TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN,
ĐẠI HỌC QUỐC GIA HÀ NỘI

Sentiment Analysis using PySpark on Multi Source Social-Media Data

Ngày 3 tháng 1 năm 2024

Nhóm 3

Họ và tên	**Mã sinh viên**
Dương Thanh Hải	21002138
Mai Thanh Hảo	21002140
Nguyễn Thị Hương Huế	21002148

Giảng viên hướng dẫn
Thầy Phạm Tiến Lâm
Thầy Đặng Văn Báu

Mục lục

Thuật ngữ viết tắt

STT	Thuật ngữ	Giải thích
1	API	Application Program Interface
2	CSV	Comma Separated Values
3	HTML	Hyper Text Mark-up Language
4	NLP	Natural Language Processing
5	SA	Sentiment Analysis
6	MLlib	Machine Learning library

Tóm tắt

Bài báo cáo trình bày cách triển khai thuật toán Logistic sử dụng PySpark để phân loại cảm xúc của các bài đăng trên hai trang mạng xã hội phổ biến Twitter và Reddit nhằm phân tích ý kiến của người dân về cuộc bầu cử tổng thống Ấn Độ đối với ứng cử viên Lok Sabha vào năm 2019. Đồng thời nhóm cũng đã thử nghiệm một số thuật toán học máy khác sử dụng PySpark như: Random Forest, Naive Bayes để chỉ ra sự khác biệt giữa các thuật toán này và thuật toán nhóm đề xuất. Để đánh giá kết quả của thuật toán, nhóm đã chạy các thuật toán trên 3 tập dữ liệu: Tập dữ liệu Reddit chứa 69561 mẫu, tập dữ liệu Twitter chứa 100K mẫu, tập dữ liệu kết hợp Reddit và Twitter chứa 230436 mẫu. Kết quả chỉ ra rằng thuật toán Logistics sử dụng PySpark có thể đạt được độ chính xác gấp khoảng 3 lần so với Random Forest và khoảng 1,1 lần so với Naive Bayes.

1 Đặt vấn đề

Hiện nay, các phương tiện truyền thông xã hội đang rất phổ biến và phát triển vượt bậc. Vậy nên đến mỗi đợt tổng tuyển cử, các bài đăng bầu cử tổng thống luôn được hiển thị rộng rãi mỗi ngày trên các trang mạng xã hội. [6] trong bài báo này đã chỉ ra rằng các trang web truyền thông xã hội đều là các nguồn có giá trị để khai thác ý kiến người dùng từ sản phẩm, dịch vụ đến chính trị.

Một điều quan trọng trong các cuộc tổng tuyển cử là các cuộc thăm dò và khảo sát ý kiến bầu cử. Việc tiến hành các cuộc khảo sát có thể tốn rất nhiều thời gian và tài nguyên, hơn nữa có thể dự đoán không chính xác kết quả bầu cử. Để giải quyết các vấn đề về độ chính xác và tài nguyên, chúng ta có thể khai thác từ hành vi của người dùng trên các trang mạng xã hội với lượng phản hồi và các hành động thể hiện cảm xúc thông qua các bài đăng trên nền tảng đó để dự đoán kết quả bầu cử.

Báo cáo tập trung vào việc phân tích cảm xúc cho những văn bản từ nhiều phương tiện truyền thông xã hội. Một trong số đó là Twitter - trang web lớn nhất thế giới, là nơi mà mọi người thể hiện quan điểm và ý kiến chính trị của họ thông qua 'tweet'. Cử tri không chỉ sử dụng nền tảng này để công khai ủng hộ ứng viên mà còn để bày tỏ ý kiến của mình về các vấn đề thời sự và vấn đề xảy ra trong đất nước. Do vậy mà gần đây Twitter đã trở thành một nền tảng chung cho các chính trị gia và các nhà lãnh đạo đảng để truyền tải thông điệp của họ đến người dân của quốc gia và tổ chức các chiến dịch truyền thông, quảng bá tên tuổi của mình. Vì vậy, việc phân tích cảm xúc của cử tri trên phương tiện truyền thông này sẽ cho phép chúng ta có một hình ảnh thu nhỏ về sự chính trị của đất nước đó.

Một phương tiện truyền thông xã hội khác cũng được xem xét ở đây là Reddit, đây là trang tin tức xã hội lớn nhất để tổng hợp, xếp hạng nội dung web và trang web thảo luận. Mọi người có thể đăng tin tức hoặc chủ đề thảo luận tại đây và sẽ được nhiều người trả lời thông qua bình luận. Các tin tức và chủ đề được đăng có thể bao gồm nhiều chủ đề chính trị. Mỗi bài đăng sẽ tạo ra hàng nghìn bình luận nơi mọi người bày tỏ ý kiến và bình luận cho nhận xét của người khác. Vì vậy, nếu tổng hợp tất cả các bình luận từ những bài đăng liên quan sẽ giúp ta có thể hiểu được tình trạng chính trị hiện tại của một quốc gia.

Sau khi đã có dữ liệu về các bình luận và lượt tương tác từ Twitter và Reddit, nhóm đã xây dựng và so sánh khi sử dụng các thuật toán học máy hiện đại để phân tích cảm xúc của các đoạn văn bản. Báo cáo sử dụng PySpark vào việc huấn luyện và kiểm tra mô hình. Nhóm tận dụng khả năng xử lý trong bộ nhớ của Spark kết hợp cùng việc chạy song song trong các quy trình cần thiết và cũng sử dụng tính năng đường ống của Spark để nâng cao hiệu suất mô hình.

2 Cơ sở lý thuyết [1]

Trong chương này, báo cáo sẽ trình bày về một số nghiên cứu đã được đưa ra bởi các nhà nghiên cứu trong lĩnh vực phân tích cảm xúc thông qua phương tiện truyền thông xã hội để dự đoán kết quả bầu cử với các phương pháp tiếp cận đã được đưa ra trước đó.

2.1 Phân tích cảm xúc (sentiment analysis) là gì?

Sentiment analysis hay còn được gọi là Opinion Mining là một lĩnh vực trong xử lý ngôn ngữ tự nhiên (NLP) dùng để xác định và trích xuất ý kiến từ những đoạn văn bản. Hiện nay, phân tích cảm xúc là một chủ đề rất được quan tâm và phát triển vì nó có nhiều ứng dụng thực tiễn, trong đó dự đoán bầu cử là nổi bật. Vì có một số lượng lớn các văn bản thể hiện quan điểm chính trị và quan điểm chính trị của các nhà lãnh đạo có sẵn trong các trang web đánh giá, diễn đàn, blog và phương tiện truyền thông xã hội.

2.2 Các phương pháp dùng trong lĩnh vực phân tích cảm xúc

Có nhiều phương pháp và thuật toán được dùng để phân tích cảm xúc của người người dùng nhưng có thể chia thành 3 loại chính.

2.2.1 Rule-based Method

Định nghĩa: Là phương pháp sử dụng một bộ quy tắc do con người tạo ra để giúp xác định cảm xúc của văn bản.

Các quy tắc này có thể bao gồm các kỹ thuật NLP khác nhau, ví dụ:

- Stemming, tokenization, part-of-speech tagging và parsing.
- Lexicons (danh sách các từ và cách diễn đạt của nó).

Cách hoạt động: Ở đây nhóm sẽ lấy một ví dụ nhỏ để trình bày về cách hoạt động của phương pháp Rule-based.

- Tạo 2 danh sách đối lập, một danh sách chứa các từ tiêu cực như: xấu, tệ, tồi,.. và một danh sách chứa các từ tích cực: tốt, đẹp, tuyệt vời,...
- Đếm số lượng các từ tích cực và tiêu cực có trong đoạn văn bản.
- Nếu số lượng các từ tích cực nhiều hơn các từ tiêu cực thì hệ thống sẽ trả về dự đoán cho đoạn văn bản đó là tích cực và ngược lại. Trong trường hợp số từ tích cực bằng số từ tiêu cực thì hệ thống sẽ trả về dự đoán là trung lập.

Các hệ thống dựa trên quy tắc rất đơn giản vì chúng không tính đến cách các từ được kết hợp trong một chuỗi. Tất nhiên, có thể sử dụng các kỹ thuật xử lý nâng cao hơn và thêm các quy tắc mới để hỗ trợ các cách diễn đạt và từ vựng mới. Tuy nhiên, việc thêm quy tắc mới có thể ảnh hưởng đến kết quả trước đó và toàn bộ hệ thống có thể trở nên rất phức tạp. Vì các hệ thống dựa trên quy tắc thường yêu cầu tinh chỉnh và bảo trì nên chúng cũng cần được đầu tư liên tục.

2.2.2 Automatic method

Định nghĩa: Khác với Rule-based method, phương pháp này không được tạo ra bằng các quy tắc một cách thủ công mà dựa vào các kỹ thuật học máy. Bài toán phân tích cảm xúc thường được mô hình hóa như một bài toán phân loại, nhận đầu vào là một đoạn văn bản và trả về 1 cảm xúc (tích cực, tiêu cực, trung lập).

Các phương pháp này liên quan đến việc đào tạo một mô hình học máy dựa trên một bộ dữ liệu văn bản trong đó mỗi đoạn văn bản được gán nhãn bằng cảm xúc của nó (tích cực, tiêu cực, trung lập) . Mô hình học cách liên kết các đặc điểm của văn bản (như các từ trong văn bản, thứ tự các từ,...) với cảm xúc. Khi được cung cấp văn bản mới, không được gán nhãn, nó có thể dự đoán cảm xúc dựa trên bộ dữ liệu đã học được này. Mô hình học máy thường được sử dụng ở đây có thể kể đến là: Naive Bayes, Decision Tree,... hoặc mô hình mạng thần kinh phức tạp hơn như RNN, CNN, cũng có thể là BERT hay GPT.

2.2.3 Hybrid method

Đây là phương pháp kết hợp của 2 phương pháp Rule-based và Automatic. Ta có thể dùng Rule-based method để tạo ra các features sau đó đưa chúng vào mô hình học máy hoặc sử dụng mô hình học máy để đưa ra dự đoán về cảm xúc của đoạn văn bản sau đó được tinh chỉnh bằng phương pháp Rule-based.

Ý tưởng chính là tận dụng cả 2 phương pháp học máy và Rule-based để học các mẫu phức tạp trong dữ liệu.

3 Phương pháp đề xuất

Trong báo cáo, nhóm sử dụng phương pháp Automatic để thực hiện việc phân tích cảm xúc trên 2 nền tảng là Twitter và Reddit trong cuộc bầu cử tổng thống Ấn Độ đối với ứng cử viên Lok Sabha vào năm 2019 bằng cách sử dụng các thuật toán học máy thông qua PySpark. Thông thường, khi chạy các thuật toán học máy, nó liên quan đến một chuỗi các tác vụ bao gồm các giai đoạn tiền xử lý, trích xuất tính năng, lắp mô hình và xác nhận. Ví dụ: khi phân loại tài liệu văn bản có thể liên quan đến việc phân đoạn và làm sạch văn bản, trích xuất các tính năng và đào tạo một mô hình phân loại với xác thực chéo. Mặc dù có nhiều thư viện mà ta có thể sử dụng tuy nhiên việc kết hợp nó lại thì không dễ đặc biệt là với các bộ dữ liệu lớn.

Sau đây,báo cáo sẽ trình bày cụ thể các bước thực hiện của phương pháp

3.1 Dataset

Nhóm sử dụng 2 bộ dữ liệu từ 2 nền tảng mạng xã hội là Twitter và Reddit với bộ dữ liệu trên Twitter chứa 100K mẫu và bộ dữ liệu trên Reddit chứa 69561 mẫu. 2 bộ dữ liệu được mô tả cụ thể như sau:

a, Bộ dữ liệu trên Twitter:

Bộ dữ liệu mô tả gồm 100K mẫu với 5 đặc trưng lần lượt là:

- date: kiểu dữ liệu DateTime, mô tả thời gian người dùng bình luận.
- user: kiểu dữ liệu String, tên người dùng.
- text: kiểu dữ liệu String, chứa các bình luận trên twitter trong thời gian ‘date’.
- likes: kiểu dữ liệu int, mô tả số lượng yêu thích của bình luận.
- retweets: kiểu dữ liệu int, mô tả số lượng chia sẻ bình luận.

b, Bộ dữ liệu trên Reddit:

Bộ dữ liệu gồm gồm 69561 mẫu với 6 đặc trưng lần lượt là:

- Sub ID: kiểu dữ liệu String, ID của subreddit

- Comment ID: kiểu dữ liệu String, ID của từng bình luận
- Comment: kiểu dữ liệu String, chứa các bình luận trên Reddit trong thời gian 'Date'.
- score: kiểu dữ liệu int, mô tả số lượng yêu thích của bài viết.
- subreddit: Kiểu dữ liệu String, tên subreddit
- Date: kiểu dữ liệu DateTime, mô tả thời gian bình luận bài viết.

3.2 Tiền xử lý dữ liệu

Tiền xử lý dữ liệu là quá trình vô cùng quan trọng. Các đoạn văn bản trực tuyến thường chứa nhiều nội dung "nhiễu" và không mang thông tin, ví dụ như các ký tự đặc biệt, đường link, hashtag, các phần tử HTML,... Ngoài ra, ở cấp độ từ, nhiều từ xuất hiện trong văn bản không ảnh hưởng đến ý nghĩa tổng quát của nó. Việc giữ lại những từ này làm tăng "chiều" của vấn đề, khiến việc phân loại trở nên khó khăn và phức tạp hơn vì mỗi từ trong văn bản được xét như một "chiều" riêng biệt. Do đó, quá trình này "làm sạch" dữ liệu, chuẩn bị các đoạn văn bản để phân loại.

3.2.1 Tiền xử lý dữ liệu Twitter

```
col_names = ['date', 'user', 'text', 'likes', 'retweets']
df = pd.read_csv('modi_data100k.csv', names = col_names)
df.head()

df = df.drop_duplicates('text')
df.shape
```

Biểu diễn tập tin CSV chứa dữ liệu từ Twitter vào một DataFrame của pandas. Thư viện pandas cho phép sử dụng nhiều hàm để áp dụng lên DataFrame của pandas. Do đó, việc biểu diễn dưới dạng DataFrame của pandas là hiệu quả cho việc tiền xử lý dữ liệu. Các tweet trùng lặp được loại bỏ bằng cách sử dụng hàm *drop_duplicates* của thư viện pandas.

```
df['date'] = pd.to_datetime(df['date'])
df = df.sort_values(by = 'date', ascending = True)
```

```
df = df.reset_index().drop('index', axis = 1)
df.head()
```

Sắp xếp các DataFrame theo thứ tự tăng dần của cột 'date'.

3.2.2 Tiền xử lý dữ liệu Reddit

```
col_names=["Sub ID", "Comment ID", 'Comment', 'score', 'subreddit', 'Date']
df = pd.read_csv('reddit_comment.csv', names=col_names)
df.head()
df = df.drop_duplicates('Comment')
df.shape

df['Date'] = pd.to_datetime(df['Date'])
df = df.sort_values(by='Date',ascending=True)
df = df.reset_index().drop('index',axis=1)
df.head()
```

Quá trình tiền xử lý dữ liệu Reddit tương tự với Twitter vì cả hai đều là tập tin CSV.

3.2.3 Quá trình "làm sạch"

```
def processTweet(tweet):
    # Loại bỏ ký tự HTML đặc biệt (ví dụ: &amp;)
    tweet = re.sub(r'\&\w*;', '', tweet)

    # Chuyển đổi @username thành chuỗi 'AT_USER'.
    tweet = re.sub('@[^\s]+','',tweet)

    # Loại bỏ các ticker (ví dụ: các ký tự bắt đầu bằng ký hiệu $)
    tweet = re.sub(r'\$\w*', '', tweet)

    # Chuyển toàn bộ văn bản thành chữ thường
    tweet = tweet.lower()

    # Loại bỏ các đường link
    tweet = re.sub(r'https?:\/\/.*\/\w*', '', tweet)

    # Loại bỏ các hashtag
    tweet = re.sub(r'#\w*', '', tweet)

    # Loại bỏ dấu câu và phân tách các từ như 's, 't, 've để phân loại
    tweet = re.sub(r'[' + punctuation.replace('@', '') + ']+', '', tweet)

    # Loại bỏ các từ có độ dài ít hơn hoặc bằng 2 ký tự
    tweet = re.sub(r'\b\w{1,2}\b', '', tweet)

    # Loại bỏ khoảng trắng dư thừa (bao gồm cả ký tự xuống dòng mới)
    tweet = re.sub(r'\s\s+', ' ', tweet)

    # Loại bỏ khoảng trắng đầu câu
    tweet = tweet.lstrip(' ')

    # Loại bỏ các ký tự nằm ngoài Basic Multilingual Plane (BMP) của
    ↪ Unicode.
    tweet = ''.join(c for c in tweet if c <= '\uffff')
    return tweet
```

```
    df['clean_text'] = df['text'].apply(processTweet)#Twitter
    df['clean_comment'] = df['Comment'].apply(processTweet)#Reddit
    df.head()
```

Hàm trên được sử dụng để "làm sạch" các tweet từ bộ dữ liệu Twitter và các bình luận từ bộ dữ liệu Reddit. Thư viện Python 're' được sử dụng để tìm các mẫu trong văn bản và loại bỏ chúng, trả về văn bản đã được "làm sạch" như một cột mới trong DataFrame.

3.3 Gán nhãn cảm xúc cho đoạn văn bản

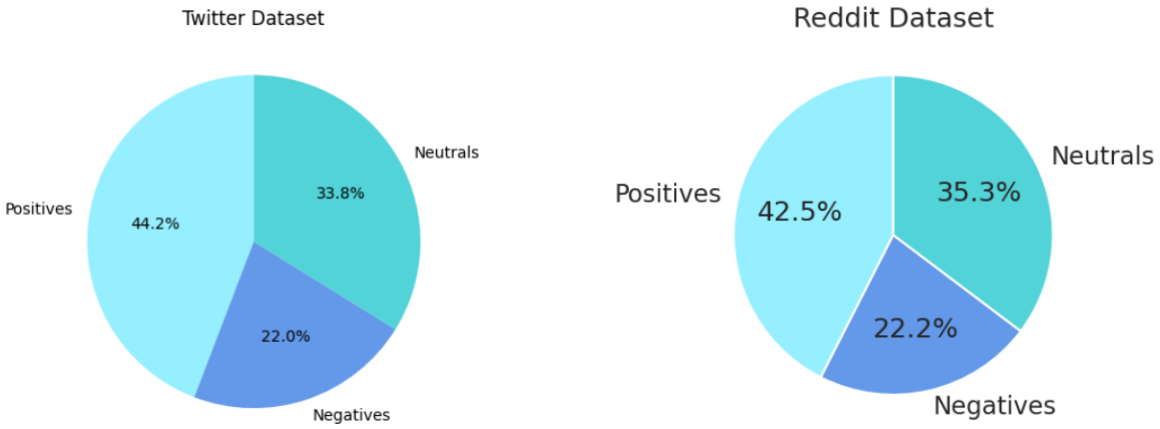
Trước khi phát triển bất kỳ mô hình nào để dự đoán cảm xúc, dữ liệu phải được gán nhãn, quá trình này được gọi là gán nhãn cảm xúc (sentiment labelling). Sau đó, bộ dữ liệu này có thể được chia thành tập train và tập test. Để gán nhãn cho dữ liệu, nhóm sử dụng thư viện TextBlob trong Python. TextBlob là một thư viện Python cho phép xử lý dữ liệu văn bản. Nó cung cấp một API đơn giản để đi sâu vào ngôn ngữ tự nhiên phổ biến: xử lý các tác vụ (NLP) như gắn thẻ một phần giọng nói, trích xuất cụm danh từ, phân tích cảm xúc, phân loại, dịch thuật và hơn thế nữa.

```
def analyze_sentiment(tweet):
    analysis = TextBlob(tweet)

    if analysis.sentiment.polarity > 0:
        return 1
    elif analysis.sentiment.polarity == 0:
        return 0
    else:
        return -1

df['category'] = df['clean_text'].apply(analyze_sentiment)
df.head()
```

Thuộc tính sentiment trả về một tuple có tên dưới dạng Sentiment (độ phân cực - polarity, tính chủ quan - subjectivity). Độ phân cực là một giá trị dạng số thực nằm trong khoảng [-1.0 đến 1.0], trong đó 0 biểu thị trạng thái trung lập, +1 biểu thị một cảm xúc rất tích cực và -1 biểu thị một cảm xúc rất tiêu cực. Tính chủ quan là một giá trị dạng số thực nằm trong khoảng [0.0 đến 1.0], trong đó 0 đại diện cho tính chất khách quan và 1 đại diện cho tính chất chủ quan cao. Câu chủ quan thể hiện một số cảm xúc cá nhân, quan điểm, niềm tin và suy đoán trong khi câu khách quan là các câu mô tả các sự thật, thông tin thực tế.

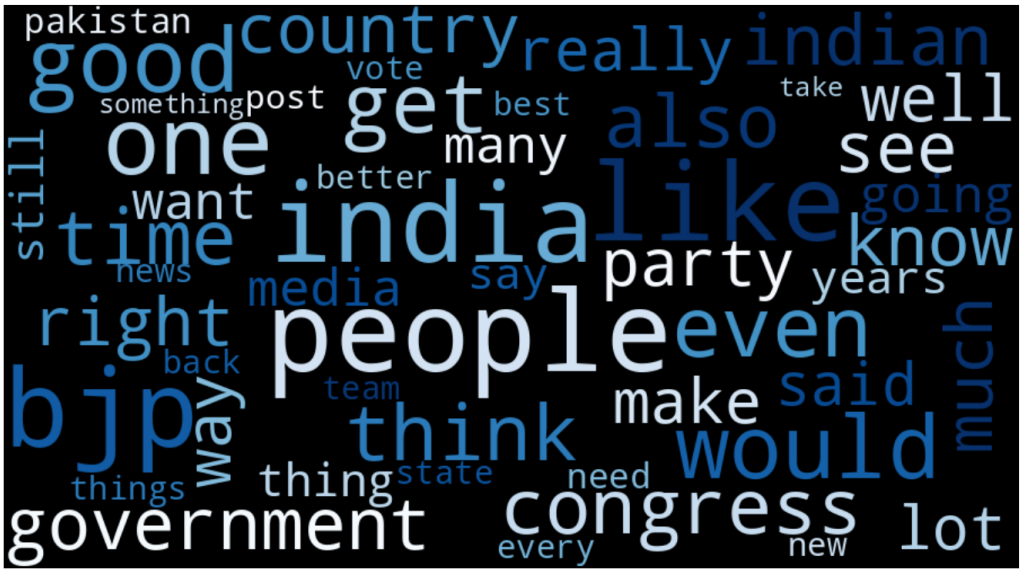
TextBlob kiểm tra, gán giá trị độ phân cực và tính chủ quan cho từng từ hoặc cụm từ mà nó có thể tìm được, sau đó nó tính trung bình cho tất cả chúng khi xử lý văn bản dài hơn. Dữ liệu được gán nhãn dựa trên giá trị độ phân cực. Tất cả các giá trị độ phân cực lớn hơn 1 (với nhãn +1) được phân loại vào cảm xúc tích cực, các giá trị nhỏ hơn 0 (với nhãn -1) được phân loại vào cảm xúc tiêu cực và các giá trị độ phân cực bằng 0 được trả về như cảm xúc trung lập.

3.4 Trực quan hóa dữ liệu



Hình 1: Tổng quan 2 bộ dữ liệu sau khi gán nhãn

Hình 2: WordCloud của Twitter Dataset



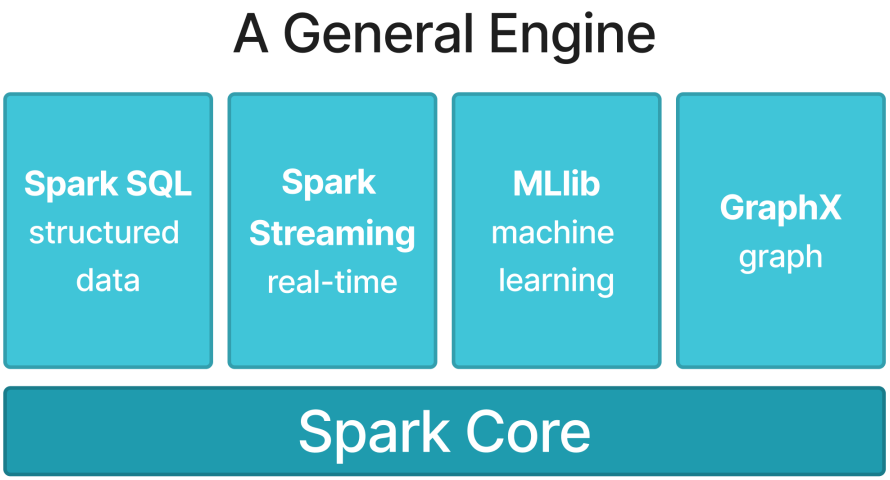
Hình 3: WordCloud của Reddit Dataset

3.5 Pipeline

3.5.1 Apache Spark

a, Apache Spark là gì? [2]

Apache Spark là một hệ thống xử lý phân tán mã nguồn mở được sử dụng cho các khối lượng công việc dữ liệu lớn. Nó cung cấp các API phát triển bằng ngôn ngữ Java, Scala, Python và R và hỗ trợ tái sử dụng mã trên nhiều khối lượng công việc, chẳng hạn như xử lý lô dữ liệu, truy vấn tương tác, phân tích theo thời gian thực, học máy và xử lý đồ thị.



Hình 4: Apache Spark

- Spark Core: đảm nhận vai trò thực hiện công việc tính toán và xử lý bộ

nhớ trong, đồng thời tham chiếu các dữ liệu được lưu trữ tại các hệ thống lưu trữ bên ngoài

- Spark SQL: cung cấp một kiểu data abstraction mới (SchemaRDD) nhằm hỗ trợ cho cả kiểu dữ liệu có cấu trúc (structured data) và dữ liệu nửa cấu trúc (semi-structured data)
- Spark Streaming: thực hiện việc phân tích stream bằng việc coi stream là các mini-batches và thực hiệc kỹ thuật RDD-transformation đối với các dữ liệu mini-batches này
- MLlib: nền tảng học máy phân tán bên trên Spark do kiến trúc phân tán dựa trên bộ nhớ
- GrapX: nền tảng xử lý đồ thị dựa trên Spark, cung cấp các Api để diễn tảcác tính toán trong đồ thị

b, PySpark là gì?

PySpark là giao diện cho Apache Spark bằng Python. Nó cho phép viết các ứng dụng Spark bằng API Python và cũng cung cấp PySpark để phân tích tương tác dữ liệu trong môi trường phân tán.

PySpark hỗ trợ hầu hết các tính năng của Spark.

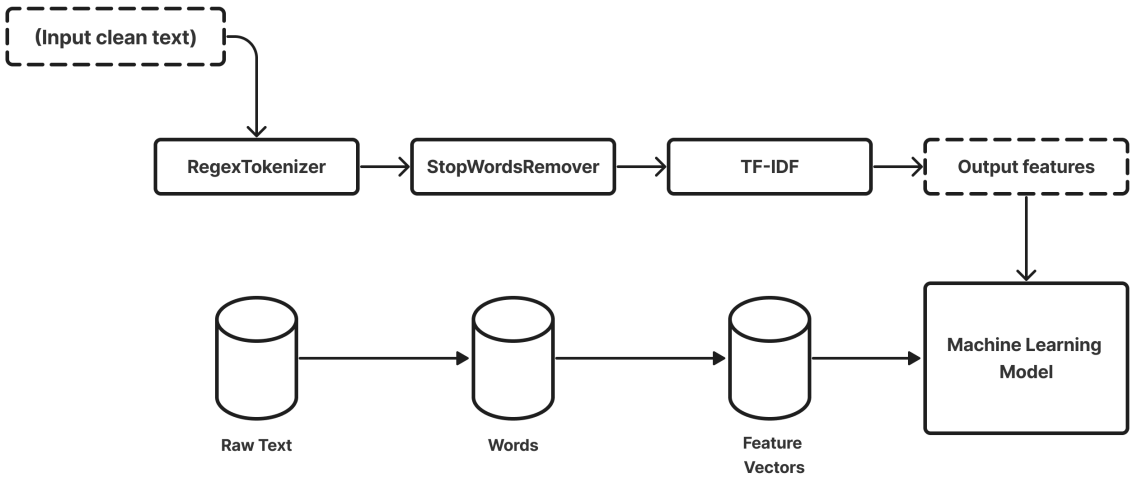
3.5.2 Các thành phần cơ bản của pipeline

[3] Pyspark cho phép xây dựng các mô hình end-to-end với dữ liệu truyền vào là các dữ liệu thô và kết quả trả ra là nhãn, xác xuất hoặc giá trị được dự báo của mô hình.

- Transformers: bao gồm các feature transformers và các mô hình đã học. Mỗi transformers có một phương thức $transform()$ nhận đầu vào là một DataFrame và trả ra một DataFrame mới có các trường đã biến đổi theo transform bằng cách thêm, xóa hoặc thay đổi giá trị.
- Estimators: được dùng để ước lượng mô hình dự báo. Mỗi Estimators có một phương thức $fit()$ để huấn luyện mô hình. Chúng cũng nhận đầu vào là một DataFrame nhưng kết quả trả về ở đầu ra là một mô hình, mô hình này có phương thức $transform()$. Hiện tại Spark hỗ trợ khá nhiều các lớp model cơ bản trong học máy, điển hình như: LogisticRegression, DecisionTreeClassifier, RandomForestModel, GBTClassifier (gradient bosting tree), MultilayerPerceptronClassifier, LinearSVC (Linear Support Vector Machine), NaiveBayes.

Trong PySpark, MLlib là thư viện chứa các thuật toán học máy giúp kết hợp dễ dàng hơn nhiều thuật toán vào một quy trình làm việc duy nhất. Các mô hình trong thư viện MLlib đại diện cho một quy trình làm việc như một Pipeline, bao gồm một chuỗi các giai đoạn $Transformer$ và $Estimators$ được chạy theo một thứ tự cụ thể.

3.6 Workflow



Hình 5: Flow Diagram

Hình 2 diễn giải quá trình làm việc để giải quyết bài toán phân loại đa lớp sử dụng các mô hình học máy có giám sát trên một pipeline theo thứ tự nhất định, được chia thành 2 giai đoạn chính:

- Transformers(): $regexTokenizer$, $StopWordsRemover$, $Tf-idf$
- Estimators(): $MachineLearningModel$

Input clean data

Dữ liệu đầu vào đã được làm "sạch" và gán nhãn thông qua quá trình tiền xử lý. Dữ liệu được đọc vào thông qua class $StructType$ và $StructField$ của thư viện $pyspark.sql.types$ [4] để tạo Dataframe.

regexTokenizer

Là 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. Đối với bài toán của nhóm, một bình luận đầu vào sẽ được tách ra thành các từ riêng lẻ.

StopWordsRemover

Là quá trình lược bớt những từ ngữ không cần thiết trong quá trình phân loại. Ở đây, nhóm sử dụng tập dữ liệu $stopwords$ từ thư viện $nltk$

Tf-idf

Là quá trình đo trọng số thể hiện sự quan trọng của các từ trong tập dữ liệu. Bước này trả về một vector thể hiện đặc trưng cho từng dữ liệu đầu vào.

Machine Learning Model

Là quá trình sử dụng các mô hình học máy có trong thư viện $spark.ml$ để giải quyết bài toán. Ở phần tiếp theo, nhóm sẽ đưa ra thuật toán chính mà nhóm sử dụng cũng như một số thuật toán khác trong thư viện. Từ đó đưa ra nhận xét và đánh giá cho các mô hình này.

3.7 Machine Learning Models

3.7.1 Một số thuật toán học máy

A. Naive Bayes

a, Định nghĩa

Naive Bayes là một thuật toán phân loại đa lớp dựa trên định lý Bayes và thuật toán này thuộc nhóm học có giám sát. Các đặc trưng khi đưa vào mô hình là độc lập nhau (sự thay đổi giá trị của một đặc trưng không ảnh hưởng đến các đặc trưng còn lại).

Theo định lý Bayes, ta có công thức sau:

$$P(y|X) = \frac{P(X|y)P(y)}{P(X)}$$

Trong đó:

$$\begin{cases} P(y|X): & \text{posterior probability - xác suất của mục tiêu y khi có đặc trưng X.} \\ P(X|y): & \text{likelihood - xác suất của đặc trưng X khi đã biết mục tiêu y.} \\ P(y): & \text{prior probability - xác suất của mục tiêu y.} \\ P(X): & \text{prior probability - xác suất của đặc trưng X.} \end{cases} \tag{1}$$

b, Mô hình thuật toán

Một trong những bài toán ứng dụng nổi tiếng hiệu quả khi sử dụng Naive

Bayes là bài toán phân loại văn bản. Trong bài toán này, mỗi văn bản được thể hiện thành dạng 'bag of words', 'bag of words' thể hiện số từ xuất hiện và tần suất xuất hiện của từ đó trong văn bản, nhưng bỏ qua thứ tự các từ.

Có 2 mô hình thuật toán Naive Bayes thường sử dụng là: mô hình Bernoulli và mô hình Multinomial. Trong báo cáo, sau khi chia bộ dữ liệu thành 2 tập là tập train và tập test, nhóm thực hiện tính toán trên 2 tập đó và mô hình Naive Bayes nhóm sử dụng trong bài toán phân tích cảm xúc là mô hình Multinomial, cụ thể như sau:

** Tập train*:

Đầu tiên ta tính xác suất xuất hiện của các mục tiêu y trong văn bản theo công thức:

$$P(y_i) = \frac{N_{y_i}}{N}$$

Trong đó:

$$\begin{cases} N_{y_i}: & \text{Số lượng nhãn mục tiêu } y_i. \\ N: & \text{Tổng số lượng nhãn mục tiêu có trong đoạn văn bản.} \end{cases} \tag{2}$$

Sau đó áp dụng định lý Bayes để tính toán các phân bố xác suất có điều kiện của các nhãn cho một quan sát và sử dụng nó cho dự đoán. Tuy nhiên có một hạn chế rằng khi từ nào đó trong tập test không xuất hiện trong tập train, xác suất trả về sẽ là 0. Để khắc phục vấn đề này, ta sử dụng kỹ thuật Laplace Smoothing bằng cách cộng thêm vào cả tử và mẫu những số khác 0. Khi đó công thức likelihood được xác định bởi:

$$P(w_i|y_i) = \frac{N_{(w_i|y_i)} + 1}{N_{w,y_i} + W}$$

Trong đó:

$$\begin{cases} P(w_i|y_i): & \text{Xác suất của từ } w_i \text{ khi đã biết nhãn } y_i \\ N_{(w_i|y_i)}: & \text{Số lượng các từ } w_i \text{ tương ứng với nhãn mục tiêu } y_i. \\ N_{w,y_i}: & \text{Tổng số lượng từ tương ứng với nhãn mục tiêu } y_i. \\ W: & \text{Tổng số từ không trùng nhau có trong đoạn văn bản.} \end{cases} \tag{3}$$

Cuối cùng khi đã thu được xác suất của đoạn văn tương ứng với từng nhãn theo công thức Bayes, xác suất của nhãn nào lớn nhất thì mô hình sẽ phân loại

đoạn văn tương ứng với nhãn đó.

** Tập test*:

Sau khi đã huấn luyện mô hình trên tập train, ta áp dụng mô hình đó vào tập test để phân loại cảm xúc cho đoạn văn.

Trên đây là cách hoạt động của thuật toán Naive Bayes. Và trong thư viện spark.mllib cũng hỗ trợ các hàm để sử dụng với Multinomial Naive Bayes. Các mô hình này thường được sử dụng để phân loại tài liệu. Trong đó, mỗi quan sát là một văn bản và mỗi đặc trưng đại diện cho một thuật ngữ có giá trị là tần số của thuật ngữ. Khi áp dụng mô hình vào tập train và test, ta chỉ cần dùng 2 hàm $fit()$ và $transform()$.

c, Sử dụng Naive Bayes với thư viện spark.mllib

```
from pyspark.ml.classification import NaiveBayes
nb = NaiveBayes(smoothing=1)
```

Trong đó:

- smoothing: Hệ số của phương pháp Laplace Smoothing để cộng thêm vào tử số khi tính xác suất likelihood.

Sau khi đã khai báo thông số smoothing cho mô hình, tiến hành huấn luyện cho mô hình trên tập train bằng lệnh:

```
model = nb.fit(trainingData)
```

Sau đó, chúng ta áp dụng mô hình đã huấn luyện được ở trên để tiến hành phân loại trên tập test và đưa ra dự đoán như sau:

```
predictions = model.transform(testData)
```

Cuối cùng, độ chính xác của mô hình được xác nhận bằng câu lệnh:

```
evaluator = MulticlassClassificationEvaluator(predictionCol="prediction")
evaluator.evaluate(predictions)
```

Để cải thiện độ chính xác của mô hình, nhóm sử dụng phương pháp xác thực chéo (Cross-Validation) để tối ưu hệ số smoothing của Naive Bayes:

```
# Create initial Naïve Bayes model
nb = NaiveBayes(labelCol="label", featuresCol="features")
```

Trong đó:

- labelCol: cột chứa nhãn của từng quan sát.
- featuresCol: cột chứa vector đặc trưng của từng quan sát.

Sau đó, nhóm sử dụng ParamGrid chỉ định các giá trị khác nhau cho siêu tham số smoothing để trả về giá trị siêu tham số tốt nhất cho mô hình:

```
# Create ParamGrid for Cross Validation
nbparamGrid = (ParamGridBuilder()
               .addGrid(nb.smoothing, [0.0, 0.2, 0.4, 0.6, 0.8, 1.0])
               .build())
```

Tiếp theo, ta sử dụng phương pháp K-fold để huấn luyện mô hình, cụ thể ở đây nhóm sử dụng K = 5.

```
# Create 5-fold CrossValidator
nbcv = CrossValidator(estimator = nb,
                      estimatorParamMaps = nbparamGrid,
                      evaluator = nbevaluator,
                      numFolds = 5)
```

Mô hình sẽ được huấn luyện trên tập train và áp dụng vào tập test như sau:

```
# Run cross validations
nbcvModel = nbcv.fit(trainingData)
# Use test set here so we can measure the accuracy of our model on new data
nbpredictions = nbcvModel.transform(testData)
```

d, Đánh giá

Ta có thể thấy rằng Naive Bayes là mô hình phân lớp đơn giản và dễ cài đặt, có tốc độ xử lý nhanh. Tuy nhiên nhược điểm lớn nhất của Naive Bayes là yêu cầu các đặc trưng đầu vào phải độc lập, mà điều này khó xảy ra trong thực tế nên nó dễ làm giảm chất lượng của mô hình. Và độ chính xác mà thuật toán Naive Bayes đạt được trong bài toán này được thống kê như sau:

STT	Bộ dữ liệu	Độ chính xác
1	Twitter	0.6696806624304865
2	Reddit	0.599204392648857
3	Kết hợp Twitter, Reddit	0.6856122072370481

B.Random Forest

a, Định nghĩa

Cây quyết định (Decision Tree)

Cây quyết định là một mô hình trong học máy được sử dụng cho cả bài toán phân loại và hồi quy. Cây quyết định bao gồm các nút (nodes) và cạnh (edges), trong đó mỗi nút đại diện cho một thuộc tính hoặc một điều kiện, và mỗi cạnh biểu thị một quyết định hoặc phân nhánh dựa trên thuộc tính đó. Quá trình xây dựng cây quyết định là tìm ra các điều kiện tốt nhất để phân chia dữ liệu dựa trên các đặc trưng, nhằm tạo ra các nhánh con cho đến khi đạt được các điều kiện dừng đã được xác định trước.

Cây quyết định có thể dễ dàng hiểu và diễn giải, đặc biệt là khi có một số lượng nhỏ các thuộc tính hoặc đặc trưng. Tuy nhiên, chúng có thể dễ bị "overfitting" nếu không kiểm soát được và có thể không hiệu quả trong việc xử lý các tương tác phức tạp giữa các đặc trưng.

Rừng ngẫu nhiên (Random Forest)

Rừng ngẫu nhiên là sự mở rộng của cây quyết định, kết hợp nhiều cây quyết định để giảm thiểu overfitting và tăng tính đa dạng của mô hình. Tương tự như cây quyết định, rừng ngẫu nhiên xử lý các đặc trưng phân loại, mở rộng sang việc phân loại đa lớp, không yêu cầu tỷ lệ đặc trưng, có khả năng bắt được tính phi tuyến và tương tác giữa các đặc trưng.

spark.mllib hỗ trợ mô hình rừng ngẫu nhiên cho việc phân loại nhị phân và đa lớp cũng như cho hồi quy, sử dụng cả đặc trưng liên tục và phân loại; thực hiện rừng ngẫu nhiên bằng cách khai cây quyết định hiện có.

Mô hình rừng ngẫu nhiên là một loại mô hình cộng hưởng (additive model) dự đoán bằng cách kết hợp các quyết định từ một chuỗi các mô hình cơ sở. Cụ thể hơn, chúng ta có thể viết lớp mô hình này như sau:

$$g(x) = f_0(x) + f_1(x) + f_2(x) + \ldots$$

Trong đó, $g(x)$ là tổng của các mô hình cơ bản đơn giản f_i. Ở đây, mỗi bộ phân loại cơ bản là một cây quyết định đơn giản. Kỹ thuật này sử dụng nhiều mô hình để đạt được hiệu suất dự đoán tốt hơn, gọi là "tập hợp mô hình" (model ensemble). Trong rừng ngẫu nhiên, tất cả các mô hình cơ bản được xây dựng độc lập bằng cách sử dụng một mẫu con dữ liệu khác nhau.

b, Mô hình thuật toán

Trong Python, sử dụng thư viện pyspark.ml để triển khai thuật toán Rừng ngẫu nhiên.

Cách hoạt động cơ bản của thuật toán Rừng ngẫu nhiên

Rừng ngẫu nhiên huấn luyện một tập hợp các cây quyết định một cách riêng biệt, cho phép việc huấn luyện được thực hiện song song. Thuật toán đưa tính ngẫu nhiên vào quá trình huấn luyện sao cho mỗi cây quyết định có chút khác biệt. Việc kết hợp các dự đoán từ mỗi cây giảm phương sai của các dự đoán, cải thiện hiệu suất trên dữ liệu test.

Huấn luyện

Tính ngẫu nhiên được đưa vào quá trình huấn luyện bao gồm:

- Lấy một phần con của bộ dữ liệu gốc ở mỗi vòng lặp để có các tập dữ liệu train khác nhau (còn được gọi là bootstrap).
- Xem xét các tập hợp con có tính năng ngẫu nhiên khác nhau để phân chia tại mỗi nút cây. Ngoài những sự ngẫu nhiên hóa này, việc huấn luyện cây quyết định được thực hiện tương tự như đối với các cây quyết định riêng lẻ.

Dự đoán

Để đưa ra dự đoán về một trường hợp mới, rừng ngẫu nhiên phải tổng hợp các dự đoán từ tập hợp cây quyết định của nó. Việc tổng hợp này được thực hiện khác nhau để phân loại và hồi quy.

Phân loại

Dự đoán của mỗi cây được tính là một phiếu bầu cho một lớp. Nhãn được dự đoán là lớp nhận được số phiếu bầu nhiều nhất.

Hồi quy

Mỗi cây dự đoán một giá trị thực. Nhãn được dự đoán là trung bình của các dự đoán từ các cây.

c, Hồi quy Rừng ngẫu nhiên với thư viện spark.mllib

```
from pyspark.ml.classification import RandomForestClassifier
rf = RandomForestClassifier(featuresCol="features", \
                            labelCol="label", \
                            numTrees = 100, \
                            maxDepth = 4, \
                            maxBins = 32)
```

Trong đó:

- featuresCol: cột chứa vector đặc trưng của từng quan sát
- labelCol: cột chứa nhãn của từng quan sát
- numTrees: số lượng cây trong mô hình RandomForest
- maxDepth: độ sâu tối đa cho mỗi cây trong RandomForest
- maxBins: số lượng bin tối đa được sử dụng khi tạo cây

Sau khi đã khai báo các thông số phù hợp cho mô hình, chúng ta tiến hành huấn luyện cho mô hình trên tập train

```
rfModel = rf.fit(trainingData)
```

Sau đó, chúng ta áp dụng mô hình đã huấn luyện để tiến hành phân loại trên tập test và đưa ra dự đoán

```
predictions = rfModel.transform(testData)
```

c, Đánh giá

Tiến hành đánh giá độ chính xác của mô hình:

STT	Bộ dữ liệu	Độ chính xác
1	Twitter	0.2711243922492973
2	Reddit	0.2510421338528298
3	Kết hợp Twitter, Reddit	0.2672369385574591

3.7.2 Thuật toán chính - Multinomial Logistic Regression

3.5.1.1 Mô hình thuật toán

a, Định nghĩa

Hồi quy Logistic(Logistic Regression) là một thuật toán hồi quy nhưng thường được sử dụng trong bài toán phân loại và thuộc nhóm học máy có giám sát (Supervised Learning). Thuật toán trả về giá trị biểu thị xác suất cho

các quan sát đầu vào đối với từng lớp rời rạc, từ đó đưa ra dự đoán cho các phân loại này.

Thư viện spark.ml của Spark hỗ trợ hai loại hồi quy Logistic:

- Hồi quy Logistic nhị thức (Binominal Logistic Regression): xem xét mối quan hệ giữa biến phụ thuộc là biến nhị phân (ví dụ: tiêu cực/tích cực) và biến độc lập có thể là biến định lượng hoặc biến định tính.
- Hồi quy Logistic đa thức (Multinomial Logistic Regression):) tương tự như mô hình hồi qui Logistic nhị thức nhưng biến phụ thuộc là biến định tính có nhiều hơn 2 trạng thái (ví dụ: tiêu cực/trung lập/tích cực).

Bài toán của chúng ta là phân tích cảm xúc với 3 nhãn là tiêu cực, trung lập và tích cực, vì vậy thuật toán được áp dụng là hồi quy Logistic đa thức cho bài toán phân loại.

b, Softmax Activation Function

Thuật toán Hồi quy Logistic đa thức trả về ma trận 1 chiều chứa các chỉ số xác suất của từng quan sát đối với từng lớp k_i thông qua hàm softmax

$$h_{\mathbf{w}}(\mathbf{x}) = \begin{bmatrix} P(y = 1|\mathbf{x}; w) \\ P(y = 2|\mathbf{x}; w) \\ \cdots \\ P(y = K|\mathbf{x}; w) \end{bmatrix}$$

Hàm softmax được dùng để tính toán xác suất xảy ra để một điểm dữ liệu cho trước thuộc về một lớp k nào đó. Cụ thể, hàm softmax sẽ nhận một vector k chiều bao gồm các giá trị thực thành một vector k chiều mới chứa các giá trị thực nằm trong khoảng [0, 1] và luôn có tổng bằng 1.

Hàm softmax được biểu diễn theo công thức:

$$\mathbf{softmax}(v) = \frac{1}{\sum_{k=1}^{K} e^{z_k}} \begin{bmatrix} e^{z_1} \\ e^{z_2} \\ \vdots \\ e^{z_K} \end{bmatrix}$$

$$với\ z_k = \boldsymbol{w}_k^T \boldsymbol{x}$$

Biểu thức $h_w(x)$ có thể viết lại thành:

$$h_{\mathbf{w}}(\mathbf{x}) = \begin{bmatrix} P(y=1|\mathbf{x};w) \\ P(y=2|\mathbf{x};w) \\ \cdots \\ P(y=K|\mathbf{x};w) \end{bmatrix} = \frac{1}{\sum_{k=1}^{K} e^{\mathbf{w}_k^T \mathbf{x}}} \begin{bmatrix} e^{\mathbf{w}_1^T \mathbf{x}} \\ e^{\mathbf{w}_2^T \mathbf{x}} \\ \cdots \\ e^{\mathbf{w}_K^T \mathbf{x}} \end{bmatrix}$$

c, Cross-entropy Loss Function

Để tối ưu hóa mô hình, chúng ta cần giảm thiểu được hàm mất mát - hàm được dùng để tính toán sự sai khác giữa kết quả dự đoán và giá trị thực tế của tập dữ liệu. Đối với bài toán phân loại đa lớp sử dụng hồi quy Logistic, kết quả trả về của dự đoán và thực tế đều là các phân phối xác suất. Vì vậy, để tính toán hàm mất mát cho mô hình, chúng ta sử dụng hàm Cross-entropy, được cho bởi công thức:

$$H(\boldsymbol{P},\boldsymbol{Q}) = -\sum_{i=1}^{K} p_i \log(q_i)$$

$$\begin{cases} P = (\mathrm{p}_1,\ p_2,...,p_K): & \text{giá trị thực tế cho dưới dạng one-hot Vector} \\ Q = (\mathrm{q}_1,\ q_2,...,q_K): & \text{giá trị dự đoán được tính toán bằng softmax} \end{cases} \tag{4}$$

Từ đó, ta có công thức của hàm mất mát:

$$\text{loss}(\boldsymbol{Y},\boldsymbol{X};\mathbf{w}) = -\sum_{i=1}^{N}\sum_{k=1}^{K} I[y_i = k] \log\left(\frac{e^{\mathbf{w}_k^T \mathbf{x}_i}}{\sum_{k=1}^{K} e^{\mathbf{w}_k^T \mathbf{x}_i}}\right)$$

d, Elastic Net Regularization

Trong học máy, overfitting là hiện tượng khi mô hình quá khít với dữ liệu, dẫn đến việc đem lại kết quả tốt trên tập train nhưng với tập test thì ngược lại. Để tránh hiện tượng này xảy ra, chúng ta cần phương pháp giúp thay đổi mô hình một chút trong khi vẫn giữ được tính tổng quát của nó, được gọi là Regularization. Một cách cụ thể hơn, ta sẽ tìm cách di chuyển nghiệm của bài toán tối ưu hàm mất mát tới một điểm gần nó. Hướng di chuyển sẽ là hướng làm cho mô hình ít phức tạp hơn mặc dù giá trị của hàm mất mát có tăng lên một chút.

Một kỹ thuật regularization phổ biến đó là thêm vào hàm mất mát một tham số nào đó. Tham số này thường dùng để đánh giá độ phức tạp của mô

hình, giá trị càng lớn, thì mô hình càng phức tạp.

Mô hình hồi quy Logistic trong Spark sử dụng kỹ thuật Elastic Net Regression, được cho bởi công thức:

$$\lambda \left[\frac{1}{2}(1-\alpha)\|\beta\|_2^2 + \alpha\|\beta\|_1 \right]$$

Trong đó:

$$\begin{cases} \lambda: & \text{tham số regularization} \\ \alpha: & \text{trọng số cho chuẩn L1 và L2} \\ \beta: & \text{vector hệ số của mô hình} \end{cases} \tag{5}$$

3.5.1.2 Hồi quy Logistic với thư viện spark.mllib [5]

```
from pyspark.ml.classification import LogisticRegression

logistic_regression = LogisticRegression(
                        featuresCol = 'features',
                        labelCol = 'label',
                        family = 'multinomial',
                        maxIter = 20,
                        regParam = 0.3,
                        elasticNetParam = 0)
```

Trong đó:

- featuresCol: cột chứa vector đặc trưng của từng quan sát
- labelCol: cột chứa nhãn của từng quan sát
- family: xác định kiểu hồi quy Logistic (auto/binomial/multinomial)
- maxIter: số vòng lặp
- regParam: tham số regularization
- elasticNetParam: trọng số α cho chuẩn L1 và L2

Sau khi đã khai báo các thông số phù hợp cho mô hình, chúng ta tiến hành huấn luyện cho mô hình trên tập train

```
lrModel = logistic_regression.fit(trainingData)
```

Sau đó, chúng ta áp dụng mô hình đã huấn luyện để tiến hành phân loại trên tập test và đưa ra dự đoán

```
predictions = lrModel.transform(testData)
```

```
+-----------------------------------------------------------+-----------------------------------------------------------+----------+
|rawPrediction                                              |probability                                                |prediction|
+-----------------------------------------------------------+-----------------------------------------------------------+----------+
|[0.07771163316623936,0.651264258474753,-0.7289758916409924]   |[0.3104727772619628,0.5509526379715356,0.13857458476650172]  |1.0       |
|[-0.06243799760335658,0.4296136744027645,-0.36717567679940843]|[0.2964732668681396,0.48493202382155715,0.21859470931030342] |1.0       |
|[1.136288026992876,0.07354616633267475,-1.2098341933255512]   |[0.6938436535795051,0.23972796263578877,0.0664283837847062]  |0.0       |
|[0.9830347643272623,0.03206941812880182,-1.0151041824560645]  |[0.6570495426607753,0.2538628287476741,0.08908762859155069]  |0.0       |
|[-0.02774965318588185,0.7196831614204332,-0.6919335082345514] |[0.27576578479928054,0.5822993800901253,0.14193483511059432] |1.0       |
|[0.954472635146401,-0.3313478951983585,-0.6231247399480427]   |[0.6743569203112979,0.18640821921130338,0.1392348604773987]  |0.0       |
|[0.3130683195066573 7,0.3176628052644272,-0.6307311247710852]  |[0.417754300572039,0.4196780827670724,0.1625676166608885]    |1.0       |
|[0.08722164767701245,0.6657745203796082,-0.7529961680566207]  |[0.3110350080685869,0.554717150270313 4,0.13424784166109965]  |1.0       |
|[-0.4744321334360954,-1.0878024641579591,1.562234597594054]   |[0.10861844233760705,0.058819349935608924,0.832562207726784] |2.0       |
|[-0.2575888901867106,0.22080484024811303,0.036784049938597296]|[0.25279560102621823,0.40788080681036826,0.33932359216341346]|1.0       |
+-----------------------------------------------------------+-----------------------------------------------------------+----------+
```

Hình 6: predictions.show()

Trong đó:

- rawPrediction: kết quả của công thức $z_k = \boldsymbol{w}_k^T \boldsymbol{x}$ với w là vector đặc trưng lấy từ cột $features$
- probability: kết quả sau khi áp dụng hàm $softmax$ vào $rawPrediction$
- prediction: kết quả dự đoán của quá trình phân loại

Để cải thiện độ chính xác của mô hình, nhóm sử dụng phương pháp xác thực chéo (Cross-Validation) để tối ưu các hệ số với Grid:

```
# Create ParamGrid for Cross Validation
paramGrid = (ParamGridBuilder()
             .addGrid(logistic_regression.regParam, [0.1, 0.3, 0.5]) #
              ↪ regularization parameter
             .addGrid(logistic_regression.elasticNetParam, [0.0, 0.1, 0.2]) #
              ↪ Elastic Net Parameter (Ridge = 0)
             .build())
```

Tiếp theo, ta sử dụng phương pháp K-fold để huấn luyện mô hình, cụ thể ở đây nhóm sử dụng K = 5

```
# Create 5-fold CrossValidator
cv = CrossValidator(estimator=logistic_regression, \
                    estimatorParamMaps=paramGrid, \
```

```
                    evaluator=evaluator, \
                    numFolds=5)
```

Cuối cùng, tiến hành kiểm thử trên tập train, khi đã huấn luyện được mô hình trên tập train ta sẽ áp dụng vào tập test như sau:

```
cvModel = cv.fit(trainingData)
predictions = cvModel.transform(testData)
```

3.5.1.3 Đánh giá

Tiến hành đánh giá độ chính xác của mô hình:

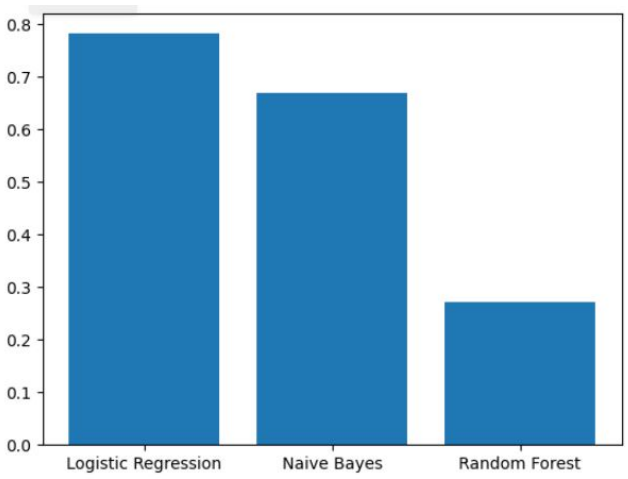
```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator

evaluator = MulticlassClassificationEvaluator(predictionCol="prediction")
evaluator.evaluate(predictions)
```

Sau khi áp dụng mô hình Logistic đã được huấn luyện vào 3 bộ dữ liệu, nhóm thu được kết quả như sau:

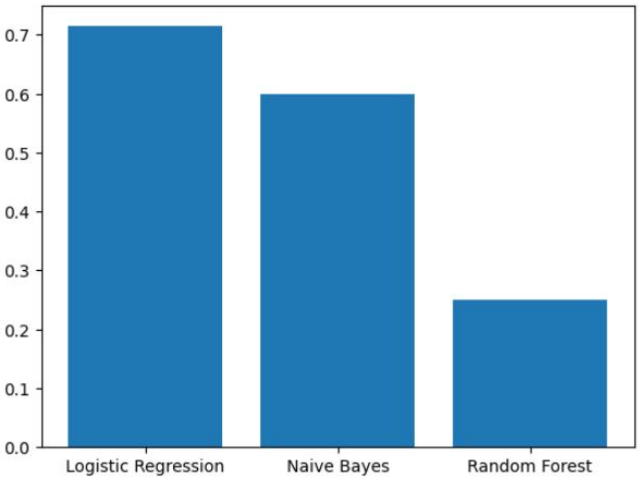
STT	Bộ dữ liệu	Độ chính xác
1	Twitter	0.7823004247616607
2	Reddit	0.7145820411995929
3	Kết hợp Twitter, Reddit	0.7906989316259325

4 Kết luận

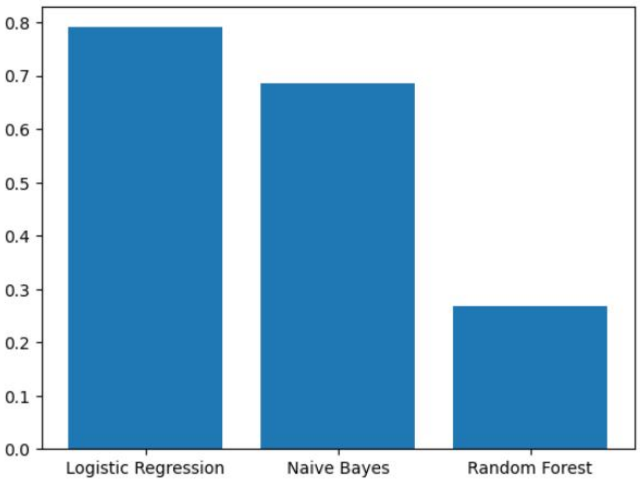
Sau khi áp dụng các mô hình học máy để giải quyết bài toán phân loại cảm xúc và có số liệu cụ thể, ta có thể thấy rằng Logistic là một phương pháp phù hợp để giải quyết bài toán này với độ chính xác có kết quả khá cao so với Random Forest và Naive Bayes, kết quả khi chạy các mô hình trên 3 bộ dữ liệu được trực quan qua 3 biểu đồ sau:



Hình 7: Độ chính xác của 3 thuật toán trên bộ dữ liệu Twitter



Hình 8: Độ chính xác của 3 thuật toán trên bộ dữ liệu Reddit



Hình 9: Độ chính xác của 3 thuật toán trên bộ dữ liệu kết hợp Twitter và Reddit

Có thể thấy, đối với cả ba tập dữ liệu đầu vào, mô hình hồi quy Logistic đều đem lại kết quả tốt hơn so với hai phương pháp còn lại là Naive Bayes và

Random Forest

Hồi quy Logistic là một mô hình phân loại đơn giản, dễ cài đặt với thời gian xử lý nhanh. Đặc biệt, mô hình có hiệu năng tốt nhất khi xử lý các tập dữ liệu nhỏ và có thể cải tiến đối với các tập dữ liệu lớn hơn thông qua một số kỹ thuật, ví dụ như Stochastic Gradient Descent. Tuy nhiên, một trong những nhược điểm lớn của mô hình đó là dễ gặp phải hiện tượng over-fitting khi số lượng đặc trưng vượt quá số lượng dữ liệu đầu vào.

5 Phương hướng phát triển

Nhóm đã thực hiện thành công việc phân tích cảm xúc cho 3 bộ dữ liệu với 3 phương pháp khác nhau sử dụng PySpark, các ưu điểm và nhược điểm của từng phương pháp cũng đã được nhóm trình bày ở từng phần trên. Tuy nhiên, trong tương lai nhóm mong rằng có thể lấy dữ liệu real-time từ hai nền tảng mạng xã hội trên kết hợp cùng Kafka để bài báo cáo của nhóm được hoàn thiện hơn.

Lời cảm ơn

Trong thời gian học tập và hoàn thiện bài báo cáo này, nhóm chúng em xin chân thành cảm ơn sự giúp đỡ nhiệt tình của thầy Phạm Tiến Lâm và thầy Đặng Văn Báu vì đã giúp đỡ, cung cấp nhiều thông tin quý báu cũng như tạo điều kiện cho bọn em có cơ hội được thực hành và học tập các khía cạnh liên quan đến PySpark để có thể hoàn thiện được bài báo cáo này trong suốt quá trình học tập.
Mặc dù nhóm đã cố gắng hoàn thiện bài báo cáo này một cách tốt nhất. Tuy nhiên do kiến thức, kinh nghiệm còn hạn chế nên bài báo cáo của nhóm vẫn chưa được trọn vẹn, còn có nhiều thiếu sót trong việc trình bày, đánh giá và đề xuất ý tưởng. Nhóm em rất mong nhận được những ý kiến đóng góp của thầy và các bạn để bài báo cáo trở nên tốt hơn.
Nhóm em xin chân thành cảm ơn!

Tài liệu

[1] URL: https://monkeylearn.com/sentiment-analysis/.

[2] URL: https://aws.amazon.com/vi/what-is/apache-spark/.

[3] URL: https://spark.apache.org/docs/latest/ml-pipeline.html#pipeline-components.

[4] URL: https://sparkbyexamples.com/pyspark/pyspark-structtype-and-structfield/.

[5] URL: https://spark.apache.org/docs/latest/ml-classification-regression.html.

[6] Alexander Pak and Patrick Paroubek. “Twitter as a Corpus for Sentiment Analysis and Opinion Mining”. In: vol. 10. Jan. 2010.