HỘI NGHỊ TOÀN QUỐC KHOA HỌC TRÁI ĐẤT
VÀ TÀI NGUYÊN VỚI PHÁT TRIỂN BỀN VỮNG (ERSD 2020)

Nghiên cứu mô hình học máy Naïve Bayes trong phân lớp văn bản; Ứng dụng phân lớp cho tập dữ liệu các nhận xét trên Twitter

Đặng Văn Nam[1,*]
[1] *Trường Đại học Mỏ - Địa chất*

TÓM TẮT

Trí tuệ nhân tạo (AI - Artificial Intelligence), Dữ liệu lớn (Big Data), Internet kết nối vạn vật (IoT – Internet of Things)…là những từ khóa của cuộc cách mạng công nghiệp 4. Các ứng dụng của trí tuệ nhân tạo nói chung và của Học máy (Machine Learning) nói riêng đã đem lại hiệu quả to lớn trong nhiều lĩnh vực của đời sống con người. Những năm gần đây, Học máy có vai trò quan trọng trong xử lý ngôn ngữ tự nhiên (NLP – Natural Language Processing). Một trong những bài toán rất quan trọng và phổ biến trong NLP đó là phân lớp văn bản (Text Classification). Có rất nhiều thuật toán học máy được sử dụng để giải quyết cho bài toán này. Tuy nhiên, Naïve Bayes là thuật toán có thời gian chạy nhanh và độ chính xác cao nên thường được sử dụng cho các bài toán phân lớp văn bản. Bài báo trình bày cụ thể việc xây dựng một mô hình học máy với thuật toán Naïve Bayes sử dụng đặc trưng TF-IDF (Term Frequency – Inverse Document Frequency) trong phân lớp văn bản. Tập dữ liệu thực nghiệm bao gồm 56 700 nhận xét (comments) trên Twitter và văn bản được phân thành 2 lớp bao gồm: lớp 0 - Non Toxic, lớp 1 – Toxic. Đồng thời, trong nội dung bài báo tác giả cũng thực hiện so sánh độ chính xác của Naïve Bayes với thuật toán SVN (Support Vector Machine) và KNN (K- Nearest Naighbors) trên cùng tập dữ liệu thực nghiệm.

Từ khóa: Học máy; Naïve Bayes; Xử lý ngôn ngữ tự nhiên; Phân lớp văn bản, TF-IDF

1. Mở đầu

Xử lý ngôn ngữ tự nhiên là một trong những lĩnh vực quan trọng của trí tuệ nhân tạo. Rất nhiều ứng dụng của NLP đã, đang và ngày càng có ảnh hưởng sâu rộng tới mọi mặt của đời sống con người. Có thể hiểu NLP là một lĩnh vực khoa học máy tính, kỹ thuật thông tin và trí tuệ nhân tạo tập trung vào nghiên cứu các tương tác về mặt ngôn ngữ giữa máy tính và con người, cụ thể hơn là làm thế nào để lập trình cho máy tính xử lý và phân tích một lượng lớn dữ liệu ngôn ngữ tự nhiên. Nói cách khác, NLP quan tâm tới việc làm thế nào để máy tính có thể hiểu và vận dụng được các tập dữ liệu sẵn có dưới dạng ngôn ngữ tự nhiên. Một số ứng dụng quan trọng của NLP có thể chỉ ra như: Các hệ thống máy dịch, ví dụ Google translation mà chúng ta đã quá quen thuộc; Ứng dụng trong xử lý văn bản và ngôn ngữ; Ứng dụng trong tóm tắt và phân loại văn bản, Hệ thống chatbot, Tổng đài tự động (ACC)….

Những năm gần đây, Học máy đang trở thành một phần không thể thiếu trong quá trình xử lý ngôn ngữ tự nhiên. Từ việc xây dựng các tập qui tắc bằng tay đòi hỏi rất nhiều thời gian, công sức, các nghiên cứu đang hướng đến việc sử dụng cơ sở dữ liệu lớn để tự động (hoặc bán tự động) sinh ra các quy tắc đó. Phương pháp này đã cho những kết quả khả quan trong nhiều lĩnh vực khác nhau của NLP.

Trong các ứng dụng của NLP, bài toán phân lớp văn bản là một trong những bài toán quan trọng và kinh điển nhất. Mục tiêu của một hệ thống phân lớp văn bản là nó có thể tự động phân lớp một văn bản cho trước, để xác định xem văn bản đó thuộc lớp nào. Các ứng dụng của phân lớp văn bản rất đa dạng như: Hiểu được ý nghĩa, đánh giá, bình luận của người dùng; Lọc email rác; Phân tích cảm xúc; Phân lớp tin tức, các bài báo điện tử…Bài toán phân lớp văn bản là một bài toán học có giám sát (supervised learning) trong học máy, tập văn bản huấn luyện đã được gán nhãn và được sử dụng để thực hiện phân lớp, việc thực hiện gán các nhãn lên một văn bản mới sẽ dựa trên trên mức độ tương tự của văn bản đó so với các văn bản đã được gán nhãn trong tập huấn luyện. Để giải quyết một bài toán phân lớp văn bản, thường trải qua 4 giai đoạn:

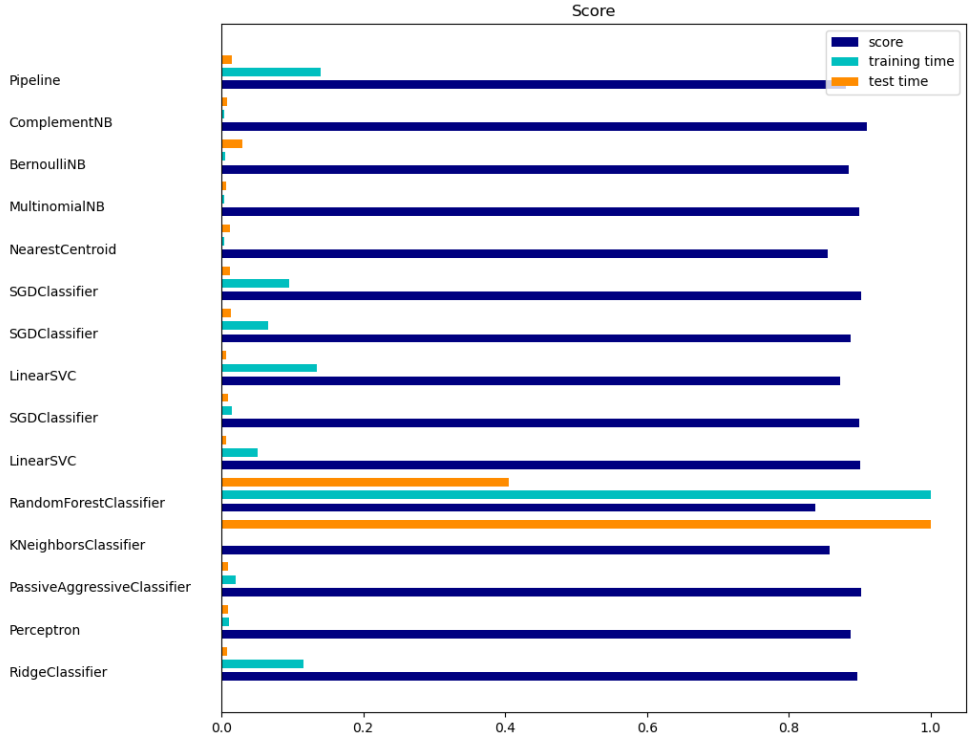
- Chuẩn bị dữ liệu (Data Preparation)
- Trích chọn đặc trưng (Feature Engineering)
- Xây dựng mô hình phân lớp (Buil Model)
- Tinh chỉnh mô hình và cải thiện hiệu năng (Improve Performance)

Như đã trình bày ở trên, Phân lớp văn bản là một bài toán học có giám sát, có rất nhiều thuật toán học

* *Tác giả liên hệ*
Email: dangvannam@humg.edu.vn

có giám sát như: Support Vector Machines, Decision Trees, Naïve Bayes, K-Nearest Neighbors, Random Forest…Tuy nhiên, Naïve Bayes Classifiers là một trong những thuật toán đem lại hiệu quả cho những bài toán phân lớp trong NLP nhờ độ chính xác cao, thời gian huấn luyện mô hình nhỏ.

Hình 1 là kết quả so sánh các thuật toán học máy khác nhau trong việc phân lớp văn bản trên cùng một tập dữ liệu bao gồm 18000 văn bản được gán nhãn vào 20 lớp, sử dụng cùng một phương pháp trích chọn đặc trưng và cùng một cấu hình máy tính như nhau. Các kết quả chỉ ra bao gồm: Độ chính xác (score) của thuật toán trên tập kiểm thử, thời gian thuật toán sử dụng để huấn luyện trên tập huấn luyện (Training time) và thời gian thuật toán sử dụng để kiểm thử mô hình trên tập kiểm thử (Test time). Kết quả cho thấy thuật toán Naïve Bayes với ba biến thể khác nhau (ComplementNB, BernoulliNB và MultinomialNB) đều cho Score cao và Training time, Test time nhỏ hơn các thuật toán khác.



Hình 1. So sánh các thuật toán học máy khác nhau trong phân lớp văn bản.

Nội dung của bài báo này tác giả sẽ nghiên cứu về thuật toán Naïve Bayes, và ứng dụng của Naïve Bayes trong bài toán phân lớp văn bản trên tập dữ liệu cụ thể là những nhận xét (comments) thu được từ mạng xã hội Twitter, các nhận xét được phân thành 2 lớp, gán nhãn 0 – Non Toxic (nhận xét không chứa các từ độc hại, không mang ý nghĩa tục tĩu, kích động….); gán nhãn 1 – Toxic (nhận xét có chứa những từ độc hại, mang ý nghĩa tục tĩu, kích động).

Như đã biết, độ chính xác của một mô hình học máy không những phụ thuộc vào thuật toán và thiết lập các tham số tương ứng của thuật toán đó, mà một trong những yếu tố rất quan trọng ảnh hưởng lớn tới độ chính xác đó là phương pháp trích chọn đặc trưng, chuyển đổi dữ liệu dạng văn bản đã được xử lý về dạng vector thuộc tính dạng số học. Có nhiều cách khác nhau để đưa dữ liệu dạng văn bản về dữ liệu dạng số như: Bag of words (BoW), TF-IDF, Word Embeddings, n-grams model, skip-Gram model…Bài báo này cũng trình bày về phương pháp trích chọn đặc trưng cơ bản được sử dụng rộng rãi là TF-IDF, và áp dụng kỹ thuật này cho tập dữ liệu thử nghiệm.

2. Thuật toán phân lớp Naïve Bayes

Naive Bayes Classifiers (NBC) là một trong những thuật toán tiêu biểu cho bài toán phân lớp dựa trên lý thuyết xác suất áp dụng định lý Bayes.

Định lý Bayes cho phép chúng ta có thể tính toán một xác suất chưa biết dựa vào các xác suất có điều kiện khác. Với công thức tổng quát tính xác suất của biến cố A với điều kiện biến cố B_k xảy ra trước (hay được gọi là xác suất hậu nghiệm):

Với P(A) > 0 và $\{B_1, B_2, \dots, B_n\}$ là một hệ đầy đủ các biến cố thỏa mãn tổng xác suất của hệ bằng 1 ($\sum_{k=1}^{n} P(B_k) = 1$) và từng đôi một xung khắc ($P(B_i \cap B_j) = 0$). Khi đó ta có:

$$P(B_k|A) = \frac{P(A|B_k)P(B_k)}{P(A)} = \frac{P(A|B_k)P(B_k)}{\sum_{i=1}^{n} P(A|B_i)P(B_i)} \quad (1)$$

Bộ phân lớp Naive bayes hoạt động như sau:

- Gọi D là tập dữ liệu huấn luyện, trong đó mỗi phần tử dữ liệu A chứa n giá trị thuộc tính B_1, B_2,...,B_n được biểu diễn bằng một vector n thành phần $\{x_1,x_2,...,x_n\}$
- Giả sử có m lớp C_1, C_2,..,C_m. Cho một phần tử dữ liệu A, bộ phân lớp sẽ gán nhãn cho A là lớp có xác suất hậu nghiệm lớn nhất. Cụ thể, bộ phân lớp Bayes sẽ dự đoán A thuộc vào lớp C_i nếu và chỉ nếu:

$$P(C_i|A) > P(C_j|A) \quad (i \le i, j \le m \; i \ne j) \quad (2)$$

Giá trị này sẽ tính dựa trên định lý Bayes.

- Để tìm xác suất lớn nhất, ta nhận thấy các giá trị P(A) là giống nhau với mọi lớp nên không cần tính. Do đó ta chỉ cần tìm giá trị lớn nhất của $P(A|C_i) * P(C_i)$. Chú ý rằng $P(C_i)$ được ước lượng bằng $|D_i|/|D|$, trong đó D_i là tập các phần tử dữ liệu thuộc lớp C_i. Nếu xác suất tiền nghiệm $P(C_i)$ cũng không xác định được thì ta coi chúng bằng nhau $P(C_1) = P(C_2) = ... = P(C_m)$, khi đó ta chỉ cần tìm giá trị $P(A|C_i)$ lớn nhất.
- Khi số lượng các thuộc tính mô tả dữ liệu là lớn thì chi phí tính toán $P(A|C_i)$ là rất lớn, do đó có thể giảm độ phức tạp của thuật toán Naive Bayes nếu giả thiết các thuộc tính độc lập nhau. Khi đó ta có thể tính:

$$P(A|C_i) = P(x_1|C_i) \ldots\ldots P(x_n|C_i) \quad (3)$$

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

Khi đó, P(xi|Cj) tỉ lệ với tần suất từ thứ i (hay thuộc tính thứ i cho trường hợp tổng quát) xuất hiện trong các văn bản của lớp Cj. Giá trị này có thể được tính bằng cách

$$P(x_i|C_j) = \frac{N_{ci}}{N_c} \quad (4)$$

Trong đó:

- N_{ci} là tổng số lần từ thứ i xuất hiện trong các văn bản của lớp C_j, nó được tính bằng tổng của tất cả các thành phần thứ i của các vector thuộc tính ứng với lớp C_j
- N_c là tổng số từ (kể cả lặp) xuất hiện trong lớp C_i. Nói cách khác, nó bằng tổng độ dài của toàn bộ các văn bản thuộc vào lớp C_j.

Cách tính này có một hạn chế là nếu có một từ mới chưa bao giờ xuất hiện trong lớp Cj thì biểu thức (1) sẽ bằng 0, điều này dẫn P(A|Ci) = 0 bất kể các giá trị còn lại có lớn thế nào. Việc này sẽ dẫn đến kết quả không chính xác. Để giải quyết việc này, một kỹ thuật được gọi là Laplace smoothing được áp dụng như trong biểu thức (5):

$$P(x_i|C_j) = \frac{N_{ci}+\alpha}{N_c+d\alpha} \quad (5)$$

Với α là một số dương, thường bằng 1, để tránh trường hợp tử số bằng 0. Mẫu số được cộng với dα để đảm bảo tổng xác suất $\sum_{i=1}^{d} P(x_i|C_j) = 1$ (Vũ Hữu Tiệp, 2018; Haiyi Zhang, DiLi, 2007; Wei Zhang, Feng Gao, 2011)

3. Trích chọn đặc trưng TF-IDF

Thuật ngữ TF-IDF (Term Frequency – Inverse Document Frequency) là một phương thức thống kê được biết đến rộng rãi để xác định độ quan trọng của một từ trong đoạn văn bản trong một tập nhiều đoạn văn bản khác nhau.

TF-IDF xác định trọng số của một từ trong văn bản thu được qua thống kê thể hiện mức độ quan trọng của từ này trong một văn bản, mà bản thân văn bản đang xét nằm trong một tập hợp các văn bản. Giá trị TF-IDF cao thể hiện độ quan trọng cao và nó phụ thuộc vào số lần từ xuất hiện trong văn bản nhưng bù lại bởi tần suất của từ đó trong tập dữ liệu. Một vài biến thể của TF-IDF thường được sử dụng trong các hệ thống tìm kiếm như một công cụ chính để đánh giá và sắp xếp văn bản dựa vào truy vấn của người dùng. TF-IDF cũng được sử dụng để lọc những từ stopwords trong các bài toán như tóm tắt văn bản và phân lớp văn bản.

TF (Term Frequency) – Tần suất xuất hiện của từ là số lần từ xuất hiện trong văn bản. Vì các văn bản có thể có độ dài ngắn khác nhau nên một số từ có thể xuất hiện nhiều lần trong một văn bản dài hơn là một văn bản ngắn. Như vậy, TF thường được chia cho độ dài văn bản (tổng số từ trong một văn bản).

$$TF(t,d) = \frac{f(t,d)}{\max\{f(w,d):w \in d\}} \quad (6)$$

Trong đó:

- TF(t, d) - Tần suất xuất hiện của từ t trong văn bản d.
- f(t,d) - Số lần xuất hiện của từ trong văn bản d.
- max{f(w,d): w ∈ d} – Số lần xuất hiện của từ có số lần xuất hiện nhiều nhất trong văn bản d.

IDF (Inverse Document Frequency) – Nghịch đảo tần suất của văn bản, giúp đánh giá tầm quan trọng của một từ. Khi tính tần số xuất hiện TF thì các từ đều được coi là quan trọng như nhau. Tuy nhiên có một số từ thường được được sử dụng nhiều nhưng không quan trọng để thể hiện ý nghĩa của đoạn văn. Vì vậy ta cần giảm đi mức độ quan trọng của những từ đó bằng cách sử dụng IDF:

$$IDF(t,D) = log\frac{|D|}{|\{d \in D: t \in d\}|} \quad (7)$$

Trong đó:

- IDF(t, D) – Giá trị IDF của từ t trong tập văn bản D.
- |D| - Tổng số văn bản trong tập D.
- |{d ∈ D : t ∈ d}| – Thể hiện số văn bản trong tập D có chứa từ t.

Cơ số logarit trong công thức này không làm thay đổi giá trị IDF của từ mà chỉ thu hẹp khoảng giá trị của từ đó. Việc sử dụng logarit nhằm giúp giá trị TF-IDF của một từ nhỏ hơn, do công thức tính TF-IDF của một từ trong một văn bản là tích của TF và IDF của từ đó. Công thức tính TF_IDF được xác định như sau:

$$TF - IDF(t,d,D) = TF(t,d)x\ IDF(t,D) \quad (8)$$

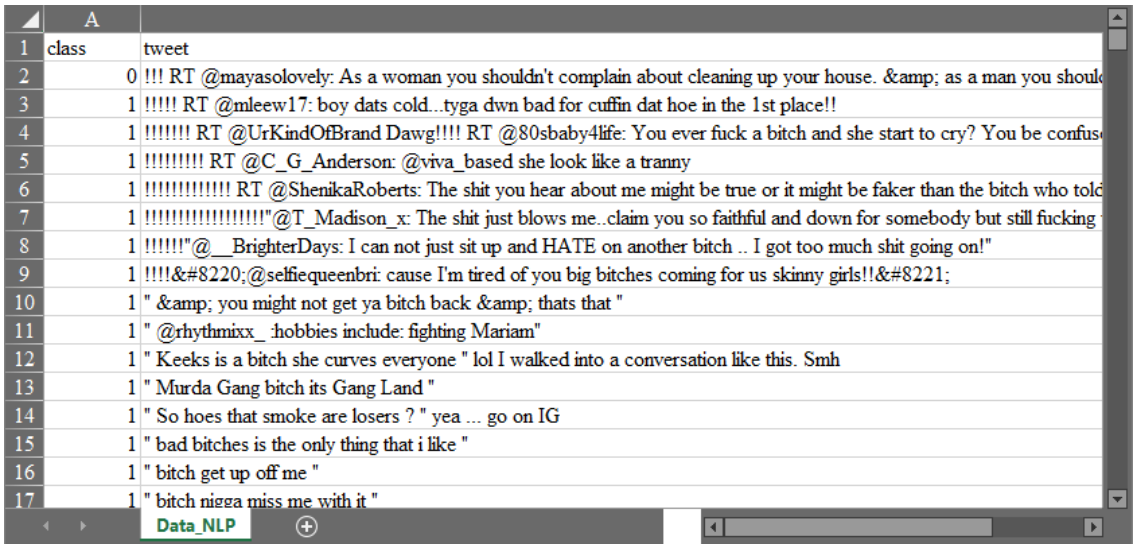
Từ (8) cho thấy những từ có giá trị TF-IDF cao là những từ xuất hiện nhiều trong văn bản này, và xuất hiện ít trong các văn bản khác. Việc này giúp lọc ra những từ phổ biến và giữ lại những từ có giá trị cao chính là các từ khóa của văn bản đó.

4. Ứng dụng thuật toán Naive Bayes trên tập dữ liệu Twitter

4.1. Chuẩn bị dữ liệu

Tập dữ liệu các nhận xét (comments) được thu thập từ Twitter và lưu trữ trong tệp Data_NLP.csv như minh họa trong hình 2, chứa 56 700 dòng dữ liệu; với 2 cột bao gồm:

- Cột class: cho biết lớp mà comments được gán nhãn; các comments được gán vào 2 lớp: 0 – Non Toxic (33 847 comments) | 1 – Toxic (22 853 comments).
- Cột tweet: cho biết nội dung của đoạn văn bản comments. Đây là đoạn văn bản gốc được lấy về từ Twitter nên có thể thấy chứa rất nhiều nhiễu (noise)



	A	
1	class	tweet
2	0	!!! RT @mayasolovely: As a woman you shouldn't complain about cleaning up your house. & as a man you should
3	1	!!!!! RT @mleew17: boy dats cold...tyga dwn bad for cuffin dat hoe in the 1st place!!
4	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby4life: You ever fuck a bitch and she start to cry? You be confuse
5	1	!!!!!!!!! RT @C_G_Anderson: @viva_based she look like a tranny
6	1	!!!!!!!!!!!!! RT @ShenikaRoberts: The shit you hear about me might be true or it might be faker than the bitch who told
7	1	!!!!!!!!!!!!!!!!!!"@T_Madison_x: The shit just blows me..claim you so faithful and down for somebody but still fucking
8	1	!!!!!!"@__BrighterDays: I can not just sit up and HATE on another bitch .. I got too much shit going on!"
9	1	!!!!“@selfiequeenbri: cause I'm tired of you big bitches coming for us skinny girls!!”
10	1	" & you might not get ya bitch back & thats that "
11	1	" @rhythmixx_ :hobbies include: fighting Mariam"
12	1	" Keeks is a bitch she curves everyone " lol I walked into a conversation like this. Smh
13	1	" Murda Gang bitch its Gang Land "
14	1	" So hoes that smoke are losers ? " yea ... go on IG
15	1	" bad bitches is the only thing that i like "
16	1	" bitch get up off me "
17	1	" bitch nigga miss me with it "

Data_NLP

Hình 2. Tập dữ liệu thô đã được gán nhãn ban đầu Data_NLP.csv

Có thể thấy các câu comments là dữ liệu thô ban đầu (raw data) được lấy về từ Twitter chứa rất nhiều nhiễu như: các ký hiệu đặc biệt, các thẻ HTML, JavaScript, các từ viết tắt, các liên kết, email, con số, tag ....Do đó cần phải được chuẩn bị trước khi sử dụng cho các mục đích tiếp theo. Như trong phần mở đầu đã chỉ ra, chuẩn bị dữ liệu (Data Preparation) là giai đoạn thực hiện đầu tiên trong quá trình xây dựng một mô hình học máy cho bài toán phân lớp văn bản.

Các phương pháp mà nhóm tác giả sử dụng để chuẩn bị dữ liệu áp dụng cho tập dữ liệu Data_NLP bao gồm:

- Loại bỏ nhiễu trong các comments bao gồm: Loại bỏ các đường link, địa chỉ email; Loại bỏ các ký tự đặc biệt, ký tự số.
- Loại bỏ các từ StopWords: StopWords là những từ xuất hiện nhiều trong ngôn ngữ tự nhiên, tuy nhiên lại không mang nhiều ý nghĩa. Để loai bỏ StopWords có 2 cách chính là dùng từ điển (corpus) hoặc dựa theo tần suất xuất hiện của từ. Với tiếng anh, những từ StopWords bao gồm: the, this, that, these, is, are, was, were….đã được tổng hợp vào trong một corpus trong thư viện NLTK.
- Chuẩn hóa từ: Trong tiếng Anh, mỗi một từ có thể có nhiều biến thể khác nhau. Điều này làm cho việc so sánh giữa các từ là không thể mặc dù về mặt ý nghĩa cơ bản là như nhau. Ví dụ như các từ "walks", "walking", "walked" đều là biến thể của từ "walk". Để biến đổi các từ này về dạng gốc 2 kỹ thuật thường được sử dụng là Stremming và Lemmatiaztion.

Nhóm tác giả sử dụng ngôn ngữ lập trình Python, các thư viện hỗ trợ bao gồm: Pandas, NumPy, NLTK, RE, mã nguồn được viết trên hệ thống Google Colab. Kết quả sau khi chạy các hàm chuẩn bị dữ liệu được thể hiện như trong bảng 1 với 5 comments trước và sau khi xử lý.

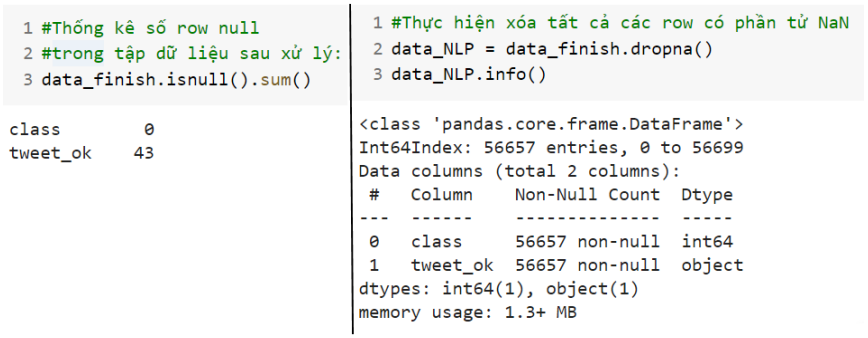
Bảng 1: Minh họa 5 comments trước và sau khi thực hiện xử lý

STT	Comments ban đầu	Comments đã xử lý
1	!!!!!!!!!!!!!!!!!"@T_Madison_x: The shit just blows me..claim you so faithful and down for somebody but still fucking with hoes! 😂😂😂"	shit blow faithful somebody still fuck hoe
2	" got ya bitch tip toeing on my hardwood floors " 😂 http://t.co/cOU2WQ5L4q	get ya bitch tip toe hardwood floor
3	"@DunderbaII: I'm an early bird and I'm a night owl, so I'm wise and have worms."	early bird night owl wise worm
4	"@Tmacc_GFG: “@VoiceOfDStreetz: "@Tmacc_GFG: “@tizzimarie: No slushes 😥”hoes nasty anyway fam"😴”them hoes taste like meds"🍇🍇🍼	slush hoe nasty anyway fam hoe taste like meds
5	beÃ°ÂŸÂ’Â“ Ã°ÂŸÂ’Â“@user @user @user @user @user @user @user @user @user	

4.2. Trích chọn đặc trưng

Như đã trình bày trong phần mở đầu, giai đoạn thứ 2 trong xây dựng mô hình học máy cho bài toán NLP đó là trích chọn đặc trưng (Feature Engineering), có rất nhiều phương pháp để thực hiện trích trọn đặc trưng chuyển đổi văn bản sang vector dữ liệu số. Tác giả sử dụng phương pháp tính TF-IDF để thực hiện việc này.

Sau khi thực hiện chuẩn bị dữ liệu như trong phần 4.1, một số comments sau xử lý chỉ còn là chuỗi rỗng. Thống kê cho thấy có 43 chuỗi comments sau khi xử lý chỉ còn lại là chuỗi rỗng (NaN). Do đó, trước khi thực hiện trích chọn đặc trưng cho toàn bộ tập comments này, các chuỗi rỗng cần phải được xử lý. Tác giả thực hiện loại bỏ các dòng comments chứa các chuỗi rỗng này ra khỏi tập dữ liệu bằng phương thức dropna() trong thư viện Pandas (Hình 3). Như vậy, tập dữ liệu ban đầu bao gồm 57 000 comments, sau khi loại bỏ thu được tập dữ liệu mới còn lại 56 657 comments.



Hình 3. Thống kê và xử lý commnets rỗng trong Pandas

Thực hiện tách tập dữ liệu thu được sau khi loại bỏ các comments rỗng thành 2 tập con sử dụng cho việc

huấn luyện và kiểm thử mô hình theo tỷ lệ 80% cho tập huấn luyện và 20% cho tập kiểm thử. Tác giả sử dụng thư viện sklearn để tự động hóa việc tách này. Kết quả sau khi thực hiện việc tách tập dữ liệu data_NLP sẽ thu được tập huấn luyện chứa 45 325 comments (chiếm 80%), tập kiểm thử chứa 11 332 comments (chiếm 20%) như Hình 4.

```
#THỰC HIỆN TÁCH TẬP DỮ LIỆU THÀNH TẬP TRAIN VÀ TEST
from sklearn import model_selection
#Tách tập dữ liệu thành Train - Test (tỷ lệ: 0.8 - 0.2)
train_x,test_x,train_y,test_y=model_selection.train_test_split(
                                                data_NLP['tweet_ok'],
                                                data_NLP['class'],
                                                test_size=0.2)

print('Tập Train (80%): ', train_x.shape)
print('Tập Test (20%): ', test_x.shape)
```

```
Tập Train (80%):  (45325,)
Tập Test (20%):  (11332,)
```

Hình 4. Phân tách tập dữ liệu thành tập Train (80%) – tập Test (20%)

Để thực hiện trích chọn đặc trưng theo phương pháp TF-IDF đã được mô tả trong phần 3, tác giả sử dụng module TfidfVectorizer trong thư viện Sklearn. Kết quả sau quá trình này sẽ thực hiện chuyển đổi dữ liệu văn bản sang vector số (Hình 5); Dữ liệu số này sẽ được sử dụng là đầu vào cho mô hình học máy Naive Bayes để thực hiện phân lớp các comments.

```
# Tính TF-IDF cho tập dữ liệu
from sklearn.feature_extraction.text import TfidfVectorizer
#Convert a collection of raw documents to a matrix of TF-IDF features.
vector = TfidfVectorizer(analyzer='word',
                         max_features=20000,
                         stop_words = 'english')

vector.fit(data_NLP['tweet_ok'])
xtrain_tfidf = vector.transform(train_x)
xtest_tfidf = vector.transform(test_x)
print('Kết quả Vector hóa tập Train sang dạng số:')
print(xtrain_tfidf.data)
print(xtrain_tfidf.shape)
print('Kết quả Vector hóa tập Test sang dạng số:')
print(xtest_tfidf.data)
print(xtest_tfidf.shape)
```

```
Kết quả Vector hóa tập Train sang dạng số:
[0.54907788 0.53160745 0.64490852 ... 0.33670852 0.13308591 0.26849291]
(45325, 20000)
Kết quả Vector hóa tập Test sang dạng số:
[0.73386444 0.51887036 0.43842506 ... 0.58960476 0.39331306 0.52415607]
(11332, 20000)
```

Hình 5. Kết quả vector hóa văn bản sang dạng số sử dụng TF-IDF

4.3. Xây dựng mô hình

Để sử dụng thuật toán phân lớp Naive Bayes, tác giả sử dụng thư viện học máy Sklearn; Sklearn cung cấp 5 giải thuật của Naive Bayes bao gồm: Gaussian Naive Bayes; Multinomial Naive Bayes; Complement Naive Bayes; Bernoulli Naive Bayes; Categorical Naive Bayes. Như so sánh thể hiện trong Hình 1, giải thuật Multinomial Naive Bayes cho bài toán phân lớp văn bản có hiệu quả cả về Score, Training time và Test time. Vì vậy, tác giả sử dụng Multinomial Naive Bayes áp dụng cho tập dữ liệu của mình. Muiltinomial Naive Bayes có 2 tham số đầu vào chính bao gồm:

- alpla: Tham số dùng để làm mịn (Laplace smoothing); giá trị thiết lập là một số thực nằm trong đoạn [0,1]; Thiết lập mặc định bằng 1.
- fit_prior: Tham số dùng để xác định có sử dụng xác suất các phần tử trước đó cho việc huấn luyện hay không; Giá trị thiết lập kiểu boolean (True|Flase), Thiết lập mặc định bằng True.

Quá trình chạy mô hình với tham số alpha =0.79, fit_prior=True cho kết quả tốt nhất. Hình 6 thể hiện việc xây dựng mô hình MultinomialNB, huấn luyện trên tập Train, và kiểm thử trên tập Test. Độ chính xác

trên khi huấn luyện mô hình đạt 93.83%, độ chính xác của mô hình khi chạy trên tập Test đạt 91.63 %.

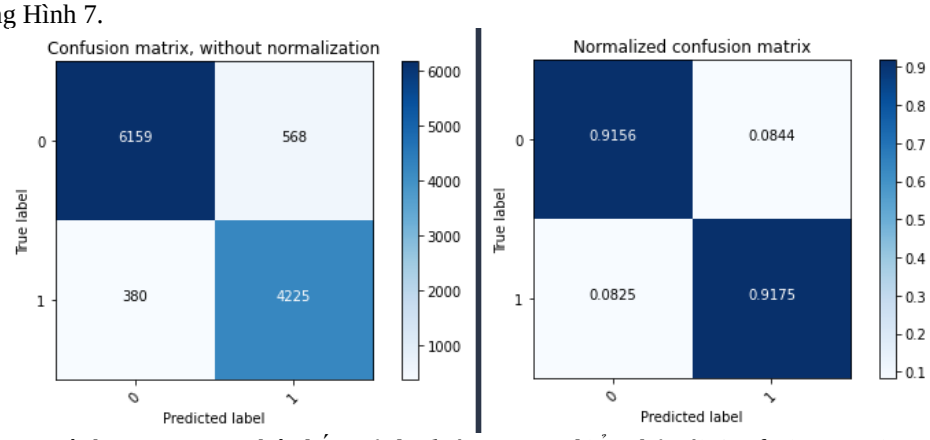
```
#Sử dụng mô hình Naive Bayes với TF-IDF
import time
from sklearn import naive_bayes as nb
from sklearn.metrics import accuracy_score

#Sử dụng thuật toán MultinomialNB
MultiNB = nb.MultinomialNB(alpha=0.79, fit_prior=True)
#Huấn luyện mô hình với tập huấn luyện Train
MultiNB.fit(xtrain_tfidf,train_y)
train_score = MultiNB.score(xtrain_tfidf, train_y)
print('Độ chính xác của mô hình trên tập Train: ', round(train_score*100,2),'%')

#Sử dụng mô hình huấn luyện dự đoán trên tập TEST
y_pred = MultiNB.predict(xtest_tfidf)
#Đánh giá độ chính xác của mô hình trên tập Test
test_sore = round(accuracy_score(test_y,y_pred)*100,2)
print('Độ chính xác của mô hình trên tập Test: ', test_sore, '%')
```

```
Độ chính xác của mô hình trên tập Train:  93.83 %
Độ chính xác của mô hình trên tập Test:  91.63 %
```

Hình 6. Xây dựng mô hình phân lớp văn bản sử dụng MultinomialNB

4.4. Đánh giá kết quả

Như ở trên cho thấy độ chính xác của mô hình phân lớp MultinomialNB trên tập dữ liệu Test bao gồm 11 332 Comments đạt 91.63%; Thời gian để huấn luyện mô hình hết 0.013969 giây, thời gian để chạy mô hình trên tập dữ liệu kiểm thử hết 0.002822 giây. Để có cái nhìn tổng quan hơn về kết quả đánh giá độ chính xác của mô hình trên tập kiểm thử, tác giả sử dụng Confusion matrix để trực quan hóa kết quả thể hiện trong Hình 7.



Hình 7. Trực quan hóa kết quả dự đoán trên tập kiểm thử với Confusion matrix

Có tất cả 10 384 comments trên tổng số 11 332 comments trong tập kiểm thử được dự đoán chính xác. Trong đó:

- 6 159 comments lớp 0 và được dự đoán đúng vào lớp 0 (91.56%);
- 4 225 comments lớp 1 và được dự đoán đúng vào lớp 1 (91.75%);
- 568 comments lớp 0 nhưng mô hình dự đoán sai sang lớp 1 (8.44%)
- 380 comments lớp 1 nhưng mô hình dự đoán sai sang lớp 0 (8.25%)

Bên cạnh đó, tác giả cũng thực hiện trích chọn đặc trưng theo phương pháp Bag of Words (BoW) và cài đặt bộ phân lớp với 2 thuật toán học máy khác để đánh giá là Support Vector Machines (SVM) và K-Nearest Neighbors (KNN).

Bảng 2. Độ chính xác của các thuật toán phân lớp trên tập kiểm thử

Phương pháp trích chọn đặc trưng	Thuật toán phân lớp		
	MultinomialNB (alpha=0.79)	SVM (C=0.5)	KNN (n_neughbors=10)
TF-IDF	91.63%	59.66%	67.94%
BoW	71.83%	59.29%	42.95%

Kết quả thể hiện độ chính xác của các mô hình trên tập kiểm thử được thống kê trong Bảng 2. Có thể

thấy rằng Naïve Bayes là thuật toán tốt cho mô hình phân lớp văn bản với độ chính xác cao. Ngoài ra phương pháp trích chọn đặc trưng cũng ảnh hưởng rất lớn tới sự chính xác của mô hình.

5. Kết luận

Học máy đã được nghiên cứu, ứng dụng trong rất nhiều lĩnh vực, trong đó có lĩnh vực xử lý ngôn ngữ tự nhiên (NLP) và thu được nhiều kết quả tích cực. Trong các bài toán NLP, phân lớp văn bản là bài toán cơ bản và rất quan trọng. Trong nội dung của bài báo này, tác giả đã nghiên cứu và tìm hiểu về một trong những thuật toán rất hiệu quả cho phân lớp văn bản đó là Naive Bayes. Các kết quả trong phần thực nghiệm phân lớp văn bản với tập 56 700 comments trên Twitter thành 2 lớp 0 – Non Toxic và 1 – Toxic đã cho thấy hiệu quả của thuật toán Naive Bayes so với các thuật toán học máy khác. Kết quả thực nghiệm cũng chỉ ra rằng, độ chính xác của một mô hình phân lớp không những phụ thuộc vào thuật toán, tham số thiết lập cho thuật toán đó mà còn phụ thuộc rất lớn vào phương pháp trích chọn đặc trưng của tập dữ liệu đầu vào. Tuy nhiên, có thể khẳng định Naive Bayes Classifiers (NBC) là một lựa chọn tốt cho các bài toán phân lớp văn bản (Text classification), NBC có thời gian huấn luyện, kiểm thử mô hình rất nhanh và độ chính xác cao.

Tài liệu tham khảo

Vũ Hữu Tiệp, 2018, *Bài giảng Học Máy (Machine Learning)*, Nhà xuất bản Khoa học kỹ thuật.145-147

Haiyi Zhang, DiLi; 2007; Naïve Bayes Text Classifier, *Cranular Computing – IEEE international*, GrC2007.

Wei Zhang, Feng Gao, 2011; An Improvement to Naïve Bayes for Text Classification, *Procedia Engineering*. 15 (2011), 2160 – 2164.

ABSTRACT

Research on Naïve Bayes classifiers to text classification; Application for Twitter's comments data set.

Dang Van Nam [1]

[1] *Hanoi University of Mining and Geology*

The content of this article will research on Naïve Bayes classifiers to text classification and application for Twitter's comments data set. Naïve Bayes classifiers which are widely used for text classification in machine learning are based on the conditional probability of features belonging to a class, in which the features are selected by feature selection methods. The paper specifically presents the construction of a machine learning model with the Naïve Bayes algorithm using TF-IDF features in text classification. The experimental data set includes 56 700 comments on Twitter and that is divided into 2 classes: class 0 - Non Toxic, class 1 - Toxic. At the same time, in the content of the article, the author also compared the accuracy of Naïve Bayes with SVN algorithm and KNN on the same experimental data set.

Keywords: Machine Learning; Naïve Bayes; NLP; Text classification; TF-IDF