BỘ GIÁO DỤC VÀ ĐÀO TẠO

ĐẠI HỌC KINH TẾ TP HỒ CHÍ MINH

TRƯỜNG CÔNG NGHỆ VÀ THIẾT KẾ

ĐỒ ÁN MÔN HỌC

ĐỀ TÀI:

PHÂN TÍCH HỘI CHỨNG TRẦM CẢM DỰA TRÊN TRẠI THÁI (STATUS) CHIA SẺ TRÊN TWITTER

Học phần: XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Nhóm Sinh Viên:

1. Đoàn Vũ Minh Thanh - 31201020910
2. Đặng Thị Cẩm Tú - 31201024511
3. Huỳnh Thị Cẩm Nhung - 31201024508

Chuyên Ngành: KHOA HỌC DỮ LIỆU

Khóa: K46

Giảng Viên: TS. Đặng Ngọc Hoàng Thành

TP. Hồ Chí Minh, Ngày 15 tháng 12 năm 2022

MỤC LỤC

MỤC LỤC HÌNH ẢNH

CHƯƠNG 1. TỔNG QUAN

1.1. Giới thiệu bài toán

Twitter là một nền tảng mạng xã hội lớn thường được dùng cho giao tiếp và chia sẽ thông tin, cảm xúc với mọi người nói chung và bạn bè nói riêng, một nền tảng khác tương tự như Facebook. Twitter được biết đến là nền tảng cho phép cho thông tin dễ dàng lan truyền và đọc dưới hình thức là các tin nhắn ngắn gọn và thường xuyên được cập nhật. Ở đó người dùng mạng xã hội này thường đăng và chia sẻ ý kiến của họ về nhiều chủ đề, thảo luận về các vấn đề hiện tại, phàn nàn và bày tỏ cảm xúc tích cực hay tiêu cực đối với các sản phẩm họ sử dụng hay đơn giản là tâm trạng của họ trong cuộc sống hàng ngày. Ngoài ra ở nền tảng này được biết là bị hạn chế 140 ký tự trên 1 dòng tweet, và cũng vì tính chất giới hạn ký tự này, mọi người sử dụng khá nhiều từ viết tắt, mắc lỗi chính tả, sử dụng biểu tượng cảm xúc và các ký tự đặc biệt khác thể hiện các ý nghĩa đặc biệt.

Trong thực tế hiện nay các công ty sản xuất các sản phẩm bắt đầu thăm dò các các nền tảng như vậy để rút ra được phân tích và kết luận về sản phẩm của họ và thị trường, ngoài ra rất nhiều trường hợp các công ty đã nghiên cứu phản ứng của người dùng và trả lời người dùng trên chính blog của họ. Ngày nay đã có rất nhiều nỗ lực nghiên cứu và báo cáo trong việc phân tích cảm xúc và ứng dụng quy trình xử lý ngôn ngữ tự nhiên (Natural Language Processing) được coi như là một nhiệm vụ ở rất nhiều cấp độ chi tiết…

Trong bài báo cáo đồ án này, nhóm chúng em sẽ xem xét và xây dựng các mô hình để phân loại “tweet” thành cảm xúc tích cực và tiêu cực - Twitter Sentiment Analysis (TSA) từ đó phát hiện được những đấu hiệu của chứng trầm cảm dựa trên nhưng cảm xúc tiêu cực.

1.2. Lý Do Chọn Lựa Đề Tài

Một thách thức được đưa ra ở đây là, làm thế nào để xây dựng nên một công nghệ hay một mô hình có thể phát hiện, phân tích và tóm tắt chỉ số cảm xúc (sentiment) cho tổng thể.

Chúng em thử nghiệm với 2 loại mô hình là: Decision Tree và Support Vector Machine. Nhóm sử dụng bộ dữ liệu Twitter được chú thích một cách thủ công cho các thử nghiệm mô hình của nhóm, lợi thế của dữ liệu này so với các bộ dữ liệu khác là các “tweet” được thu thập theo kiểu thông tin được truyền trực tuyến và vì vậy nó sẽ đại diện cho một mẫu tweet thực sự về mặt ngôn ngữ và nội dung. Và ngoài ra là các ký tự nhiễu như các link URL, ký tự đặc biệt, các emoji thể hiện cảm xúc của họ,... cần phải được xử lý. Để thực hiện đề tài này, đầu tiên nhóm sẽ tiến hành các kỹ thuật tiền xử lí để làm sạch dữ liệu và sử dụng các tài nguyên bổ sung, sau khi tiền xử lí nhóm sẽ thực hiện việc phân tích và trực quan dữ liệu dựa trên bộ dữ liệu đã được tiền xử lí. Và cuối cùng là tiến hành lấy mẫu, xây dựng mô hình dự báo để phân tích và đánh giá kết quả thu được để đưa ra các hướng nghiên cứu và phát triển tiếp theo.

CHƯƠNG 2. CÁC MÔ HÌNH PHÂN LỚP DỮ LIỆU

2.1. Các phương pháp tiền xử lý dữ liệu

- ***Lowercase text***: Trong NLP, các mô hình trong quá trình xử lý sẽ cho rằng các từ như 'Goat' và 'goat' là khác nhau, kể cả trong trường hợp chúng là 1. Do đó, để khắc phục điều này nhóm thực hiện viết thường toàn bộ các từ.
- ***Replace repeating letters:*** Khi đánh giá các dòng text có thể thấy rằng đôi khi người dùng lặp lại các chữ cái trong một từ nhằm để nhấn mạnh cảm xúc của họ (ví dụ: 'gooooo' đại diện 'go', 'thissss' đại diện cho 'this',..). Do đó việc tìm kiếm nhiều hơn 2 chữ cái lặp lại trong 1 từ sẽ được thay thế 2 trong số những chữ đó.
- ***Remove stop words***: Stop words thường là những từ cực kỳ phổ biến, và nó ít có giá trị trong phân phân loại cảm xúc.
- ***Tokenization***: quá trình tách một cụm từ, câu, đoạn văn, một hoặc nhiều tài liệu văn bản thành các đơn vị nhỏ hơn. Mỗi đơn vị nhỏ hơn này được gọi là Tokens. Có thể coi tokens là các khối xây dựng của NLP và tất cả các mô hình NLP đều xử lý văn bản thô ở cấp độ các Tokens.
- ***Lemmatization***: xác định lemma của 1 từ dựa trên ý nghĩa tạm thời của nó với việc sử dụng từ cùng và phân tích hình thái từ. Điều này dẫn đến việc chỉ loại bỏ các kết thúc biến tố và trả về dạng cơ sở hoặc dạng từ điển của một từ. Và để Lemmatization thì phải chuyển các câu ở mỗi dòng thành các list các token
- ***Xử lý các emoji***: trên các trang mạng xã hội, emoji là một công cụ quen thuộc, phổ biến trong việc thể hiện cảm xúc, tâm trạng của người dùng, vậy nên để phân tích tâm lý của người dùng thông qua dòng trạng thái, nhóm sẽ xử lý các emoji qua các phương pháp như thay thế kí tự thành chữ, đếm số lượng emoji xuất hiện trong dòng trạng thái, liệt kê các loại emoji xuất hiện trong từng dòng trạng thái

2.2. Các Mô Hình Phân Lớp Dữ Liệu

2.2.1. Mô Hình Support Vector Machine

SVM là một thuật toán giám sát, nó có thể sử dụng cho cả việc phân lớp, phát hiện outliers và phân tích đệ quy. Và nó được sử dụng chủ yếu cho việc phân lớp. Trong thuật toán này, chúng ta vẽ đồi thị dữ liệu là các điểm trong n chiều (n-grams feature) (ở đây n là số lượng các tính năng bạn có) với giá trị của mỗi tính năng sẽ là một phần liên kết. Sau đó thực hiện tìm "đường bay" (hyper-plane) để phân chia các lớp. Hyper-plane nó chỉ hiểu đơn giản là 1 đường thẳng có thể phân chia các lớp ra thành hai phần riêng biệt.

Ngoài ra SVM có những điểm nổi bật như:

- Xử lý trên không gian số chiều cao: SVM là một công cụ tính toán hiệu quả trong không gian chiều cao, trong đó đặc biệt áp dụng cho các bài toán phân loại văn bản và phân tích quan điểm nơi chiều có thể cực kỳ lớn.

- Tiết kiệm bộ nhớ: Do chỉ có một tập hợp con của các điểm được sử dụng trong quá trình huấn luyện và ra quyết định thực tế cho các điểm dữ liệu mới nên chỉ có những điểm cần thiết mới được lưu trữ trong bộ nhớ khi ra quyết định.
- Tính linh hoạt - phân lớp thường là phi tuyến tính. Khả năng áp dụng Kernel mới cho phép linh động giữa các phương pháp tuyến tính và phi tuyến tính từ đó khiến cho hiệu suất phân loại lớn hơn.

2.2.2. Mô Hình Decision Tree

Thuật toán cây quyết định là một thuật toán khai thác dữ liệu phân vùng đệ quy cho một tập dữ liệu gồm các bản ghi bằng cách sử dụng phương pháp tiếp cận ưu tiên theo chiều sâu hoặc phương pháp tiếp cận theo chiều rộng cho đến khi tất cả các mục dữ liệu thuộc về một lớp cụ thể. Cấu trúc cây quyết định bao gồm các nút gốc, nút trong và nút lá. Cấu trúc cây được sử dụng để phân loại các bản ghi dữ liệu chưa biết. Tại mỗi nút bên trong của cây quyết định phân chia tốt nhất được đưa ra bằng cách sử dụng các phương pháp đánh giá khác nhau. Các lá cây được tạo thành từ các nhãn mà dữ liệu đã được phân lớp.

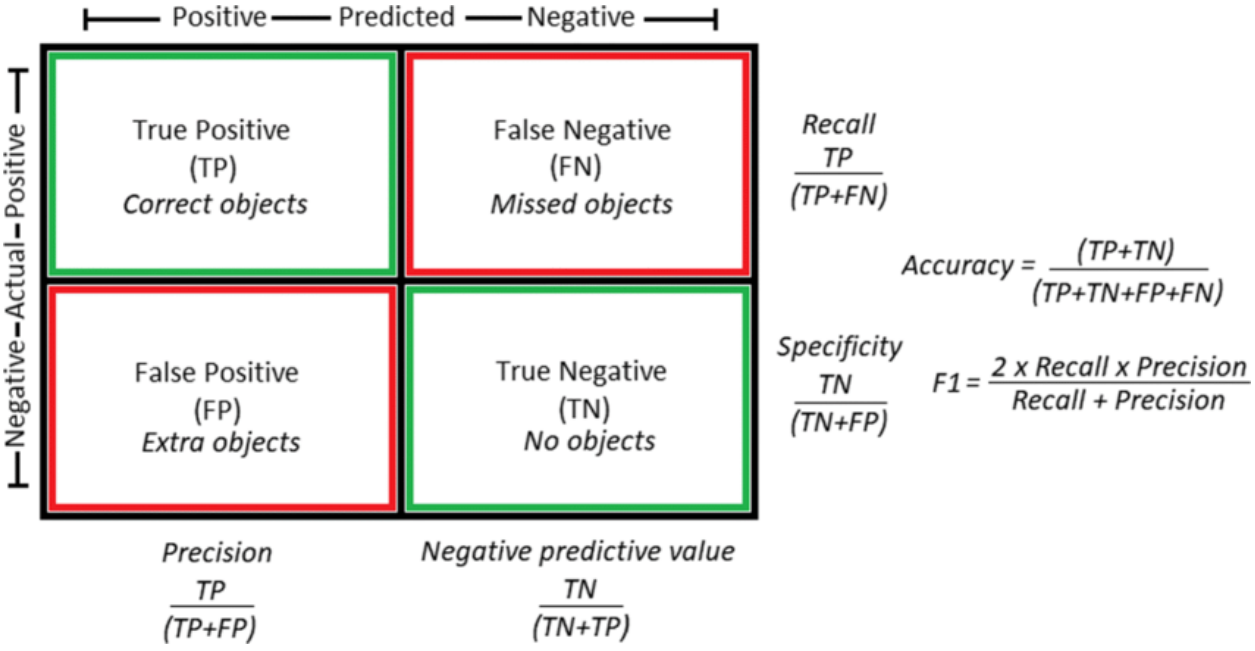
Kỹ thuật phân loại cây quyết định được thực hiện theo hai giai đoạn: xây dựng cây và cắt tỉa cây. Việc xây dựng cây được thực hiện theo cách từ trên xuống. Trong giai đoạn này, cây được phân vùng đệ quy cho đến khi tất cả các mục dữ liệu thuộc về cùng một nhãn lớp. Nó đòi hỏi rất nhiều nhiệm vụ và tính toán chuyên sâu vì tập dữ liệu huấn luyện được duyệt đi duyệt lại nhiều lần. Cắt tỉa cây được thực hiện theo cách từ dưới lên. Nó được sử dụng để cải thiện độ chính xác của dự đoán và phân loại của thuật toán bằng cách giảm thiểu quá khớp(overfitting) (bị nhiễu hoặc nhiều chi tiết trong tập dữ liệu huấn luyện). Over-fitting trong thuật toán cây quyết định dẫn đến lỗi phân loại sai. Việc cắt tỉa cây ít nhiệm vụ hơn so với giai đoạn “phát triển” của cây vì tập dữ liệu huấn luyện chỉ được xử lý qua một lần.

Trong hệ thống được đề xuất, phân loại cây quyết định cung cấp tùy chọn tốt hơn cho người dùng cuối để phân loại các tweet tích cực và tiêu cực. Nó được thực hiện bằng cách so sánh các mục phổ biến tối đa được tạo bởi các quy tắc trong dữ liệu huấn luyện đã được so sánh với các mục phổ biến tối đa của dữ liệu thử nghiệm và do đó việc phân loại có thể được thực hiện dễ dàng.

2.2.3 Chỉ số đánh giá

Để đánh giá về hiệu quả của mô hình được đề xuất là SVM và Decision Tree,mô hình nào phù hợp nhất để phân loại cho tập dữ liệu phân loại Twitter này thì nhóm đánh giá mô hình bằng các keys metric precision, recall, classification accuracy, và f-measure.

Ngoài ra, ma trận nhầm lẫn (confusion matrix) dùng để đánh giá độ phân loại cho dữ liệu nhị phân được trình bày theo bảng dưới, và các chỉ số trong ma trận nhầm lẫn này được dùng để tính cho các chỉ số đánh giá mô hình trên



Hình 1. Confusion Matrix

2.3. Các thư viện quan trọng

2.2.1. Thư viện NLTK

Thư viện NLTK - Natural Language Toolkit được coi là một công cụ tuyệt vời để giảng dạy và làm việc trong xử lí ngôn ngữ tự nhiên bằng Python. Nó là một trong những thư viện open-soure phổ biến nhất trong NLP và có một cộng đồng rất lớn những người sử dụng. NLTK là một bộ chứa các thư viện và chương trình để xử lý ngôn ngữ thống kê. Đây là một trong những thư viện NLP mạnh mẽ nhất, chứa các packages giúp máy tính hiểu ngôn ngữ của con người và trả lời ngôn ngữ đó bằng một phản hồi thích hợp.

Ở bài báo cáo này, nhóm đã sử dụng một số phương thức trong NLTK đó là stopwords (loại bỏ các từ có tần suất xuất hiện nhiều như “the”, “to”..); SnowballStemmer (xử lí các từ không phải tiếng anh); WordNetLemmatizer (sửa đổi các từ đưa về dạng gốc của nó, chẳng hạn như “go”,”goes”,”went” sẽ trả về dạng gốc như nhau); tokenize (tách từ).

2.2.2. Thư viện sklearn

Scikit-learning là một thư viện máy học miễn phí dành cho Python. Nó có các thuật toán khác nhau như SVM, random forests và k-neighbors, đồng thời nó cũng hỗ trợ các thư viện Python như NumPy và SciPy. Scikit-learning có lẽ là thư viện hữu ích nhất cho Machine learning bằng Python. Thư viện sklearn chứa rất nhiều công cụ hiệu quả cho học máy và thiết lập các mô hình thống kê như phân nhóm, hồi quy, phân cụm và giảm chiều dữ liệu.

Nhóm đã sử dụng hàm .LabelEncoder() trong Sklearn để xử lí dữ liệu và hàm train_test_split() để chia tập dữ liệu thành các mẫu thử nghiệm và tiến hành train (đào tạo) chúng. Cuối cùng, nhóm sẽ sử dụng thuật toán như LinearSVC và DecisionTreeClassifie để đưa ra dự đoán và so sánh hiệu suất của chúng bằng các phương pháp như precision_score() do thư viện scikit-learning cung cấp. Chúng tôi cũng sẽ trực quan hóa điểm hiệu suất của các mô hình khác nhau bằng cách sử dụng trực quan hóa scikit-learning.

2.2.3. Thư viện Mathlob và Seaborn

Matplotlib là một thư viện gần như toàn diện và rất phổ biến để tạo nên các biểu diễn trực quan dưới dạng hình ảnh tĩnh, hoạt ảnh và tạo nên sự tương tác giữa các biến với nhau. Nói một cách dễ hiểu hơn, đây là một thư viện dùng để vẽ biểu đồ, đồ thị trong Python..Khi biểu diễn trực quan với matplotlib, người phân tích có thể xoay, phóng to và cập nhật các hình ảnh. Chúng ta có thể dùng thư viện để vẽ hầu như mọi dạng biểu đồ từ hình tròn, biểu đồ cột, điểm phân tán, biểu đồ phân phối cho đến biểu đồ quang phổ công suất… Module được sử dụng nhiều nhất hiện nay trong thư viện chính là Pyplot, ở báo cáo này nhóm cũng sẽ sử dụng module này cho phần Trực quan phân tích. Pyplot cung cấp các hàm đơn giản để thêm vào hình ảnh trực quan các thành phần như text, tiêu đề, hình ảnh, màu sắc, chỉ số phần trăm, kích thước, định dạng…

Một số phương thức xuất hiện khá nhiều trong thư viện này là: plot(x-axis values, y-axis values), show() - hiển thị hình ảnh, figure() - tùy chỉnh kích thước, legend() - tạo chú thích, title() - đặt tiêu đề cho biểu đồ, axes() - thêm trục vào hình…

Matplotlib thực sự là một thư viện rất hữu ích mà hầu như người phân tích nào cũng sẽ cần đến cho các trực quan dữ liệu của mình.

Đây là một thư viện được đánh giá rất cao, nó thực hiện việc trực quan hóa dễ dàng chỉ với vài bước đơn giản và có thể tạo nên một hình ảnh trực quan rất đẹp mắt với nhiều màu sắc và bố cục rất gọn gàng, đem lại cho người phân tích nhiều thông tin hữu ích từ biểu đồ. Thư viện seaborn này sẽ dựa trên thư viện Matplotlib, hay nói cách khác nó là phần mở rộng hơn của Python. Chính vì vậy, trong trực quan hóa, sự kết hợp giữa hai thư viện này sẽ tạo nên một biểu diễn hình ảnh bắt mắt và cung cấp cho người phân tích những cái nhìn mới hơn với thông tin trong bộ dữ liệu.

Ưu điểm của thư viện này là có thể tạo nhiều hình trong cùng một khung và hoạt động với toàn bộ dữ liệu. Khác với Matplotlib chỉ tạo một hình duy nhất (muốn tạo thêm phải hủy đi cái hình hiện tại) và chỉ hoạt động với Dataframes và Arrays.

CHƯƠNG 3. CÁC KẾT QUẢ THỰC NGHIỆM

3.1. Nhập Thư Viện Cần Dùng Và Tiền Xử Lý Bộ Dữ Liệu

Trước khi thực hiện việc khai phá và phân tích dữ liệu, nhóm thực hiện việc tiền xử lý dữ liệu để dễ dàng hơn trong việc khám phá thông tin từ các status và ứng dụng vào việc áp dụng các thuật toán máy học cho bài toán phân lớp dữ liệu. Nếu bỏ qua bước này thì nhóm phải làm việc với dữ liệu bị nhiều và không đồng nhất, ảnh hưởng trong việc đưa ra các kết luận và insight rút ra từ đó bị sai.

Đầu tiên, nhóm khai báo các thư viện và các hàm nhóm dự định sẽ sử dụng trong bài đồ án này, bao gồm thư viện sklearn, thư viện nltk, thư viện mathlob, thư viện pandas,…

```
import tensorflow as tf
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np


import nltk
from nltk.corpus import stopwords
from nltk.stem import SnowballStemmer
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.feature_extraction.text import CountVectorizer,TfidfTransformer
from nltk.stem.snowball import EnglishStemmer
import re

print("Tensorflow Version",tf.__version__)

import warnings
warnings.filterwarnings('ignore')
```

Hình 2. Import các thư viện và các hàm cần thiết

Bộ dữ liệu nhóm sẽ sử dụng là “training.1600000.processed.noemoticon.csv”. Bộ dữ liệu bao gồm 1,600,000 bài tweet trên trang Twitter, được trích xuất bởi twitter api. Các bài tweet đã được gán nhãn (0 = negative, 2 = neutral, 4 = positive) và chúng có thể được dùng để điều tra về cảm xúc. Bộ dữ liệu gồm 6 biến:

- target: tính chất của tweet (0 = negative, 2 = neutral, 4 = positive)
- ids: mã ID của tweet (ví dụ: 2087)
- date: thời gian tweet được đăng tải (ví dụ: Sat May 16 23:58:44 UTC 2009)
- flag: query (lyx); nếu không có query, giá trị của ô này sẽ là NO_QUERY.
- user: chủ nhân dòng tweet (ví dụ: robotickilldozr)
- text: nội dung của tweet (ví dụ: Lyx is cool)

```
df = pd.read_csv('training.1600000.processed.noemoticon.csv',
                 encoding = 'ISO-8859-1',header=None, index_col=None)
df.head()
```

	0	1	2	3	4	5
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....

Hình 3. Kết nối với bộ dữ liệu

Nhóm nhận thấy bộ dữ liệu này không có tiêu đề cho từng features, nên nhóm thực hiện đặt tên cho các features để dễ dàng hơn trong việc phân biệt giữa giữa các features.

```
df.rename(columns={0:"target",1: "ids",2: "date",3: "flag",4: "user", 5:"text"},
          inplace=True, errors='raise')
df.columns
```

```
Index(['target', 'ids', 'date', 'flag', 'user', 'text'], dtype='object')
```

Hình 4. Đặt tên các cột dữ liệu

Nhóm kiểm tra có giá trị rỗng nào trong bộ dữ liệu không bằng hàm .isnull(), và kết quả cho thấy không có dữ liệu bị thiếu trong bộ dữ liệu này.

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

```
target    0
ids       0
date      0
flag      0
user      0
text      0
dtype: int64
```

Hình 5. Kiểm tra missing values

Trọng tâm của bài đồ án này là phân tích sự trầm cảm qua nội dung các dòng trạng thái nên nhóm sẽ giữ lại hai cột dữ liệu quan trọng phục vụ cho bài, là cột *target* và *text*.

```
# Removing the unnecessary columns.
df = df[['target','text']]
df.head()
```

	target	text
0	0	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	is upset that he can't update his Facebook by ...
2	0	@Kenichan I dived many times for the ball. Man...
3	0	my whole body feels itchy and like its on fire
4	0	@nationwideclass no, it's not behaving at all....

Hình 6. Giữ 2 cột target và text

Nhóm làm sạch văn bản (text) qua các bước:

- Lowercase

```
# LowerCase text
df['cleaned']=df['text'].apply(lambda x: x.lower())
```

Hình 7. Lowercase

- Bỏ đi các chữ số
- Bỏ tên user được gắn thẻ và URLs
- Thay thế các chữ cái bị lặp lại

```
# Defining regex patterns.
urlPattern        = r"((http://)[^ ]*|(https://)[^ ]*|( www\.)[^ ]*)"
userPattern       = '@[^\s]+'
alphaPattern      = "[^a-zA-Z0-9]"
sequencePattern   = r"(.)\1\1+"
seqReplacePattern = r"\1\1"
```

```
# Remove url Pattern
df['cleaned']=df['cleaned'].apply(lambda x: re.sub('((http://)[^ ]*|(https://)[^ ]*|( www\.)[^ ]*)',' '
```

```
# Remove user Pattern
df['cleaned']=df['cleaned'].apply(lambda x: re.sub(userPattern,' ', x))
```

```
# Remove alpha Pattern
df['cleaned']=df['cleaned'].apply(lambda x: re.sub(alphaPattern,' ', x))
```

```
# Replace two or more consecutive repetitions of a letter with two of the same
df['cleaned']=df['cleaned'].apply(lambda x: re.sub(sequencePattern, seqReplacePattern, x))
```

Hình 8. Loại bỏ tagged user, URLs, chữ số, các chữ cái lặp lại

- Xóa các dấu cách thừa

```
# Removing extra spaces
df['cleaned']=df['cleaned'].apply(lambda x: re.sub(' +',' ',x))
```

Hình 9. Loại bỏ dấu cách thừa

- Group emoticons. Mỗi emojis đều có 1 ý nghĩa riêng của nó (ví dụ: :): (Slightly) smile, :(:sad) và chúng rất có ý nghĩa trong việc phân biệt cảm xúc, tích cực hay tiêu cực, qua các dòng status. Ở đây nhóm thực hiện duyệt các các từ của mỗi dòng status và thay thế chúng bằng những từ mang tính phân loại hơn, dễ dàng phân thích hơn

```
# Defining dictionary containing all emojis with their meanings.
emojis = {':)': 'smile', ':-)': 'smile', ';d': 'wink', ':-E': 'vampire', ':(': 'sad',
          ':-(': 'sad', ':-<': 'sad', ':P': 'raspberry', ':O': 'surprised',
          ':-@': 'shocked', ':@': 'shocked',':-$': 'confused', ':\\': 'annoyed',
          ':#': 'mute', ':X': 'mute', ':^)': 'smile', ':-&': 'confused', '$_$': 'greedy',
          '@@': 'eyeroll', ':-!': 'confused', ':-D': 'smile', ':-0': 'yell', 'O.o': 'confused',
          '<(-_-)>': 'robot', 'd[-_-]b': 'dj', ":'-)": 'sadsmile', ';)': 'wink',
          ';-)': 'wink', 'O:-)': 'angel','O*-)': 'angel','(:-D': 'gossip', '=^.^=': 'cat'}
```

Hình 10. Định nghĩa các emoji

```
# Replace all emojis.
for emoji in emojis.keys():
    df['cleaned']=df['cleaned'].apply(lambda x: x.replace(emoji, "EMOJI" + emojis[emoji]))
```

Hình 11. Thay thế kí tự emoji thành tên của nó

- Loại bỏ stopwords

```
stop_words = set(stopwords.words("english"))
def remove_stopwords(text):
    words = [w for w in text if w not in stop_words]
    return words
df['stopwordremove_tokens'] = df['tokens'].apply(lambda x : remove_stopwords(x))
df.head()
```

Hình 12. Loại bỏ stopwords

- Lemmatization + Tokenization

```
#convert cleaned text into tokens
tokenizer=nltk.tokenize.RegexpTokenizer(r'\w+')
df['tokens'] = df['cleaned'].apply(lambda x:tokenizer.tokenize(x))
df.head()
```

Hình 13. tokenization

```
# lemmatization
lem = WordNetLemmatizer()
def lem_word(x):
    return [lem.lemmatize(w) for w in x]

df['lemmatized_text'] = df['stopwordremove_tokens'].apply(lem_word)
```

```
def combine_text(list_of_text):

    combined_text = ' '.join(list_of_text)
    return combined_text

df['final_text'] = df['lemmatized_text'].apply(lambda x : combine_text(x))
df.head()
```

Hình 14. lemmanization

- Đếm số lượng emoji xuất hiện trong bài viết: nhóm xây dựng một hàm đếm, với từng danh sách từ đã được lemmatize, hàm sẽ đếm có bao nhiêu phần tử có giá trị là ‘EMOJI’ và hàm sẽ trả lại kết quả là số lượng từ ‘EMOJI’ trong bài viết đó

```
def emoji_count(w):

    return w.count('EMOJI')

df['emoji_count'] = df['lemmatized_text'].apply(emoji_count)
```

Hình 15. Đếm số lượng emoji trong từng bài tweet

- Thành lập danh sách tên các emoji xuất hiện trong bài viết: nhóm xây dựng một hàm tạo danh sách: với từng danh sách từ đã được lemmatize, nếu phần tử đứng trước phần tử đang xét hiện tại có giá trị là ‘EMOJI’ thì phần tử đang xét sẽ được thêm vào list - danh sách emoji của bài đăng đó; kết quả hàm trả về là một list chứa tên/ý nghĩa các emoji.

```
def emoji_list(wlist):
  return [word for word in wlist if wlist[wlist.index(word)-1] == 'EMOJI']

df['emoji_list'] = df['lemmatized_text'].apply(emoji_list)
```

Hình 16. Danh sách các emoji trong tweet

- Nối các từ đã được xử lý từ dạng list sang dạng chuỗi

```
def combine_text(list_of_text):

    combined_text = ' '.join(list_of_text)
    return combined_text

df['final_text'] = df['lemmatized_text'].apply(lambda x : combine_text(x))
df.head()
```

Hình 17. Chuyển hóa thành dạng chuỗi

	target	text	cleaned	tokens	stopwordremove_tokens	lemmatized_text	emoji_count	emoji_list	final_text
0	0	@switchfoot http://twitpic.com/2y1zl - Awww, t...	aww that s a bummer you shoulda got david car...	[aww, that, s, a, bummer, you, shoulda, got, d...	[aww, bummer, shoulda, got, david, carr, third...	[aww, bummer, shoulda, got, david, carr, third...	1	[llo]	aww bummer shoulda got david carr third day EM...
1	0	is upset that he can't update his Facebook by ...	is upset that he can t update his facebook by ...	[is, upset, that, he, can, t, update, his, fac...	[upset, update, facebook, texting, might, cry,...	[upset, update, facebook, texting, might, cry,...	0	[]	upset update facebook texting might cry result...
2	0	@Kenichan I dived many times for the ball. Man...	i dived many times for the ball managed to sa...	[i, dived, many, times, for, the, ball, manage...	[dived, many, times, ball, managed, save, rest...	[dived, many, time, ball, managed, save, rest,...	0	[]	dived many time ball managed save rest go bound
3	0	my whole body feels itchy and like its on fire	my whole body feels itchy and like its on fire	[my, whole, body, feels, itchy, and, like, its...	[whole, body, feels, itchy, like, fire]	[whole, body, feel, itchy, like, fire]	0	[]	whole body feel itchy like fire
4	0	@nationwideclass no, it's not behaving at all....	no it s not behaving at all i m mad why am i ...	[no, it, s, not, behaving, at, all, i, m, mad,...	[behaving, mad, see]	[behaving, mad, see]	0	[]	behaving mad see

Hình 18. Bộ dữ liệu sau khi tiền xử lý

3.2. Khai phá dữ liệu

Việc phân tích và trực quan hóa dữ liệu sẽ đem lại cho chúng ta cái nhìn tổng quan hơn về bộ dữ liệu, cũng như xem xét về các bài đăng tiêu cực và tích cực, xem liệu chúng chênh lệch như thế nào, các từ ngữ phổ biến trong các bài đăng được phân loại tích cực - tiêu cực và biểu cảm được sử dụng thường xuyên như thế nào trong các bài đăng. Điều này sẽ giúp cho việc xây dựng mô hình phân loại và khiến cho công ty, nhà sản xuất và nghiên cứu có thể có hướng phát triển trong thời gian tới.

Đầu tiên, nhóm tiến hành xem xét tỉ lệ giữa bài đăng tiêu cực và bài đăng tích cực, xem liệu chúng chênh lệch như thế nào thông qua việc vẽ biểu đồ trực quan Pie Chart (The Ratio of Negative and Positive).

▾ The Ratio of Negative and Positive

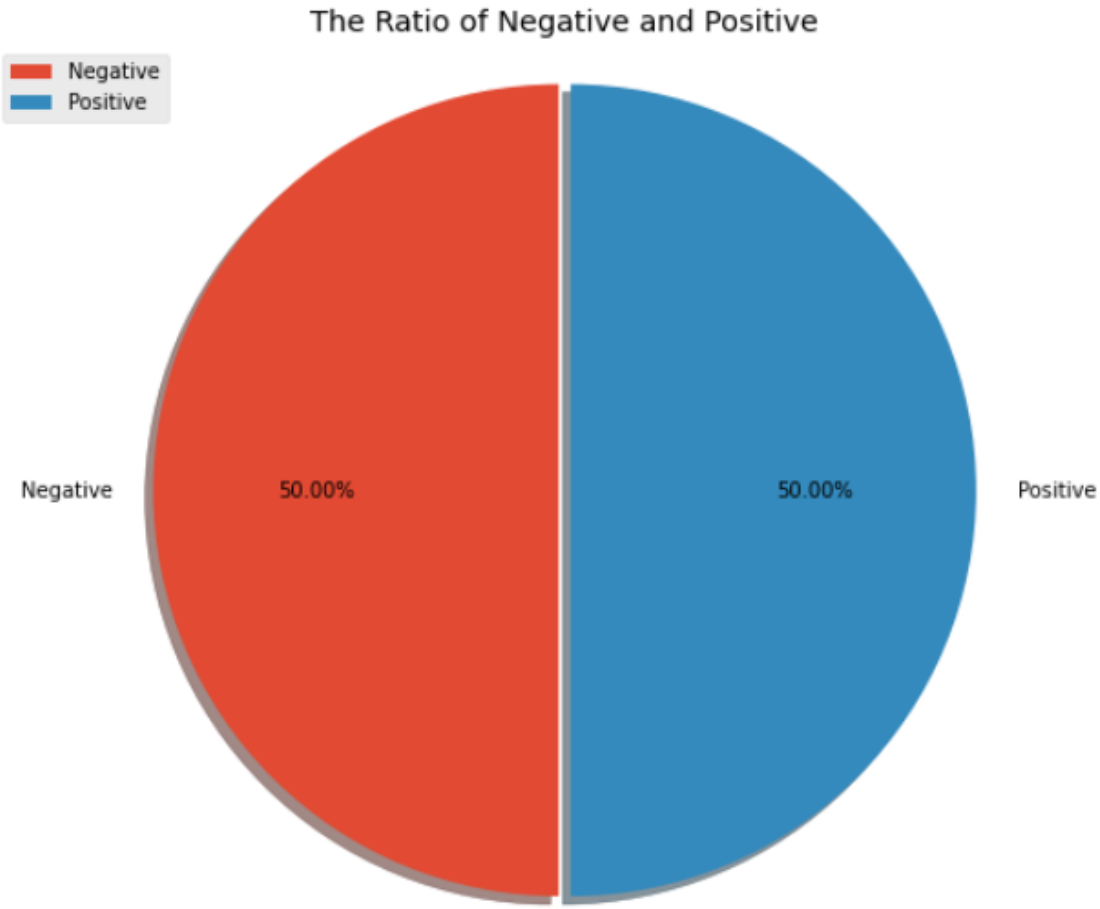
```
[ ]  data_pie = df[['target']].value_counts()
     data_pie
     labels = ['Negative','Positive']
     explode = (0.03,0)
     plt.style.use('ggplot')
     plt.subplots(figsize=(10,8))
     plt.title('The Ratio of Negative and Positive')
     plt.pie(x=data_pie, explode=explode, labels=labels, autopct='%.2f%%', shadow=True, startangle=90)
     plt.axis('equal')
     plt.legend(loc='upper left')
     plt.show()
```

Hình 19. Code biểu đồ trực quan thể hiện tỉ lệ số lượng bài đăng tích cực và tiêu cực

Với dòng lệnh trên, đầu tiên, nhóm tạo một bộ dữ liệu mới "data_pie" để liệt kê số lượng bài đăng tiêu cực và tích cực. Sau đó nhóm thực hiện việc vẽ biểu đồ với thư viện matplotlib, sử dụng hàm plt.pie() với:

- x là data_pie
- labels ở đây nhóm sẽ chọn là Negative và Positive (labels nghĩa là chọn nhãn được đưa vào biểu đồ hình tròn)
- Autopct = %.2f%% thể hiện phần trăm được chia được đưa vào biểu đồ
- Shadow = True, hình tròn sẽ được làm bóng lên, tạo sự ưa nhìn cho biểu đồ
- Explode tao độ phân cách cho các labels được chia trong biểu đồ

Sau đó nhóm thực hiện lệnh plt.show() và chạy sẽ thu được biểu đồ hình tròn như sau:



Hình 20. Biểu đồ thể hiện tỷ lệ giữa bài đăng tiêu cực và tích cực

Qua biểu đồ Pie Chart mà chúng ta thu được như trên có thể thấy rằng sự phân bổ của bài đăng tiêu cực và tích cực là bằng nhau, chúng xuất hiện đồng đều, có tỷ lệ như nhau và không có sự chênh lệch nào về số lượng giữa 2 loại bài đăng.

Sau khi có được biểu đồ thể hiện tỉ lệ giữa các bài đăng tiêu cực, nhóm tiếp tục đếm số lượng bài post tiêu cực (Negative) và tích cực (Positive), sau đó vẽ biểu đồ phân phối (Distribution Plot) để rút ra được độ dài (số từ) trung bình trong các bài đăng theo phân loại tiêu cực và tích cực.

Trước tiên nhóm sẽ định nghĩa một hàm lấy tên là wl , hàm này sẽ thực hiện nhiệm vụ đếm số lượng từ trong từng dòng của các bài post theo phân loại.

```
[ ]  def wl(x):
        return len(x.split(" "));
```

Hình 21. Định nghĩa một hàm đếm số lượng từ trong các bài post theo phân loại

- Đối với bài tweet tiêu cực:

Tạo một dataframe mới với ‘target’ = 0 để hiển thị những bài post tiêu cực

```
[ ]  df_negative = pd.DataFrame()
     df_negative = df1[df1.target == 0]
     df_negative
```

	target	final_text
0	0	aww bummer shoulda got david carr third day EM...
1	0	upset update facebook texting might cry result...
2	0	dived many time ball managed save rest go bound
3	0	whole body feel itchy like fire
4	0	behaving mad see
...	...	...
799995	0	sick spending day laying bed listening
799996	0	gmail
799997	0	rest peace farrah sad
799998	0	sound like rival flagging ad much though
799999	0	resit exam summer wish worked harder first yea...

800000 rows × 2 columns

Hình 22. Tạo dataframe mới với target = 0

Tiếp tục thêm vào dataframe vừa tạo một cột mới bằng hàm apply() và truyền vào hàm apply đó hàm wl vừa được định nghĩa ở trên. Cột mới này sẽ được lấy tên là 'word_count', cột này sẽ hiển thị số lượng từ theo từng bài post tiêu cực.

```
[ ]  df_negative['word_count']=df_negative['final_text'].apply(wl)
     df_negative
```

	target	final_text	word_count
0	0	aww bummer shoulda got david carr third day EM...	10
1	0	upset update facebook texting might cry result...	11
2	0	dived many time ball managed save rest go bound	9
3	0	whole body feel itchy like fire	6
4	0	behaving mad see	3
...	...	...	...
799995	0	sick spending day laying bed listening	6
799996	0	gmail	1
799997	0	rest peace farrah sad	4
799998	0	sound like rival flagging ad much though	7
799999	0	resit exam summer wish worked harder first yea...	9

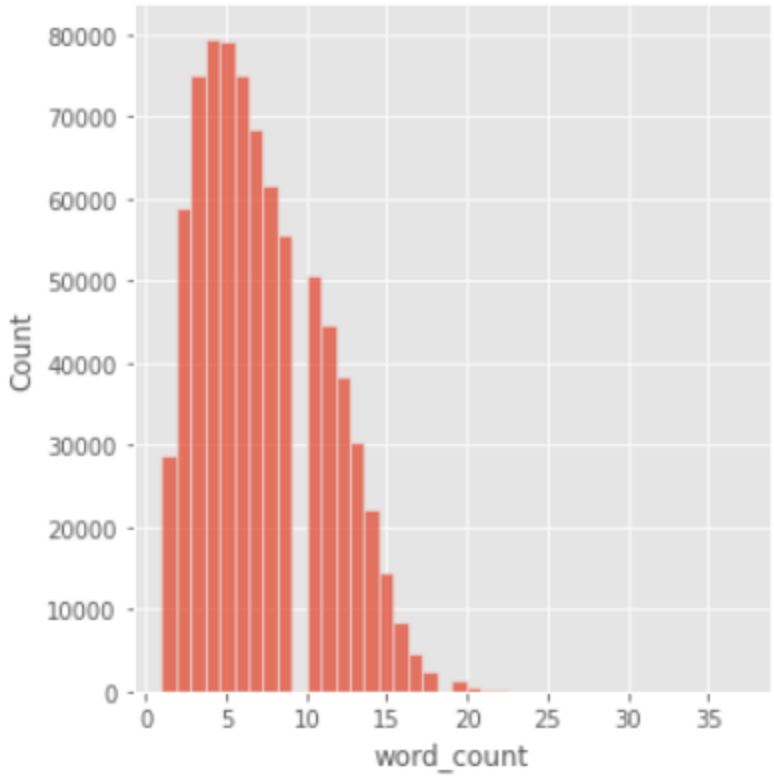
800000 rows × 3 columns

Hình 23. Thêm cột word_count cho những bài post tiêu cực

Sau khi có được cột word_count, nhóm import thư viện seaborn để vẽ biểu đồ phân phối, sử dụng hàm sns.displot với x là dữ liệu số lượng từ trong cột word_count. Ta thu được kết quả như hình.

```
[ ]  import seaborn as sns
     sns.displot(df_negative['word_count'], bins=40, kde=False)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f33772664f0>
```



Hình 24. Biểu đồ phân phối về số lượng từ trong các bài post tiêu cực

Qua biểu đồ phân phối trên nhóm rút ra được kết luận rằng, độ dài (lượng từ) trung bình của các bài Tweet tiêu cực sẽ dao động từ 5 từ đến 7 từ, phân phối cao nhất rơi vào tầm khoảng 7 từ.

- Đối với các bài tweet tích cực:

Tương tự đối với bài post tiêu cực, nhóm cũng sẽ tạo một dataframe mới với 'target' = 4 và lấy tên dataframe là df_positive. Sau đó, thêm vào cột word_count mới là số lượng từ trong bài post tích cực bằng hàm apply() và vẫn truyền vào hàm wl được định nghĩa ở trên. Ta sẽ thu được kết quả như sau.

```
[ ]  df_positive['word_count']=df_positive['final_text'].apply(wl)
     df_positive
```

	target	final_text	word_count
800000	4	love u guy r best	5
800001	4	im meeting one besties tonight cant wait girl ...	9
800002	4	thanks twitter add sunisa got meet hin show dc...	11
800003	4	sick really cheap hurt much eat real food plus...	12
800004	4	effect everyone	2
...	...	...	...
1599995	4	woke school best feeling ever	5
1599996	4	thewdb com cool hear old walt interview	7
1599997	4	ready mojo makeover ask detail	5
1599998	4	happy birthday boo time tupac amaru shakur	7
1599999	4	happy charitytuesday	2

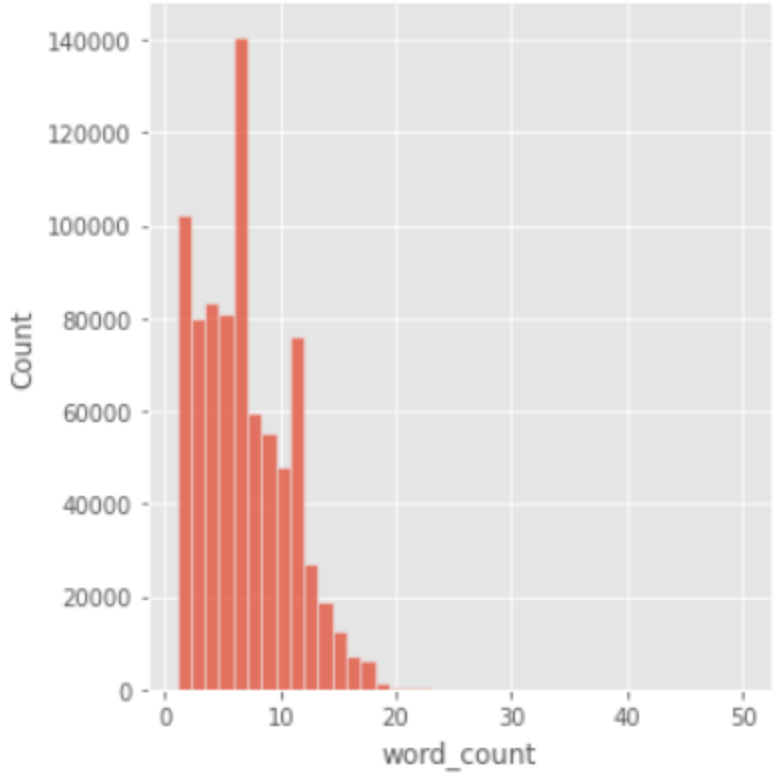
800000 rows × 3 columns

Hình 25. Thêm cột word_count cho những bài post tích cực

Sau khi có được cột word_count cho những bài post tích cực, nhóm vẫn tiếp tục sử dụng hàm sns.displot với x là dữ liệu số lượng từ trong cột word_count. Ta thu được kết quả như hình.

```
[ ]  sns.displot(df_positive['word_count'], bins=40, kde=False)
```

```
<seaborn.axisgrid.FacetGrid at 0x7f3377411040>
```



Hình 26. Biểu đồ phân phối về số lượng từ trong các bài post tích cực

Biểu đồ phân phối trên đã cho thấy rằng, độ dài (lượng từ) trung bình của các bài Tweet tích cực sẽ dao động vào khoảng gần 7 từ, phân phối cao nhất rơi vào 2 từ, 7 từ và 11 từ.

Word Cloud, hay còn gọi là Tag Cloud, là một dạng biểu diễn trực quan dưới dạng đồ họa về tần suất xuất hiện của các từ, qua đó làm nổi bật các từ có độ phổ biến cao trong dữ liệu hoặc văn bản gốc. Những dấu hiệu cho thấy sự khác biệt về tần suất hoặc độ phổ biến của từ là kích thước, màu sắc của từ hiển thị trên biểu đồ; từ nào có tần suất xuất hiện càng cao thì kích thước của từ đó sẽ càng lớn, và nếu hình vẽ sử dụng màu sắc để phân biệt mức độ thì các từ này sẽ có màu đậm nhất.

Trong quá trình phân tích ngôn ngữ tự nhiên, Python có sẵn thư viện wordcloud để hỗ trợ cho việc trực quan hóa độ phổ biến của các từ. Để sử dụng thư viện này, nhóm sẽ khai báo thư viện và tải một số hàm cần thiết gồm WordCloud, ImageColorGenerator, STOPWORDS.

```
from wordcloud import WordCloud
from wordcloud import ImageColorGenerator
from wordcloud import STOPWORDS
```

Hình 27. import thư viện wordcloud và các hàm quan trọng

Nhóm tiến hành tìm các từ phổ biến trong 2 nhóm bài viết được gán nhãn tích cực và tiêu cực. Để làm được công việc này, bước đầu nhóm sẽ lập 2 bảng dữ liệu cho từng loại bài viết. Bảng dữ liệu này sẽ gồm cột nhãn bài viết (target) và nội dung bài viết đã được tiền xử lý (final_text).

```
data_ne = pd.DataFrame()
data_ne = df1[df1.target == 0]
data_ne
```

	target	final_text
0	0	aww bummer shoulda got david carr third day EM...
1	0	upset update facebook texting might cry result...
2	0	dived many time ball managed save rest go bound
3	0	whole body feel itchy like fire
4	0	behaving mad see
...	...	...
799995	0	sick spending day laying bed listening
799996	0	gmail
799997	0	rest peace farrah sad
799998	0	sound like rival flagging ad much though
799999	0	resit exam summer wish worked harder first yea...

800000 rows × 2 columns

Hình 28. bảng dữ liệu các bài viết được gán nhãn tiêu cực

```
data_po = pd.DataFrame()
data_po = df1[df.target == 4]
data_po
```

	target	final_text
800000	1	love u guy r best
800001	1	im meeting one besties tonight cant wait girl ...
800002	1	thanks twitter add sunisa got meet hin show dc...
800003	1	sick really cheap hurt much eat real food plus...
800004	1	effect everyone
...	...	...
1599995	1	woke school best feeling ever
1599996	1	thewdb com cool hear old walt interview
1599997	1	ready mojo makeover ask detail
1599998	1	happy birthday boo time tupac amaru shakur
1599999	1	happy charitytuesday

800000 rows × 2 columns

Hình 29. bảng dữ liệu các bài viết được gán nhãn tích cực

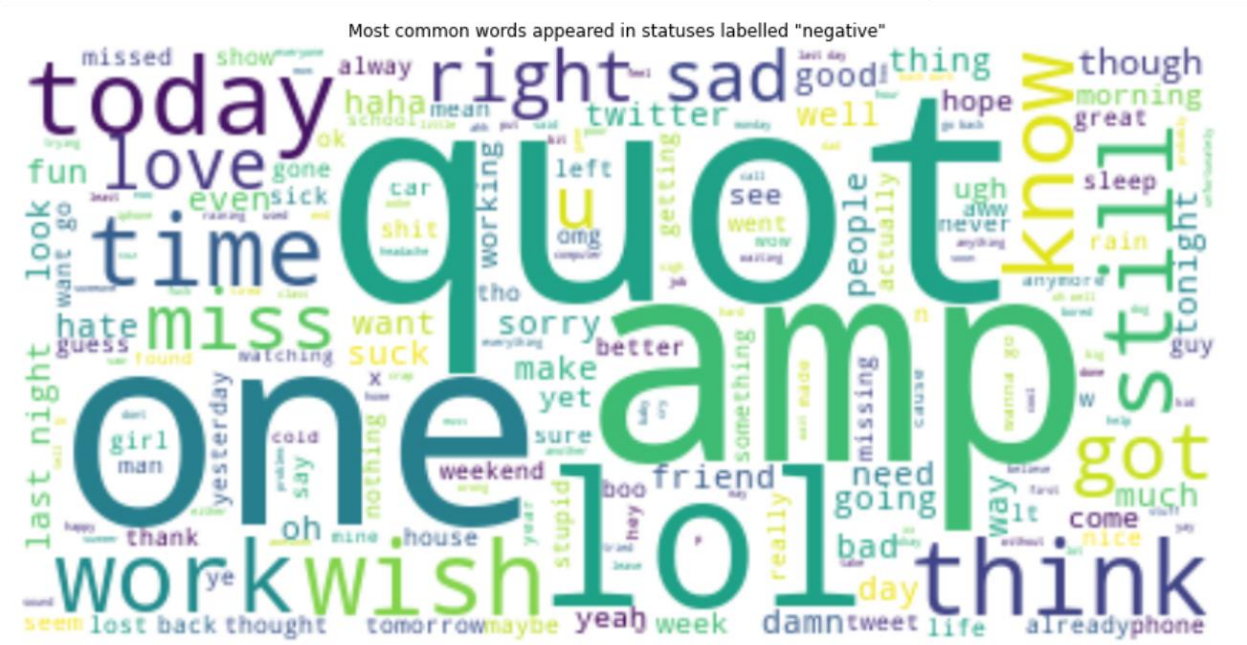
Khi đã có được 2 bảng dữ liệu, nhóm sẽ tiến hành biểu diễn word cloud cho từng loại bài viết. Quá trình biểu diễn đồ họa là như nhau đối với 2 bảng dữ liệu, bao gồm các câu lệnh sau:

```
text_ne = " ".join(i for i in data_ne['final_text'])
stopwords = set(STOPWORDS)
wordcloud_ne = WordCloud(stopwords=stopwords, background_color="white").generate(text_ne)
plt.figure( figsize=(15,10))
plt.imshow(wordcloud_ne, interpolation='bilinear')
plt.title('Most common words appeared in statuses labelled "negative" ')
plt.axis("off")
plt.show()
```

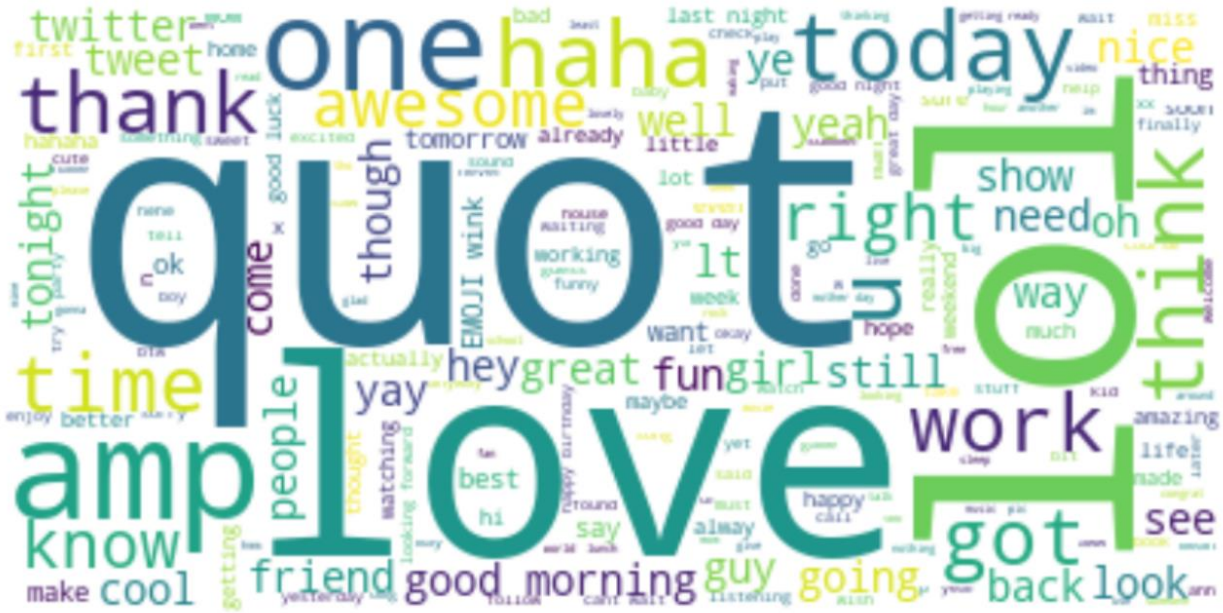
Hình 30. code xây dựng wordcloud

- Nối các nội dung bài viết ở cột "final_text" lại với nhau thành một chuỗi văn bản text, 2 nội dung bài viết cách nhau bởi một dấu cách.
- Dùng hàm stopwords để lập một danh sách (set) các từ hay sử dụng như 'a', 'the', 'an',...; việc định dạng danh sách stopwords ở dạng set() nhằm loại bỏ sự trùng lặp các từ khi tổng hợp danh sách, vì trong set(), mỗi phần tử chỉ xuất hiện 1 lần duy nhất.
- Xây dựng đồ họa wordcloud bằng hàm WordCloud() với các tham số:
    - stopwords: danh sách stopwords vừa tìm ở trên
    - background_color: chọn màu nền cho hình
    - .generate(text): áp dụng việc đồ họa wordcloud trên chuỗi text
- Một số điều chỉnh khác cho đồ họa như kích thước, tên đồ họa, chiều trên đồ họa
- Khai báo trình chiếu đồ họa wordcloud plt.imshow với 2 tham số là loại đồ họa cần biểu diễn và phương pháp nội suy hình ảnh (interpolation) muốn áp dụng

Sau khi hoàn tất việc khai báo các câu lệnh, đây là kết quả wordcloud của hai nhóm bài viết:



Hình 31. wordcloud của nhóm bài viết tiêu cực



Hình 32. wordcloud của nhóm bài viết tích cực

Các từ phổ biến nhất trong nhóm bài viết tiêu cực là: quot, amp, one, lol, think, work, miss, today, time, wish -> những từ này không thể hiện rõ tâm trạng buồn chán, tiêu cực của người đăng. Tuy nhiên các từ thể hiện rõ sự tiêu cực như sad, hate, bad, suck,... có thể dễ dàng nhìn ra trong hình trên, tần suất các từ này xuất hiện trong các bài đăng được gán nhãn "tiêu cực" ở mức khá thường xuyên.

Các từ phổ biến nhất trong nhóm bài viết tích cực là: love, quot, amp, lol, one, haha, thank, awesome, well. Đây là những từ thể hiện rõ sự tích cực, tâm trạng vui vẻ của người

đăng. Các từ khác cũng mang ý nghĩa tích cực và có tần suất xuất hiện ở mức khá thường xuyên đó là nice, great, good morning, fun, yay, cool.

Emoji là một dạng đồ họa (hình ảnh, hình động) thể hiện cảm xúc, ý tưởng hoặc tâm trạng nhúng trong văn bản. Đây là một từ tiếng Nhật được ghép từ chữ e (hình ảnh) và moji (chữ cái, ký tự). Ví dụ, khi ta muốn thể hiện sự vui vẻ, gửi cho đối phương một khuôn mặt cười, thì ta có thể nhập các ký tự như :), =) thì thiết bị của bạn sẽ tự động chuyển đổi từ dạng ký tự sang dạng hình ảnh nếu ký tự đó nằm trong bộ từ điển emoji của hệ thống. Đôi khi từ ngữ không thể hiện rõ tâm trạng của người viết, và emoji chính là một phương tiện tuyệt vời để thể hiện đúng cường độ cảm xúc mà người viết muốn bày tỏ.

Nhóm lập một bảng dữ liệu con gồm 3 cột: nhãn bài viết ‘target’, nội dung bài viết ‘final_text’ và số emoji sử dụng trong bài viết ‘emoji_count’.

```
1 data_icon = df[['target', 'final_text', 'emoji_count']]
2 data_icon
```

	target	final_text	emoji_count
0	0	aww bummer shoulda got david carr third day EM...	1
1	0	upset update facebook texting might cry result...	0
2	0	dived many time ball managed save rest go bound	0
3	0	whole body feel itchy like fire	0
4	0	behaving mad see	0
...	...	...	...
1599995	4	woke school best feeling ever	0
1599996	4	thewdb com cool hear old walt interview	0
1599997	4	ready mojo makeover ask detail	0
1599998	4	happy birthday boo time tupac amaru shakur	0
1599999	4	happy charitytuesday	0

1600000 rows × 3 columns

Hình 33. Bảng dữ liệu emoji

Sau đó, nhóm sẽ thống kê số lượng các bài viết sử dụng theo số emoji xuất hiện trong bài, phân loại theo 2 nhóm bài viết tích cực và tiêu cực. Đối với các ô chứa giá trị rỗng, nhóm sẽ xử lý và thay thế bằng giá trị '0'.

```
d_emoji = pd.DataFrame(index=index_emo)
emo_ne_gb = data_icon[data_icon['target'] == 0].groupby ('emoji_count')['emoji_count'].count()
d_emoji['Count (Ne)'] = emo_ne_gb
emo_po_gb = data_icon[data_icon['target'] == 4].groupby ('emoji_count')['emoji_count'].count()
d_emoji['Count (Po)'] = emo_po_gb
display(d_emoji)
```

	Count (Ne)	Count (Po)
0	792594	791708.0
1	7155	8101.0
2	203	178.0
3	34	11.0
4	7	1.0
5	4	NaN
6	1	1.0
10	1	NaN
11	1	NaN

Hình 34. thống kê số lượng bài tweet sử dụng số emoji tương ứng trong 2 nhóm tích cực và tiêu cực

```
d_emoji = d_emoji.fillna(0)
d_emoji
```

Hình 35. xử lý missing values

	Count (Ne)	Count (Po)
0	792594	791708.0
1	7155	8101.0
2	203	178.0
3	34	11.0
4	7	1.0
5	4	0.0
6	1	1.0
10	1	0.0
11	1	0.0

Hình 36. bảng thống kê emoji sau khi xử lý missing values

Số lượng emoji xuất hiện trong các status ở mỗi nhóm tích cực và tiêu cực có sự chênh lệch. Trong nhóm bài viết tích cực, phần lớn các bài viết không sử dụng emoji hoặc dùng 1 emoji. Còn với các bài viết tiêu cực, khi dùng emoji để thể hiện tâm trạng, số lượng dùng emoji dao động từ 1-3, và có thể sử dụng tận 10-11 emoji trong 1 bài viết.

Để tìm hiểu sâu hơn đối với từng nhóm bài viết, emoji nào được sử dụng nhiều, nhóm sẽ ứng dụng WordCloud.

Nhóm lập một bảng dữ liệu gồm các cột phân loại ‘target’, nội dung bài viết ‘final_text’, số lượng emoji trong bài viết ‘emoji_count’, nội dung các emoji ‘emoji_list’.

```
data_icon = df[['target', 'final_text', 'emoji_count', 'emoji_list']]
data_icon
```

	target	final_text	emoji_count	emoji_list
0	0	aww bummer shoulda got david carr third day EM...	1	[llo]
1	0	upset update facebook texting might cry result...	0	[]
2	0	dived many time ball managed save rest go bound	0	[]
3	0	whole body feel itchy like fire	0	[]
4	0	behaving mad see	0	[]
...	...	...	...	...
1599995	4	woke school best feeling ever	0	[]
1599996	4	thewdb com cool hear old walt interview	0	[]
1599997	4	ready mojo makeover ask detail	0	[]
1599998	4	happy birthday boo time tupac amaru shakur	0	[]
1599999	4	happy charitytuesday	0	[]

1600000 rows × 4 columns

Hình 37. bảng dữ liệu nội dung emoji

Vì dữ liệu trong cột emoji_list đang ở dạng list - danh sách, nhóm sẽ chuyển về dạng dữ liệu chuỗi string bằng hàm combine_text() đã được định nghĩa ở trên.

```
data_icon['emoji_list'] = data_icon['emoji_list'].apply(lambda x : combine_text(x))
data_icon.head()
```

	target	final_text	emoji_count	emoji_list
594642	0	hate job scratch hate life EMOJI shocked EMOJI...	11	shocked shocked shocked shocked shocked shocke...
594449	0	going meet best friend chris EMOJI wink EMOJI ...	10	wink wink wink wink wink wink wink wink wink wink
679678	0	EMOJI wink p gt EMOJI surprised EMOJI confused...	6	wink surprised confused confused sad smile
1060248	4	hey take look zing wish EMOJI wink EMOJI wink ...	6	wink wink wink wink wink wink
623897	0	missed radiohead EMOJI shocked EMOJI shocked E...	5	shocked shocked shocked shocked shocked

Hình 38. Chuyển từ list sang chuỗi

Các bước để trực quan hóa sử dụng hàm WordCloud() cho cột emoji_list tương tự như khi áp dụng cho nội dung bài viết, thông qua các bước thiết lập bảng dữ liệu cho 2 nhóm phân loại bài đăng, nối chuỗi, xây dựng đồ họa và trình bày.

```
data_icon_ne = pd.DataFrame()
data_icon_ne = data_icon[data_icon.target == 0]
```

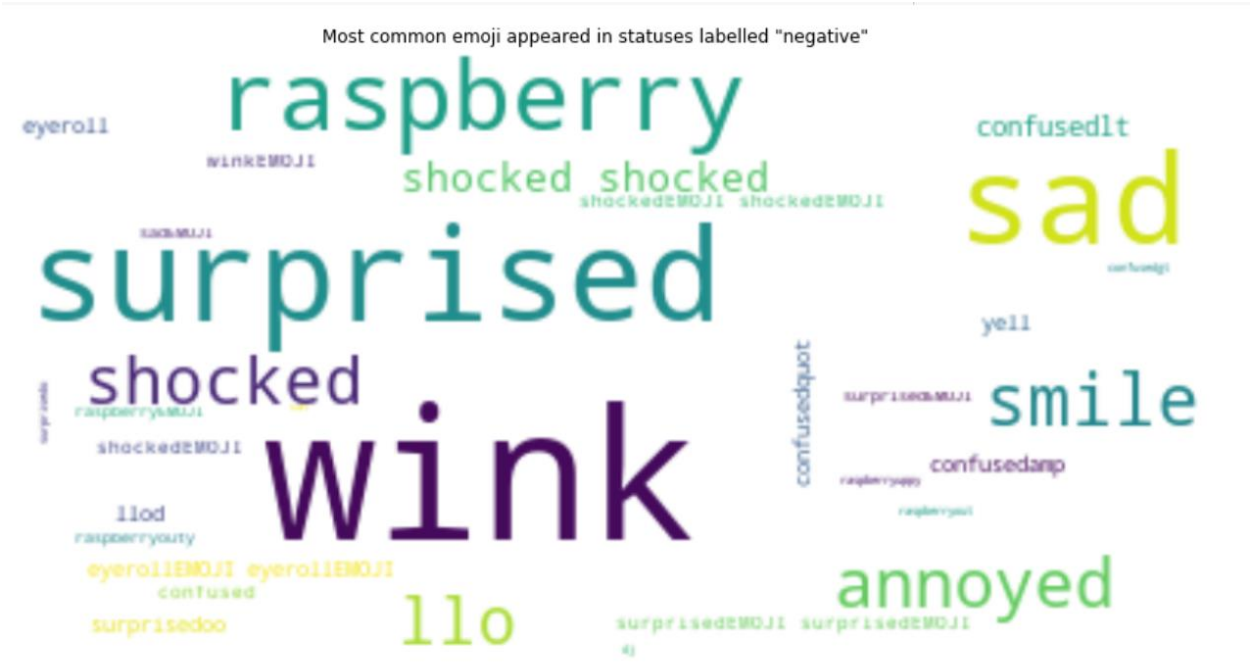
```
icon_ne = " ".join(i for i in data_icon_ne['emoji_list'])
stopwords = set(STOPWORDS)
wordcloud_icon_ne = WordCloud(stopwords=stopwords, background_color="white").generate(icon_ne)
plt.figure( figsize=(15,10))
plt.imshow(wordcloud_icon_ne, interpolation='bilinear')
plt.title('Most common emoji appeared in statuses labelled "negative" ')
plt.axis("off")
plt.show()
```

Hình 39. code emojicloud - negative tweets

```
data_icon_po = pd.DataFrame()
data_icon_po = data_icon[data_icon.target == 4]
```

```
icon_po = " ".join(i for i in data_icon_po['emoji_list'])
stopwords = set(STOPWORDS)
wordcloud_icon_po = WordCloud(stopwords=stopwords, background_color="white").generate(icon_po)
plt.figure( figsize=(15,10))
plt.imshow(wordcloud_icon_po, interpolation='bilinear')
plt.title('Most common emoji appeared in statuses labelled "positive" ')
plt.axis("off")
plt.show()
```

Hình 40. code emojicloud - positive tweets



Hình 41. EmojiCloud – negative

Hình 42. EmojiCLoud – positive

Các emoji xuất hiện nhiều nhất trong các bài viết tích cực là:

- smile: ':)', ':-)', ':^)'
- lol: ';p'
- surprised: ':o'
- wink: ';)', ';-)', ';p'
- raspberry: ':p'

⇨ Các emoji thể hiện rõ tâm trạng tích cực như smile, lol, wink, raspberry xuất hiện nhiều trong các bài viết được gán nhãn tích cực.

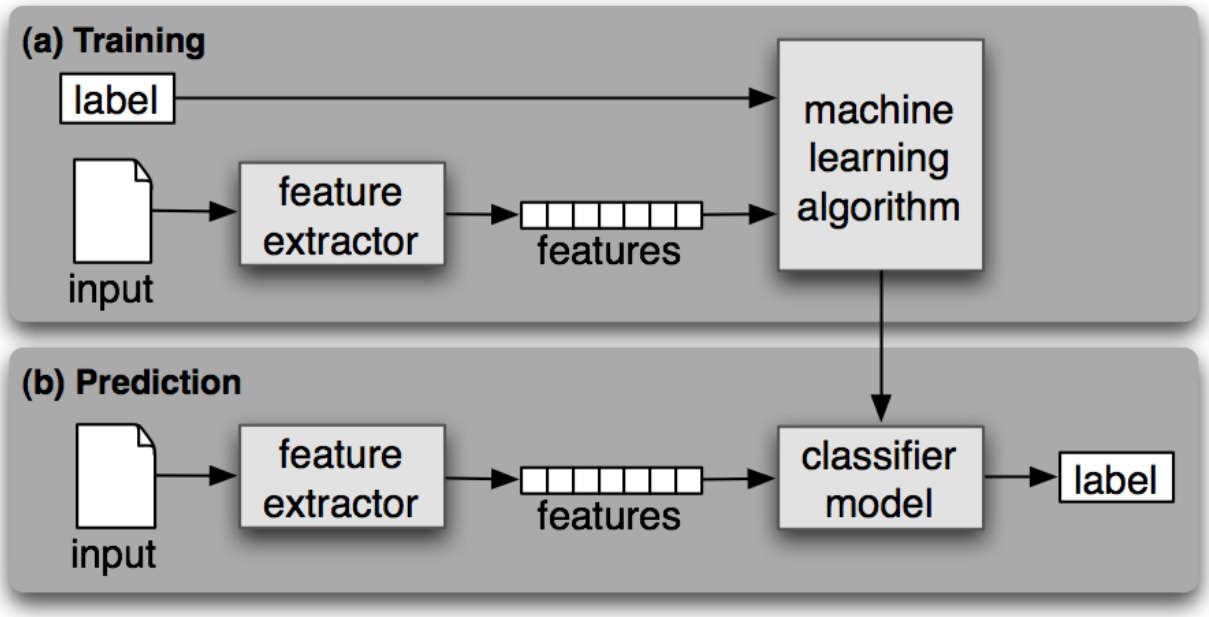
Các emoji xuất hiện nhiều nhất trong các bài viết tiêu cực là:

- sad: ':(', ':'(', '* sight *', ':-(', ':-<'
- surprised: ':o'
- shocked: ':-@', ':@'
- wink: ';)', ';-)'
- raspberry: ':p'
- annoyed: ':\\'

Các emoji thể hiện rõ tâm trạng tiêu cực như sad, shocked, annoyed xuất hiện nhiều trong các bài viết được gán nhãn tiêu cực.

➔ Điều này cho thấy việc thể hiện cảm xúc, tâm trạng qua hình vẽ, emoji dễ hơn là qua câu từ trực tiếp.

3.3. Xây dựng mô hình



Hình 43. quy trình huấn luyện mô hình

3.3.1 Phân chia dữ liệu

Thực hiện lấy mẫu để chạy mô hình, vì dữ liệu khá lớn (hơn 1,6 triệu dòng) nên ở bài toán này nhóm sẽ lấy mẫu là 5% so với tổng thể để mô hình hoạt động tốt hơn, cũng như tăng tốc độ xử lý dữ liệu

```
: X=data.final_text
  y=data.target
```

```
: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.05, random_state=2)
```

Hình 44. Lấy mẫu training set và testing set

Sau khi tách, thực hiện vector hóa các chuỗi ký tự thành các feature, với giạn hạn số lượng feature là 100000 features

```
vectoriser = TfidfVectorizer(ngram_range=(1,2), max_features=100000)
vectoriser.fit(X_train)
print(f'Vectoriser fitted.')
print('No. of feature_words: ', len(vectoriser.get_feature_names()))
```

```
Vectoriser fitted.
No. of feature_words:  100000
```

```
X_train = vectoriser.transform(X_train)
X_test  = vectoriser.transform(X_test)
print(f'Data Transformed.')
```

```
Data Transformed.
```

Hình 45. Word2Vec

Thực hiện gọi hàm đánh giá mô hình với các chỉ số được đặt trong hàm là confusion matrix và các chỉ số đánh giá phân lớp (precision, recall, classification accuracy, và f-measure.)

Evaluate model funtion

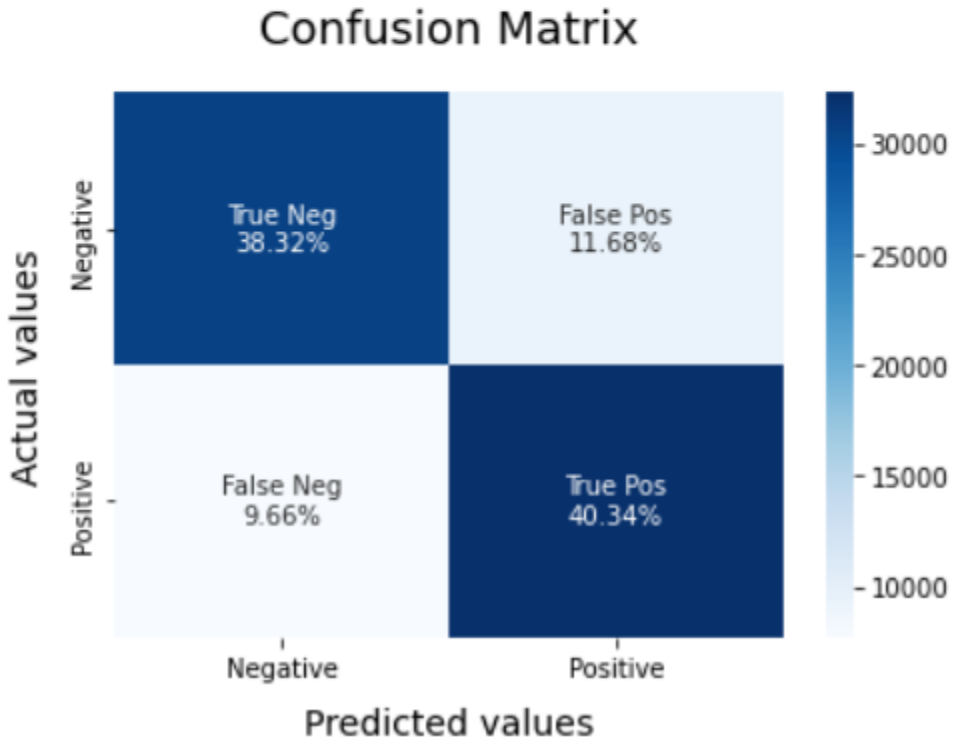
```
def model_Evaluate(model):

    # Predict values for Test dataset
    y_pred = model.predict(X_test)

    # Print the evaluation metrics for the dataset.
    print(classification_report(y_test, y_pred))

    # Compute and plot the Confusion matrix
    cf_matrix = confusion_matrix(y_test, y_pred)

    categories  = ['Negative','Positive']
    group_names = ['True Neg','False Pos', 'False Neg','True Pos']
    group_percentages = ['{0:.2%}'.format(value) for value in cf_matrix.flatten() / np.sum(cf_matrix)]

    labels = [f'{v1}\n{v2}' for v1, v2 in zip(group_names,group_percentages)]
    labels = np.asarray(labels).reshape(2,2)

    sns.heatmap(cf_matrix, annot = labels, cmap = 'Blues',fmt = '',
                xticklabels = categories, yticklabels = categories)

    plt.xlabel("Predicted values", fontdict = {'size':14}, labelpad = 10)
    plt.ylabel("Actual values"   , fontdict = {'size':14}, labelpad = 10)
    plt.title ("Confusion Matrix", fontdict = {'size':18}, pad = 20)
```

Hình 46. Hàm đánh giá các mô hình

3.3.2 Huấn luyện dữ liệu và đánh giá mô hình

```
SVCmodel = LinearSVC()
SVCmodel.fit(X_train, y_train)
model_Evaluate(SVCmodel)
```

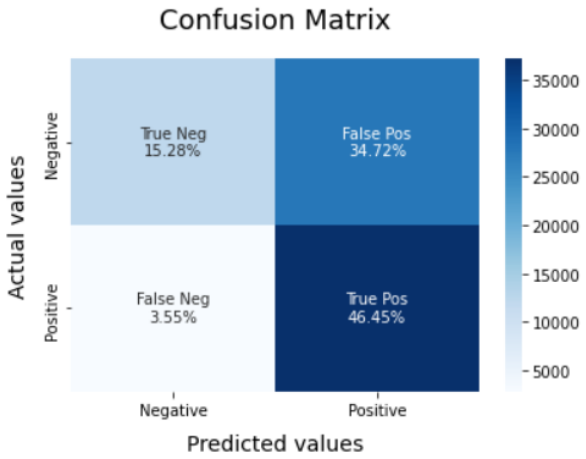
	precision	recall	f1-score	support
0	0.80	0.77	0.78	39999
1	0.78	0.81	0.79	40001
accuracy			0.79	80000
macro avg	0.79	0.79	0.79	80000
weighted avg	0.79	0.79	0.79	80000



Hình 47. đánh giá Linear Support Vector Machine

```
: from sklearn.tree import DecisionTreeClassifier
dtree_2 = DecisionTreeClassifier(max_depth = 20, random_state= 41,
                                 max_features =None , min_samples_leaf = 30).fit(X_train, y_train)
model_Evaluate(dtree_2)
```

```
              precision    recall  f1-score   support

           0       0.81      0.31      0.44     39999
           1       0.57      0.93      0.71     40001

    accuracy                           0.62     80000
   macro avg       0.69      0.62      0.58     80000
weighted avg       0.69      0.62      0.58     80000
```



Hình 48. đánh giá Decision Tree Classifer

⇒ Kết luận: Dựa theo các kết quả ở trên có thể thấy được rằng mô hình LinearSVM là phù hợp nhất cho bộ dữ liệu này với các chỉ số, keys metric đều tốt hơn mà nằm trong mức đánh giá khá tốt cho việc dự báo này.

3.3.3 Phân tích kết quả dự đoán

Sau khi đã chọn được mô hình LinearSVM là mô hình phù hợp nhất cho bộ dữ liệu này, nhóm phân loại các bài viết trong tập mẫu X_test với mô hình LinearSVM vừa huấn luyện bằng hàm SVCmodel.predict()

```
predictions_SVM = SVCmodel.predict(X_test)
```

Hình 49. Dự đoán testing set bằng mô hình SVM

Nhóm lưu kết quả dự đoán thành 1 cột mới ‘SVM predictions’ vào bảng dữ liệu data_modelSVM, đồng thời bổ sung cột phân loại gốc Y_test.

```
data_modelSVM['SVM prediction'] = predictions_SVM
```

```
data_modelSVM['Original Label'] = Y_test
```

Nhóm tạo thêm một cột mới "Correction" trong bảng dữ liệu data_modelSVM chứa kết quả so sánh giữa dữ liệu dự đoán theo mô hình SVM với dữ liệu phân loại gốc, nếu dự đoán đúng thì sẽ trả về giá trị "Right", ngược lại sẽ là "Wrong".

```
data_modelSVM['Correction'] = data_modelSVM.apply(lambda x: 'Right' if x['SVM prediction'] == x['Original Label'] else 'Wrong', axis=1)
```

```
data_modelSVM
```

	Status	SVM prediction	Original Label	Correction
670081	oh im going wow mona vale real place afterall ...	0	0	Right
408251	baby growing	0	0	Right
1559739	painted black rolling stone best	1	1	Right
571248	kk logging byezz	1	0	Wrong
524639	shitty shitty shitty news today	0	0	Right
...	...	...	...	...
380683	hacked mine username kristenjstewart make one	1	0	Wrong
475445	hate smell hospital brings back bad memory	0	0	Right
1413672	enough office buy something tasty supper back ...	1	1	Right
1048028	yeah reinstall find login let know mate	0	1	Wrong
1049616	mega brain hilarity turning general mundanenes...	1	1	Right

32000 rows × 4 columns

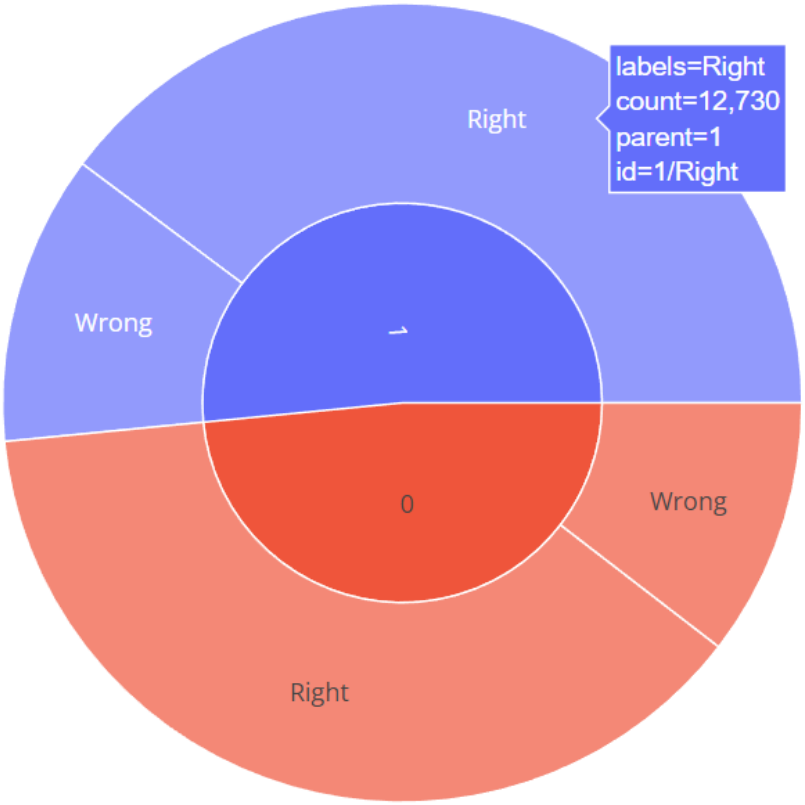
Hình 50. Đối chiếu kết quả dự đoán với dữ liệu phân loại gốc

```
print('Tỷ lệ dự đoán đúng:', str(round(data_modelSVM[data_modelSVM['Correction']=='Right']['Correction'].count()/len(data_modelSVM),2)*100)+'%')
```

```
Tỷ lệ dự đoán đúng: 78.0%
```

Cụ thể hơn tỷ lệ giữa việc đoán đúng và sai giữa 2 nhóm bài đăng tích cực và tiêu cực, nhóm sử dụng biểu đồ hình tròn ghép để trực quan hóa kết quả phân tích.

```
import plotly.express as px
fig = px.sunburst(data_modelSVM,
                  path=["SVM prediction", 'Correction'],
                  )
fig.show()
```



Hình 51. Biểu đồ thể hiện tỷ lệ đoán đúng-sai giữa 2 nhóm phân loại bài tweet

⇨ Tỷ lệ phân loại đúng của cả 2 nhóm tích cực và tiêu cực đều chiếm khoảng 4/5 trong từng nhóm.

Các nhóm từ xuất hiện nhiều trong từng nhóm phân loại sẽ được biểu diễn qua hàm wordcloud. Các bước đồ họa hóa giống với cách áp dụng ở phần khai phá dữ liệu, đó là lập 2 bảng dữ liệu cho 2 nhóm phân loại, nối chuỗi và trình bày.

```
data_model_ne = pd.DataFrame()
data_model_ne = data_modelSVM[data_modelSVM['SVM prediction'] == 0]
data_model_ne
```

```
text_ne = " ".join(i for i in data_model_ne['Status'])
stopwords = set(STOPWORDS)
wordcloud_ne = WordCloud(stopwords=stopwords, background_color="white").generate(text_ne)
plt.figure(figsize=(15,10))
plt.imshow(wordcloud_ne, interpolation='bilinear')
plt.title('Most common words appeared in statuses labelled "negative" ')
plt.axis("off")
plt.show()
```

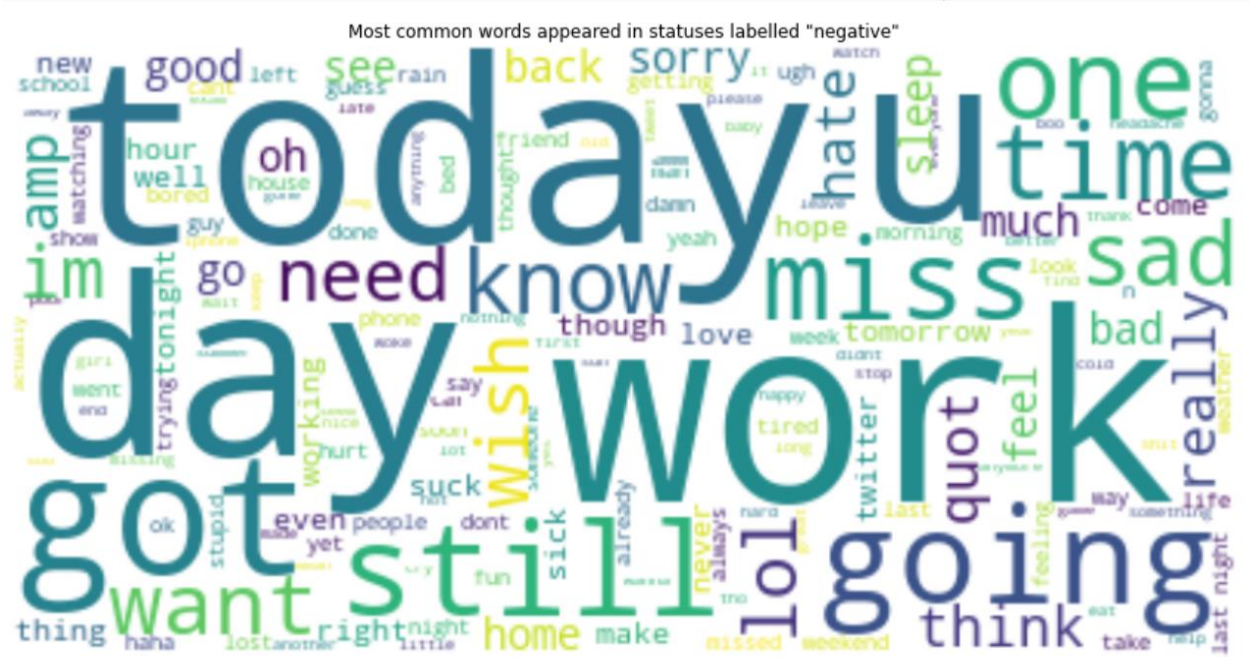
Hình 52. Code wordcloud cho nhóm tiêu cực (mô hình)

```
data_model_po = pd.DataFrame()
data_model_po = data_modelSVM[data_modelSVM['SVM prediction'] == 1]
data_model_po
```

```
text_po = " ".join(i for i in data_model_po['Status'])
stopwords = set(STOPWORDS)
wordcloud_po = WordCloud(stopwords=stopwords, background_color="white").generate(text_po)
plt.figure(figsize=(15,10))
plt.imshow(wordcloud_po, interpolation='bilinear')
plt.title('Most common words appeared in statuses labelled "positive" ')
plt.axis("off")
plt.show()
```

Hình 53. Code wordcloud cho nhóm tích cực (mô hình)

Đây là kết quả wordcloud của 2 nhóm bài viết:



Hình 54. wordcloud nhóm negative

Hình 55. wordcloud nhóm positive

Những từ phổ biến nhất đối với các bài viết được phân loại là "tiêu cực" gồm: today, day, got, one, work, u, stil, know, going. Những từ này không thể hiện quá rõ cảm xúc tiêu cực của người viết. Các từ thể hiện rõ thái độ tiêu cực như hate, sad, sorry, wish, tired, bad tuy có xuất hiện trong các bài viết thuộc nhóm tiêu cực này nhưng ở tần suất vừa đến thấp, không được sử dụng nhiều.

Những từ phổ biến nhất trong nhóm các bài đăng tích cực là: love, quot, thank, u, lol, good, well, today, haha, day, know, time. Phần lớn các từ trên đều thể hiện rõ tâm trạng vui vẻ, thoải mái, tích cực của người viết.

CHƯƠNG 4. KẾT LUẬN

4.1. Các Kết Quả Đạt Được

Qua quá trình phân tích và đánh giá, nhóm đã đạt được các kết quả sau:

- Trong phần tiền xử lý dữ liệu: nhóm đã gần như làm sạch được dữ liệu, loại bỏ các yếu tố gây nhiễu, dư thừa như các từ nối, dấu cách dư, các ký tự lặp lại, và đặc biệt đã xử lý được emoji, một nhân tố quan trọng giúp nhóm phát hiện được xu hướng của nhóm tiêu cực – trầm cảm.
- Trong phần biểu diễn trực quan, nhóm đã thể hiện được các thống kê, phát biểu cơ bản về NLP như số từ trong một bài tweet, các từ phổ biến qua WordCLoud, các emoji xuất hiện nhiều trong 2 loại bài tweet
- Trong phần đánh giá mô hình, nhóm đã xây dựng thành công 2 mô hình SVM và Decision Tree, tính được các chỉ số đánh giá để lựa ra được mô hình tốt nhất dự đoán bộ dữ liệu. Kết quả dự đoán cũng không quá chênh lệch so với dữ liệu gốc.

Thông qua bài đồ án, nhóm rút ra được 2 ý lớn về tình trạng trầm cảm trong các dòng trạng thái trên Twitter như sau:

- Người dùng có xu hướng che dấu cảm xúc, tâm trạng thật của họ; đối với nhóm bài tweet của những người tiêu cực, họ không bộc lộ rõ tâm trạng của họ qua từ ngữ, do vậy wordcloud của nhóm tiêu cực không có sự khác biệt mấy so với nhóm tích cực. Nguyên do cho sự che dấu này đó là vì một phần họ không muốn những người bạn trên mạng xã hội bị tiếp nhận năng lượng tiêu cực từ họ, cũng như không muốn người khác lo lắng về tình trạng sức khỏe tinh thần của họ. Đồng thời, có đôi khi việc biểu lộ tâm trạng qua từ ngữ khá khó khăn, không thể hiện hết được mức độ trầm cảm, tiêu cực của người viết.
- Tuy nhiên, thông qua các sử dụng emoji trong bài tweet, ta có thể cảm nhận được tâm trạng của người viết tại thời điểm đó như thế nào. Emoji là một dạng hình ảnh thể hiện các cung bậc cảm xúc, thế nên chỉ với 1 emoji nhỏ nhắn cũng đủ thể hiện được khá chính xác và trọn vẹn cảm xúc của người dùng mà không gây phản cảm hay khó hiểu cho người đọc

4.2. Những Hạn Chế và Hướng Phát Triển

- Nhóm chưa ứng dụng việc phân tích cấu trúc ngữ pháp, gán nhãn từ trong bài đồ án này, vì các bài tweet trên mạng xã hội thường là một cụm từ, không theo cú pháp chuẩn, sử dụng nhiều từ lóng nên việc phân tích sẽ khó khăn hơn nhiều
- Nhóm chỉ mới làm việc với dữ liệu dạng chuỗi, kí tự, chưa làm việc với dữ liệu dạng hình ảnh.
- Để có những phân tích sâu hơn về tình trạng trầm cảm, nhóm có thể thêm các yếu tố khác như tần suất đăng bài, thời điểm đăng bài, các loại bài đăng mà người dùng đã thả tim,…

- Bộ dữ liệu của nhóm có kích thước là khoảng 1,6 triệu bài tweet, nhưng khi chạy mô hình thì chỉ lấy 5%, một tỉ lệ khá thấp và có thể không đại diện hết cho tổng thể, nhưng vì giới hạn của các mô hình nên nhóm phải lấy mẫu số lượng ít. Nhóm nên tìm hiểu thêm các thuật toán có thể làm việc với số lượng phần tử đông hơn.

TÀI LIỆU THAM KHẢO

[NLP] Mã nguồn bài đồ án nhóm 3. (n.d.). GitHub. Retrieved December 15, 2022, from https://github.com/camtu-1310/NLP-FINAL_PROJECT?fbclid=IwAR3k8efgNmuCPD2sE19JX7Zn8fwYLox422XkTGtH3exyqx1TtOSZym-vZmo

Giới thiệu về Support Vector Machine (SVM). (n.d.). Viblo. Retrieved December 15, 2022, from https://viblo.asia/p/gioi-thieu-ve-support-vector-machine-svm-6J3ZgPVElmB

Python Word Clouds Tutorial: How to Create a Word Cloud. (n.d.). DataCamp. Retrieved December 15, 2022, from https://www.datacamp.com/tutorial/wordcloud-python

Sentiment140 dataset with 1.6 million tweets. (2009, May 16). Kaggle. Retrieved December 15, 2022, from https://www.kaggle.com/datasets/kazanova/sentiment140

Sentiment Analysis of Twitter Data. (n.d.). ACL Anthology. Retrieved December 15, 2022, from https://aclanthology.org/W11-0705.pdf

Sharma, A. (2020, April 27). *Exploratory Data Analysis For Text Data | EDA Using Python*. Analytics Vidhya. Retrieved December 15, 2022, from https://www.analyticsvidhya.com/blog/2020/04/beginners-guide-exploratory-data-analysis-text-data/

sklearn.svm.LinearSVR. (n.d.). Scikit-learn. Retrieved December 15, 2022, from https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVR.html#sklearn.svm.LinearSVR

Support vector machine. (n.d.). Wikipedia. Retrieved December 15, 2022, from https://en.wikipedia.org/wiki/Support_vector_machine#Linear_SVM

Twitter sentiment classification. (n.d.). GitHub. Retrieved December 15, 2022, from https://github.com/e-bug/twitter-sentiment-classification

BẢNG PHÂN CÔNG

Thành viên	Công việc	Mức độ tham gia
Đoàn Vũ Minh Thanh	• Tiền xử lý dữ liệu: emoji • Khai phá dữ liệu: wordcloud, emoji • Dự đoán và kết quả mô hình SVM • Chương 4: kết luận • Tổng hợp word báo cáo, nộp bài	100%
Đặng Thị Cẩm Tú	• Cơ sở dữ liệu • Tiền xử lý dữ liệu • Đánh giá các mô hình	100%
Huỳnh Thị Cẩm Nhung	• Tổng quan • Khai phá dữ liệu: wordcount, ratio of target	80%