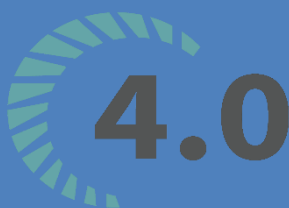


KHOA CÔNG NGHỆ THÔNG TIN
ĐẠI HỌC KHOA HỌC TỰ NHIÊN, ĐẠI HỌC QUỐC GIA TP HCM

MÔN HỌC THỐNG KÊ ĐỒ ÁN CUỐI KỲ



Sinh viên:

20120084 - Nguyễn Văn Hiếu

20120085 – Trần Xuân Hòa

HỌC THỐNG KÊ
HỌC KỲ I – NĂM HỌC 2023-2024



Mục lục

I. Giới thiệu đề tài	2
1. Lý do chọn đề tài	2
2. Chọn bộ dữ liệu	2
3. Mô tả bộ dữ liệu	2
4. Đầu ra dự kiến	3
II. Xây dựng mô hình Phân loại cảm xúc	3
1. Tiền xử lý dữ liệu	3
1.1 Missing value	3
1.2 Duplicated Values	4
1.2 Data cleaning	4
2. Khám phá dữ liệu	5
3. Chọn đặc trưng	6
4. Modeling	8
4.1 Multinomial Naive Bayes	8
4.2 Bert Model	9
III. Xây dựng giao diện	12
IV. Tham khảo	18

I. Giới thiệu đề tài

1. Lý do chọn đề tài

Phân tích cảm xúc dựa trên dữ liệu văn bản là một trong những ứng dụng phổ biến của học máy và xử lý ngôn ngữ tự nhiên. Và nó mang lại nhiều lợi ích thực tiễn:

- Ứng dụng trong kinh doanh khi doanh nghiệp muốn biết ý kiến cũng như sự hài lòng của khách hàng về sản phẩm, dịch vụ của mình.
- Dự đoán xu hướng của thị trường, dựa trên cảm xúc của người dùng với sản phẩm, dịch vụ để điều chỉnh các chiến lược kinh doanh phù hợp.
- Có nhiều bộ dữ liệu về cảm xúc trên mạng xã hội, diễn đàn, trang web... giúp việc nghiên cứu và phát triển mô hình phân loại cảm xúc trở nên dễ dàng hơn.

2. Chọn bộ dữ liệu

Nhóm chọn bộ dữ liệu **Twitter tweets** vì:

- Có lượng dữ liệu lớn và đa dạng: Có hàng triệu tweet được tạo ra mỗi ngày trên Twitter, với nhiều chủ đề khác nhau. Tạo ra một nguồn dữ liệu lớn và đa dạng để phát triển và đánh giá mô hình Sentiment Analysis.
- Dữ liệu có ngôn ngữ đa dạng: Tweet thường được viết theo ngôn ngữ tự nhiên và đa dạng, bao gồm cả việc sử dụng ngôn ngữ không chuẩn, viết tắt và biểu cảm cảm xúc. Điều này tạo ra một thách thức thú vị cho việc phát triển các mô hình Sentiment Analysis có khả năng xử lý ngôn ngữ tự nhiên đa dạng.
- Tính ngắn gọn: Tweet có giới hạn ký tự 280, điều này yêu cầu người dùng phải tóm tắt ý kiến hoặc cảm xúc của họ trong một khoảng thời gian ngắn.
- Thời gian thực: Tweet thường được tạo ra và cập nhật liên tục, điều này giúp cho việc phát triển mô hình Sentiment Analysis có khả năng phản ứng nhanh với cảm xúc mới được thể hiện trên mạng xã hội.

3. Mô tả bộ dữ liệu

Nhóm sử dụng dữ liệu Twitter Tweets Sentiment dataset

<https://www.kaggle.com/datasets/yasserh/twitter-tweets-sentiment-dataset> được lấy từ Kaggle.

LICENSE của dữ liệu là **CC0: Public Domain**

<https://creativecommons.org/publicdomain/zero/1.0/>

cho phép bất kỳ ai sử dụng, sao chép, sửa đổi, phân phối và thậm chí kinh doanh tác phẩm hoặc dữ liệu đó mà không cần yêu cầu sự cho phép hay phải trả bất kỳ khoản phí nào.

Mô tả dữ liệu:

- Dữ liệu ghi lại các tweet trên Twitter với mỗi hàng là một tweet ghi lại nội dung bài viết của người dùng. Dữ liệu bao gồm 27418 dòng và 4 cột:

- textID: ID duy nhất cho mỗi đoạn văn bản.
- text: nội dung/văn bản của tweet.
- sentiment: nhãn cảm xúc của tweet.
- selected_text: một phần của văn bản được chọn để đại diện cho cảm xúc của tweet.

4. Đầu ra dự kiến

Xây dựng mô hình phân tích cảm xúc dựa trên dữ liệu có nhãn (neutral, negative, positive). Sau đó, xây dựng một giao diện web từ mô hình đã được huấn luyện để.

II. Xây dựng mô hình Phân loại cảm xúc

1. Tiền xử lý dữ liệu

1.1 Missing value

```
df.isnull().sum()
```

```
textID      0
text        1
selected_text 1
sentiment    0
dtype: int64
```

Qua kiểm tra thì dữ liệu 1 có giá trị thiếu ở 2 cột text và **selected_text**.

```
df[df.isnull().any(axis=1)]
```

	textID	text	selected_text	sentiment
314	fdb77c3752	NaN	NaN	neutral

Khi kiểm tra kỹ hơn thì ta thấy 2 giá trị thiếu ở 2 cột text và **selected_text** là ở cùng 1 dòng có thể là lỗi khi thu thập dữ liệu nên nhóm sẽ xóa dòng này. Tuy nhiên nó được gán nhãn **neutral**. Nên ở đây nhóm sẽ thay giá trị **NaN** thành giá trị trống.

```
df.fillna(' ',inplace=True)
df.isnull().sum()
```

```
textID      0
text        0
selected_text  0
sentiment   0
dtype: int64
```

1.2 Duplicated Values

```
df.duplicated().sum()
```

0

Qua kiểm tra thì không có hàng dữ liệu nào bị lặp trong tập dữ liệu.

1.2 Data cleaning

Trong các bài Tweet thường có các yếu tố như tên người dùng, đường dẫn (URL), biểu tượng cảm xúc (emojis) và các yếu tố không mong muốn khác. Việc xử lý trước dữ liệu (preprocessing) là rất quan trọng để cải thiện chất lượng mô hình phân tích cảm xúc.

Nhóm sẽ thêm các hàm xử lý:

- Xóa tên người dùng
- Xóa đường dẫn
- Xóa html
- Xóa emojis

- Xóa dấu câu
- Loại bỏ các khoảng trắng
- Chuyển về chữ thường

Sau đó, cho tất cả vào hàm `clean_text` để xử lý những vấn đề trên cũng như tiện cho việc làm giao diện về sau.

2. Khám phá dữ liệu

Tính điểm phân tích cảm xúc dựa trên `SentimentIntensityAnalyzer`, nó là một công cụ phân tích cảm xúc trong thư viện Natural Language Toolkit (NLTK) của Python. Nó được thiết kế để đánh giá và phân loại cảm xúc của văn bản đầu vào. Sử dụng phương thức `polarity_scores()` để đánh giá cảm xúc của một đoạn văn bản. Trả về các giá trị sau:

`'neg'`: Điểm số cảm xúc tiêu cực.

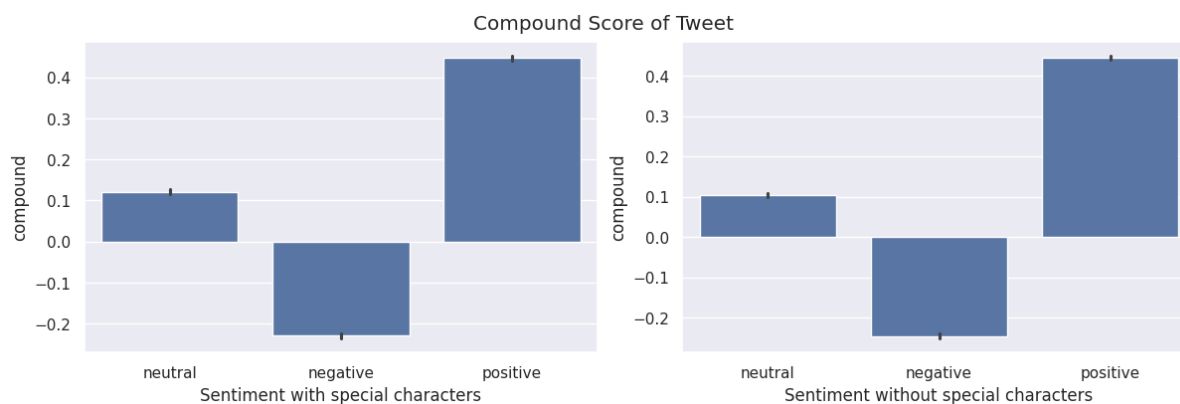
`'neu'`: Điểm số cảm xúc trung lập.

`'pos'`: Điểm số cảm xúc tích cực.

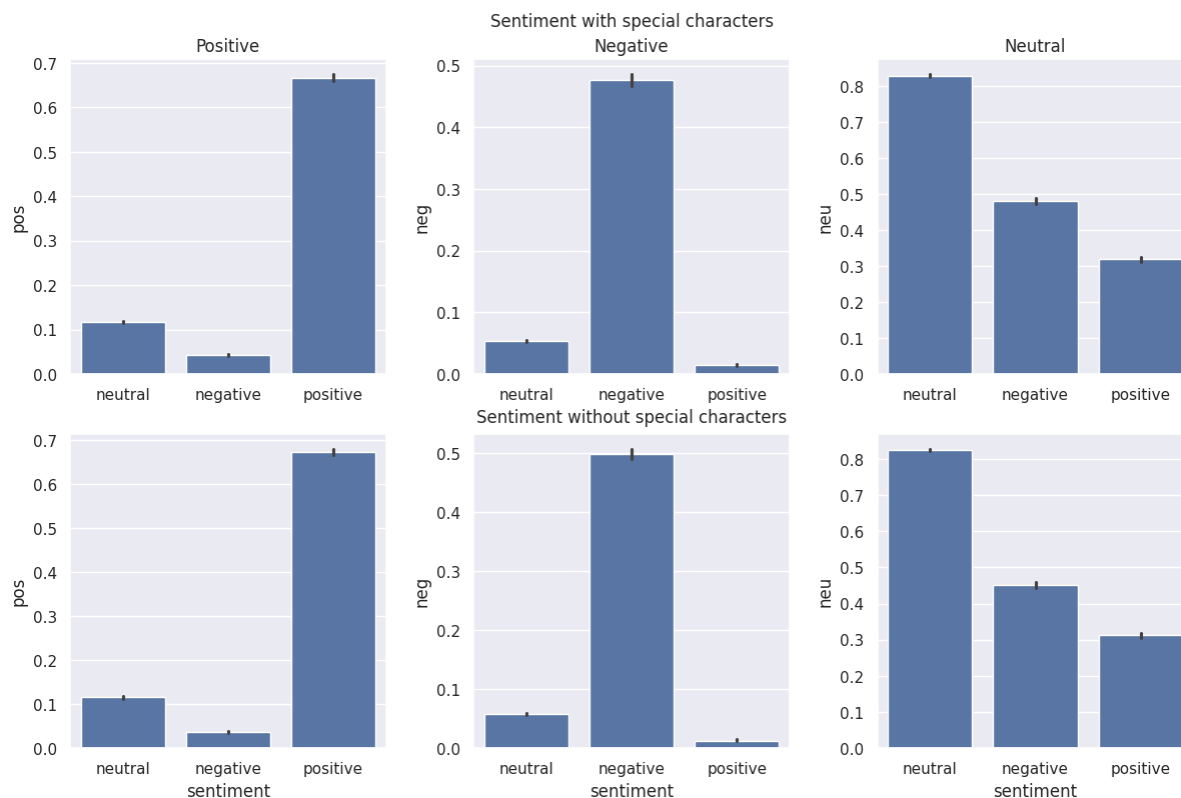
`'compound'`: Điểm số tổng hợp, đại diện cho cảm xúc tổng thể của văn bản (từ -1 đến 1, trong đó giá trị dương là cảm xúc tích cực và giá trị âm là cảm xúc tiêu cực).

→ Đánh giá nhanh chóng, tự động và có độ chính xác cao.

Tạo 2 DataFrame: 1 DataFrame phân tích cảm xúc của thuộc tính `selected_text` không thực hiện xóa các ký tự đặc biệt và có thực hiện xóa các ký tự đặc biệt. Để so sánh kết quả phân tích cảm xúc giữa 2 DataFrame.



Sentiment score of each components in a tweet



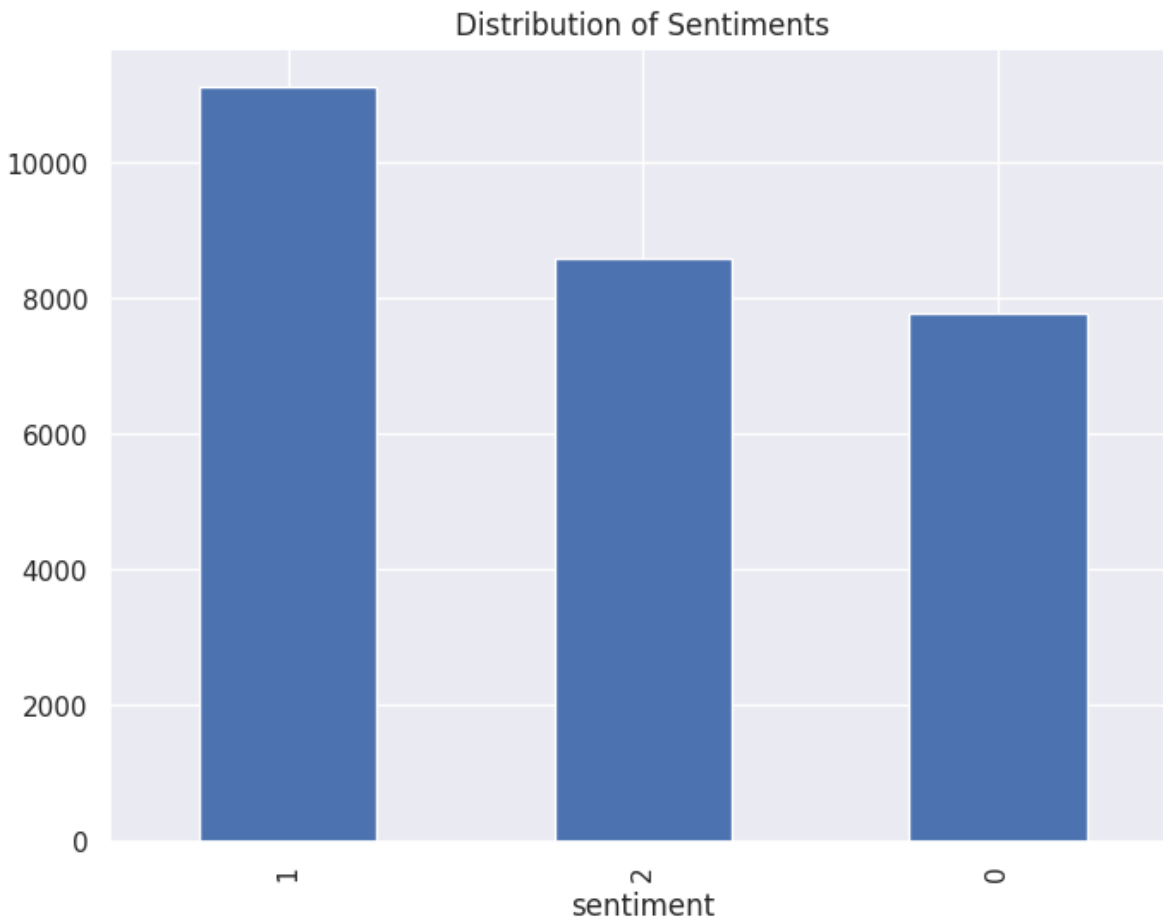
Qua 2 biểu đồ trên, ta thấy phân phối cảm xúc (Sentiment Analysis) của văn bản không thực hiện xóa các ký tự đặc biệt và có thực hiện xóa các ký tự đặc biệt không có sự thay đổi lớn. Điều này cho thấy việc xóa các ký tự đặc biệt không ảnh hưởng nhiều đến cảm xúc trong văn bản của người dùng.

→ Ta xóa đi các ký tự đặc biệt chỉ để giảm nhiễu khi train mô hình để tăng hiệu suất cùng độ chính xác của mô hình

3. Chọn đặc trưng

Chuẩn hóa nhãn dữ liệu:

```
df['sentiment'] = df['sentiment'].map({'neutral': 1, 'positive': 2, 'negative': 0})
```



Để tăng độ chính xác của mô hình ta sẽ chọn cột `selected_text` huấn luyện mô hình. Vì:

- `selected_text` chứa phần của văn bản được chọn để đại diện cho cảm xúc của tweet. Điều này có nghĩa là `selected_text` chứa những từ ngữ hoặc cụm từ mang ý nghĩa cảm xúc mạnh nhất trong tweet. Sử dụng phần này để huấn luyện mô hình giúp mô hình tập trung vào những từ ngữ quan trọng nhất liên quan đến cảm xúc, từ đó cải thiện độ chính xác của phân loại cảm xúc.
- Toàn bộ text của tweet có thể chứa nhiều thông tin không liên quan đến cảm xúc của người dùng. `selected_text` giúp loại bỏ các yếu tố này, giảm nhiễu và giúp mô hình tập trung vào thông tin cảm xúc cần thiết.
- Ngoài ra do `selected_text` chỉ lấy những cụm từ mang ý nghĩa cảm xúc từ trong text của Tweet nó làm tăng hiệu suất và độ chính xác của mô hình hơn.

4. Modeling

Ở đây nhóm sẽ sử dụng 2 mô hình để huấn luyện là Multinomial Naive Bayes và Bert để có thể có sự so sánh giữa 2 mô hình. Cũng như chọn mô hình tốt hơn để xây dựng giao diện.

4.1 Multinomial Naive Bayes

Multinomial Naive Bayes đặc biệt hiệu quả với dữ liệu văn bản, đặc biệt là khi áp dụng cho bài toán phân loại văn bản. Điều này là do mô hình này hoạt động dựa trên xác suất xuất hiện của từ trong tài liệu, điều này phù hợp với việc phân tích cảm xúc (sentiment analysis) nơi các từ ngữ cụ thể thường mang những trọng số cảm xúc nhất định.

Quá trình:

- Chia dữ liệu thành 3 tập train, test, validation theo tỷ lệ 70%:15%:15%.
- Sử dụng Pipeline: để kết hợp TfidfVectorizer và MultinomialNB thành một quy trình duy nhất để biến đổi và huấn luyện dữ liệu. Trong đó, TfidfVectorizer: Biến đổi văn bản thành các vector TF-IDF, giúp mô hình hiểu được tầm quan trọng của từ trong văn bản. MultinomialNB: Thuật toán Naive Bayes dành cho dữ liệu phân phối đa biến, phù hợp cho việc phân loại văn bản.
- Sử dụng LabelEncoder: Mã hóa nhãn đích từ chuỗi văn bản thành số nguyên, cần thiết cho việc huấn luyện mô hình.
- Cuối cùng sử dụng classification_report: Đánh giá hiệu suất của mô hình dựa trên các chỉ số như Precision, Recall và F1-score.

Kết quả của Naive Bayes:

Trên tập Val:

	precision	recall	f1-score	support
0	0.88	0.61	0.72	1184
1	0.70	0.91	0.79	1686
2	0.86	0.77	0.81	1252
accuracy			0.78	4122
macro avg	0.81	0.76	0.77	4122
weighted avg	0.80	0.78	0.78	4122

Trên tập Test:

	precision	recall	f1-score	support
0	0.89	0.57	0.69	1154
1	0.68	0.91	0.78	1685
2	0.88	0.77	0.82	1284
accuracy			0.77	4123
macro avg	0.81	0.75	0.76	4123
weighted avg	0.80	0.77	0.77	4123

4.2 Bert Model

Mô hình pre-trained thường được huấn luyện trên các tập dữ liệu lớn và phức tạp, giúp cho việc train mô hình:

- Tăng độ chính xác: Vì có thể học được nhiều đặc trưng từ dữ liệu đa dạng.
- Tiết kiệm được thời gian: Việc huấn luyện một mô hình từ đầu có thể mất rất nhiều thời gian và tài nguyên.



- Khả năng tái sử dụng: Các mô hình pre-trained có thể được sử dụng lại trong nhiều tác vụ khác nhau và không cần phải huấn luyện lại từ đầu.

Pre-trained distilbert-base-uncased là một mô hình xử lý ngôn ngữ tự nhiên pre-trained được phát triển bởi Hugging Face. DistilBERT là một phiên bản nhỏ hơn và nhanh hơn của BERT, một mô hình học sâu nổi tiếng trong lĩnh vực xử lý ngôn ngữ tự nhiên.

- Mô hình distilbert-base-uncased bao gồm 12 lớp mạng và khoảng 66 triệu tham số. Nó được huấn luyện trên tập dữ liệu lớn và đa dạng, bao gồm tập dữ liệu tiếng Anh của Wikipedia và các tập dữ liệu ngôn ngữ tự nhiên khác.

Cách thức hoạt động của DistilBERT:

DistilBERT sử dụng kiến trúc Transformer, cụ thể là phần encoder của Transformer, bao gồm các thành phần chính như sau:

Bước 1: Kiến trúc Transformer:

Attention Mechanism: Cơ chế chú ý cho phép mô hình tập trung vào các phần khác nhau của câu để nắm bắt ngữ cảnh. Đặc biệt là Self-Attention, nơi mỗi từ trong câu có thể chú ý đến tất cả các từ khác trong câu, giúp mô hình nắm bắt được các mối quan hệ ngữ nghĩa giữa các từ.

Feed-Forward Neural Networks: Mỗi lớp encoder bao gồm một mạng nơ-ron truyền thẳng được áp dụng cho từng vị trí của từ.

Layer Normalization và Residual Connections: Các lớp này giúp ổn định và tăng cường khả năng học của mô hình.

Bước 2: Knowledge Distillation

DistilBERT được huấn luyện sử dụng kỹ thuật knowledge distillation từ mô hình BERT lớn hơn, với các bước chính sau:

Teacher Model: BERT lớn đóng vai trò là mô hình "teacher" (giáo viên).

Student Model: DistilBERT là mô hình "student" (học sinh), được huấn luyện để học lại kiến thức từ mô hình teacher.

Loss Functions: Quá trình distillation sử dụng ba hàm mất mát chính:

Soft Target Loss: Giúp mô hình student học từ các xác suất dự đoán của mô hình teacher.

MLM Loss (Masked Language Modeling Loss): Giúp mô hình student học dự đoán từ bị che trong văn bản, tương tự như BERT.

Cosine Embedding Loss: Giúp giữ sự tương đồng giữa các biểu diễn ngữ nghĩa từ các lớp của mô hình student và teacher.

Cuối cùng:

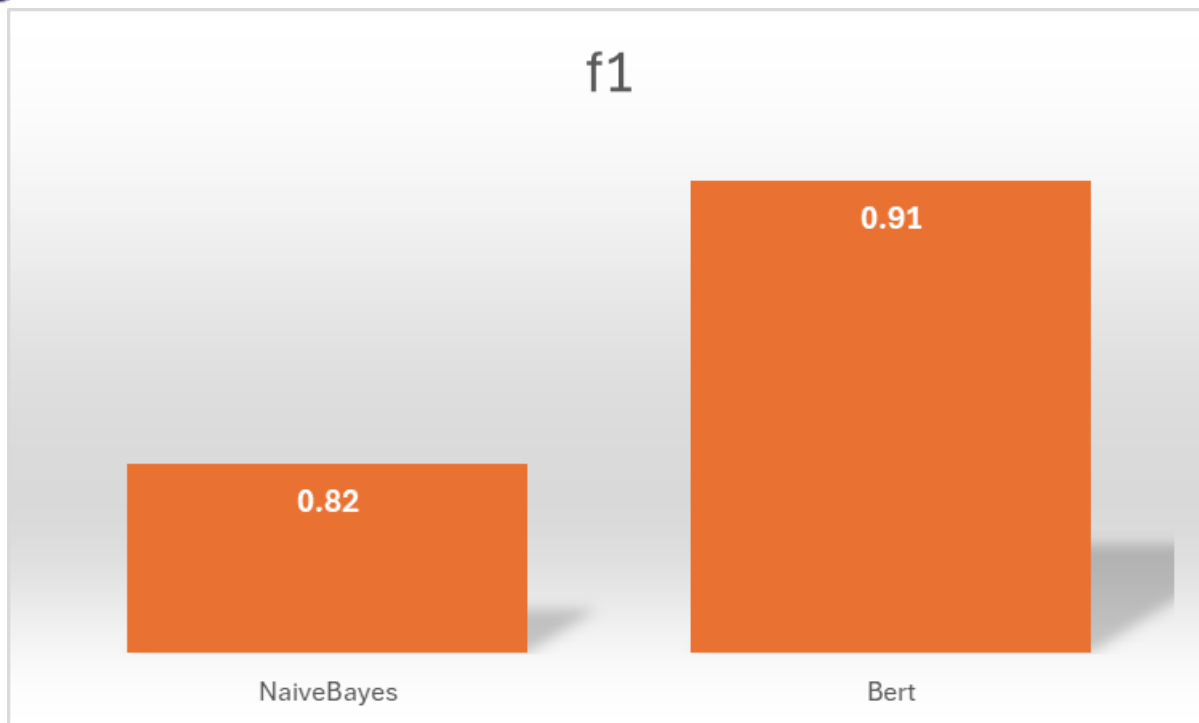
Quá trình Huấn luyện

- Pre-training: DistilBERT được huấn luyện trước trên một tập dữ liệu lớn với nhiệm vụ Masked Language Modeling (MLM), giống như BERT, để học các biểu diễn ngữ nghĩa của từ.
- Fine-tuning: Sau khi pre-training, DistilBERT có thể được fine-tuned trên các nhiệm vụ cụ thể, như phân loại cảm xúc, bằng cách sử dụng tập dữ liệu gắn nhãn.

Kết quả của quá trình huấn luyện mô hình dựa trên Bert.

	precision	recall	f1-score	support
0	0.91	0.85	0.88	1154
1	0.87	0.91	0.89	1685
2	0.91	0.91	0.91	1284
accuracy			0.89	4123
macro avg	0.90	0.89	0.89	4123
weighted avg	0.89	0.89	0.89	4123

Ta được kết quả của 2 mô hình là



Dựa vào biểu đồ ta thấy f1-score của mô hình Bert cao hơn NaiveBayes rất nhiều. Vì:

- Bert hiểu rõ hơn các từ trong ngữ cảnh cụ thể.
- Bert nắm bắt được mối quan hệ giữa các từ.
- Bert nắm bắt được các mối quan hệ phi tuyến.
- Bert có khả năng tổng quát hóa.

Do đó, nhóm sẽ chọn model Bert làm mô hình chính thức để đưa lên xây dựng giao diện.

III. Xây dựng giao diện

Giao diện được xây dựng từ framework Streamlit.

Streamlit là một open-source Python lib, nó giúp ta dễ dàng tạo một web app cho Machine Learning và Data Science.

Kết quả: <https://sentiment-classify.streamlit.app/>

Github: <https://github.com/hieuus/sentiment-classification>



Giao diện sản phẩm của nhóm:

The screenshot shows a web application for Sentiment Classification. On the left, a sidebar contains 'Team Information' with members: 20120084 - Nguyễn Văn Hiếu and 20120085 - Trần Xuân Hòa. The main area features three large circular icons representing 'Negative' (red with a sad face), 'Neutral' (yellow with a straight line), and 'Positive' (green with a smiley face). Below these is the title 'Sentiment Classification' and a section 'Input Tweet' with a text box and a 'Predict' button. A red crown icon is in the bottom right corner.

Nhập câu, đoạn văn cần dự đoán vào ô Input Tweet để dự đoán kết quả.

Các kết quả ví dụ:



Positive


×


Team Information


Members:

- 20120084 - Nguyễn Văn Hiếu
- 20120085 - Trần Xuân Hòa

Fork


Negative


Neutral


Positive

Sentiment Classification

Input Tweet

Please type the tweet in the blank field

I love you

Predict

Result: Positive

Score: 0.9950823783874512

Negative


×


Team Information


Members:

- 20120084 - Nguyễn Văn Hiếu
- 20120085 - Trần Xuân Hòa

Fork


Negative


Neutral


Positive

Sentiment Classification

Input Tweet

Please type the tweet in the blank field

I hate you

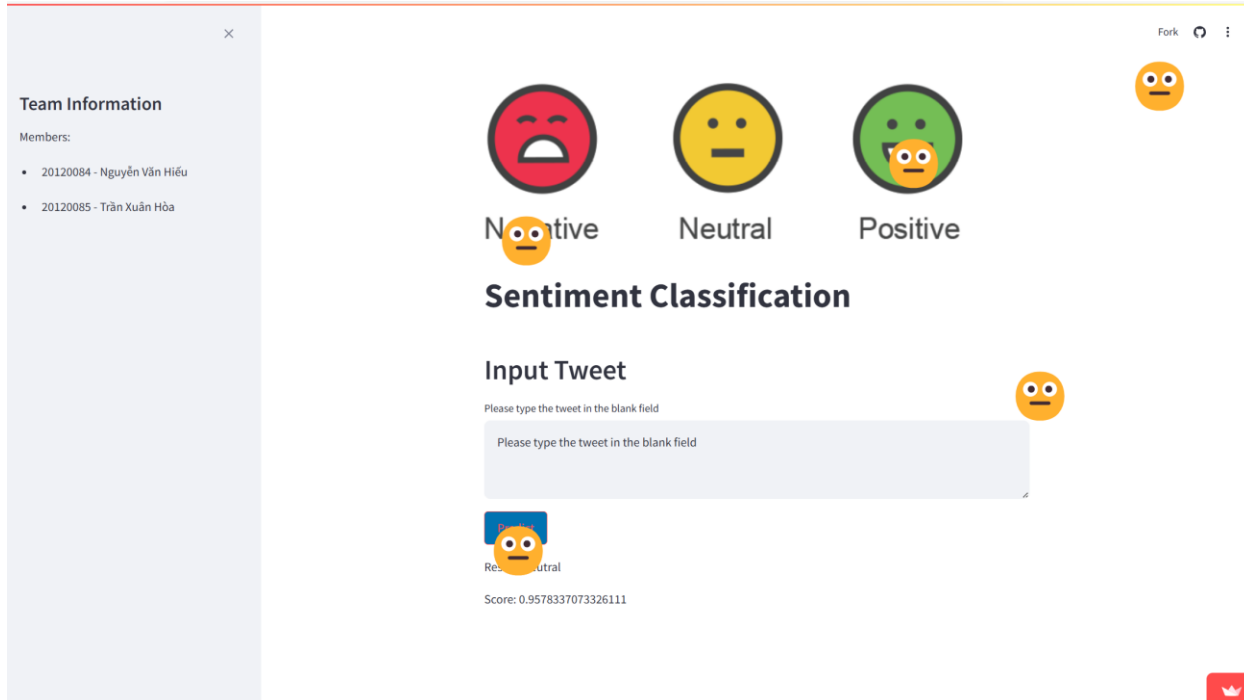
Predict

Result: Negative

Score: 0.9895367622375488



Neutral



Cách cài đặt Streamlit

```
pip install streamlit
```

Sau khi cài đặt xong có thể chạy ngay câu lệnh sau, nó đưa chúng ta tới tab với đường link <http://localhost:8501/>

```
streamlit hello
```

Ở đây chúng ta có thể xem một số demo của streamlit có sẵn.

Một số tính năng qua ví dụ

```
import streamlit as st
import pandas as pd
import numpy as np
```




```
# app title
st.title("My first deployed DL model")
st.header("Header đây chú đâu")
st.text("Đây là text")
st.markdown('Markdown đây **anh em ơi**')
st.text("Còn dưới đây là latex")
st.latex(r''' a + b = 3''')
# Viết một cái gì đó
st.write(12345)

# Hiển thị code luôn
code = '''def(hello):
    print("Hello world!")'''
st.code(code, language='python')

st.text("Hiển thị luôn cả chart")
hart_data = pd.DataFrame(np.random.randn(20, 3), columns=['a', 'b', 'c'])
st.line_chart(hart_data)
```

My first deployed DL model

Header đây chứ đâu

Đây là text

Markdown đây **anh em ơi**

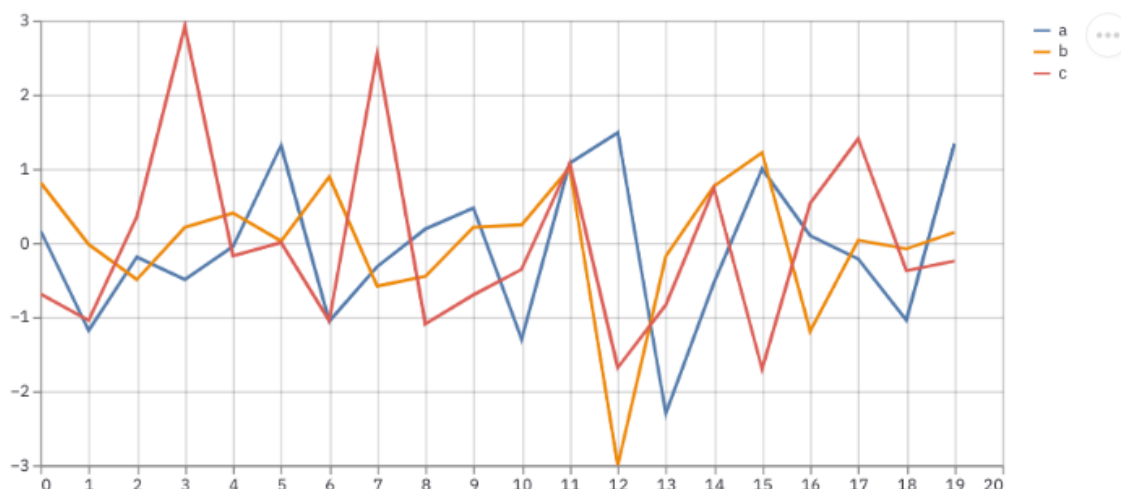
Còn dưới đây là latex

$$a + b = 3$$

12345

```
def(hello):
    print("Hello world!")
```

Hiển thị luôn cả chart





IV. Tham khảo

- [1] Twitter Tweets Sentiment Dataset , <https://www.kaggle.com/datasets/yasserh/twitter-tweets-sentiment-dataset>
- [2] BERT, https://huggingface.co/docs/transformers/model_doc/bert
- [3], Multinomial Naive Bayes, <https://www.geeksforgeeks.org/multinomial-naive-bayes/>
- [4] Streamlit docs, <https://docs.streamlit.io/>