the left, under the label 'text', there is a text input field and a 'Clear' button below it. In the center, there is an orange 'Submit' button. On the right, under the label 'output', there is a text output field and a 'Flag' button below it. The interface is clean and uses a light gray color scheme for buttons and labels.

Hình 42: Giao diện demo

PHỤ LỤC

1. Source code: [Bigdata - Google Drive](#)

2. Bảng phân công nhiệm vụ

Họ và tên	Nhiệm vụ	Mức độ đóng góp
Võ Tuấn Cường	<ol style="list-style-type: none">1. Mục tiêu nghiên cứu2. Phương pháp nghiên cứu3. Mô hình Logistic Regression4. Sentiment Analysis5. Emotions Analysis6. Giao diện demo7. Phần 5: Hạn chế và hướng phát triển8. Làm slide thuyết trình	100%
Nguyễn Thị Thom	<ol style="list-style-type: none">1. Tiền xử lý dữ liệu.2. Mô hình Naïve Bayes và Deep Learning.3. Phần 4: Đánh giá mô hình với dữ liệu crawl.4. Crawl data từ Amazon5. Phần 5: I, II: So sánh và tóm tắt kết quả6. Xây dựng giao diện demo7. Làm slide thuyết trình8. Leader và tổng hợp đồ án.	100%
Lưu Trọng Tốt	<ol style="list-style-type: none">1. Lời mở đầu2. Giới thiệu vấn đề3. Crawl dữ liệu4. Giao diện demo5. Làm slide	60%

3. Danh mục hình ảnh

Hình 1: Kết quả kiểm tra dữ liệu bị thiếu.....	4
Hình 2: Gán nhãn cột Rating.....	4
Hình 3: Tiền xử lý dữ liệu cột Review_Text	5
Hình 4: Word Cloud của bộ dữ liệu	7
Hình 5: Biểu đồ thể hiện số lượng reviews theo sentiment type	7
Hình 6: Biểu đồ thể hiện tính khách quan.....	8
Hình 7: Biểu đồ thể hiện tính chủ quan	9
Hình 8: Biểu đồ Figure về 2 mức độ polarity và subjectivity.....	9
Hình 9: Biểu đồ phân phối VADER theo địa điểm.....	10
Hình 10: Biểu đồ phân phối NLTK theo địa điểm.....	11
Hình 11: Thực hiện phân tích emotion	11
Hình 12: Kết quả phân tích emotion	12
Hình 13: Biểu đồ trực quan emotion tại Disneyland ở Paris	12
Hình 14: Biểu đồ trực quan emotion tại Disneyland ở Hongkong	13
Hình 15: Biểu đồ trực quan emotion tại Disneyland ở Paris	13
Hình 16: Phân tách tập dữ liệu thành tập Train - Test	14
Hình 17: Kết quả vector hóa Review_Text sang dạng số sử dụng TF_IDF	15
Hình 18: Sử dụng Grid Search tìm ra alpha có chỉ số accuracy cao nhất.....	17
Hình 19: Xây dựng mô hình phân lớp văn bản sử dụng MultinomialNB.....	18
Hình 20: Kết quả mô hình Naive Bayes sử dụng MultinomialNB	18

Hình 21: Kết quả dự đoán trên tập test của Naïve Bayes với Confusion matrix .	19
Hình 22: Mã code tìm tham số tốt nhất cho Logistic Regression	21
Hình 23: Áp dụng Logistic Regression với TF-IDF	21
Hình 24: Áp dụng Logistic Regression với BoW	21
Hình 25: Kết quả mô hình dự đoán dựa trên TF-IDF	22
Hình 26: Kết quả mô hình dự đoán dựa trên BoW	22
Hình 27: Confusion matrix của mô hình Logistic Regression.....	23
Hình 28: Mô hình RNN cơ bản.....	25
Hình 29: Các ứng dụng của RNN trong xử lý ngôn ngữ tự nhiên	26
Hình 30: Chuẩn bị dữ liệu cho huấn luyện RNN	27
Hình 31: Chuẩn bị mô hình RNN	27
Hình 32: Mô hình thu được sau khi chuẩn bị.....	28
Hình 33: Chia tập dữ liệu huấn luyện và kiểm tra	28
Hình 34: Set cấu hình cho mô hình RNN	29
Hình 35: Kết quả sau khi huấn luyện mô hình.....	29
Hình 36: Trực quan kết quả sau khi huấn luyện mô hình.	30
Hình 37: Trực quan kết quả dự đoán trên tập kiểm thử với Cofusion matrix.....	30
Hình 38: Kết quả kiểm tra dữ liệu bị thiếu.....	32
Hình 39: Phân tách tập dữ liệu thành tập Train – Test	33
Hình 40: install và import Gradio trên colabatory	37
Hình 41: Code xây dựng giao diện demo.....	38
Hình 42: Giao diện demo	38

2. Danh mục bảng biểu

Bảng 1: Thuộc tính trong dữ liệu	3
Bảng 2: Kết quả thu được từ mô hình Naive Bayes	20
Bảng 3: Kết quả thu được từ mô hình Logistic Regression	24
Bảng 4: So sánh giá trị loss và accuracy cho từng epoch.....	29
Bảng 5: Kết quả so sánh kết quả các mô hình phân lớp với amazon_reviews.csv ..	33
Bảng 6: So sánh kết quả các mô hình trên tập test của bộ dữ liệu Disneyland	34
Bảng 7: So sánh kết quả các mô hình trên tập test của bộ dữ liệu crawl Amazon...	34

TÀI LIỆU THAM KHẢO

- [1] Amidi, A., & Amidi, S. (2018). Vip cheatsheet: Recurrent neural networks.
- [2] It, E. (2022, April 18). *Picking a Disneyland — Natural Language Processing of Reviews. Part 2*. CodeX. <https://medium.com/codex/picking-a-disneyland-natural-language-processing-of-reviews-part-2-a74658c2b16d>
- [3] Nguyễn,Â.T.(2017).
Microsoft Word - SA Twitter Tomtac NguyenThaiAn.docx (jvtek.com.vn)
- [4] Cruz, M. D. (2021, June 24). *Analyzing Disneyland reviews using NLP*. Medium.
Analyzing Disneyland Reviews with NLP | by Marielle Dela Cruz | Towards Data Science
- [5] Deepanshi. (2023, April 26). *Text preprocessing in NLP with python codes*. Analytics Vidhya. Text Preprocessing in NLP with Python codes (analyticsvidhya.com)
- [6] Nhữ Bảo, V. (2017). *Xây dựng mô hình đối thoại cho tiếng Việt trên miền mở dựa vào phương pháp học chuỗi liên tiếp* (Doctoral dissertation).
- [7] *Amazon Product Review Sentiment Analysis using RNN*. (2023a, June 30). GeeksforGeeks. <https://www.geeksforgeeks.org/amazon-product-review-sentiment-analysis-using-rnn/>
- [8] Đỗ, H. Đ. (2015). *Phân loại cảm xúc người dùng trong mạng xã hội* (Doctoral dissertation, Học viện Công nghệ Bưu chính Viễn thông).
- [9] 3.2. Tuning the hyper-parameters of an estimator — scikit-learn 1.3.2 documentation
- [10] 1.9. Naive Bayes — scikit-learn 1.3.2 documentation
- [11] sklearn.linear_model.LogisticRegression — scikit-learn 1.3.2 documentation
- [12] Sharda, R., Delen, D., & Turban, E. (2018). *Business intelligence, analytics, and data science: a managerial perspective*. pearson.