# Bảng phân công công việc và đánh giá

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MSSV** | **Thành viên** | **Danh sách công việc** | **Mức độ hoàn thành** | **Tổng hợp đánh giá chéo** |
| 16520724 | Bùi Nguyên Mão |  |  |  |
| 2 | Cao Văn Nam |  |  |  |
| 3 | Thừa Lê Thanh Chương |  |  |  |
| 4 | Bùi Văn Thuận |  |  |  |

# Nội dung

## Giới thiệu đề tài:

Công nghệ thông tin hiện nay đã mở ra một thời kỳ phát triển mạnh mẽ của xã hội tri thức, nơi mà tất cả mọi người đều có thể tiếp cận dễ dàng với tin tức bất kể không, thời gian. Tuy nhiên, mặt trái của nó đó là sự phát tán các tin tức nhảm, sai lệch vẫn luôn hiện diện như một căn bệnh từng bước đẩy con người vào bước đường lạc lối. Với số lượng tin bài được xuất hiện mỗi giây, rất khó có một tổ chức hay các nhân nào có thể kiểm tra, rà soát được sự đúng đắn hay nội dung vì vậy đây là một vấn đề nhức nhối mà cần phải có biện pháp để giải quyết.

Sự phát triển của machine learning và natural language processing như một vị cứu tinh cho đứa con chiên đang lạc lối. Với những công nghệ này, con người có thể giải quyết được một phần vấn đề trên.

Đề tài này chỉ là một bước tìm hiểu nhỏ nhưng sẽ mang lại một cái nhìn rõ hơn cho mọi người về cách mà NLP đã và đang tiếp tục hoạt động, phát triển để góp phần xây dựng một môi trường dữ liệu trong sạch.

*Bài toán đặt ra:*

kenh12 sau khi nhận đươc nhiều báo cáo vì số lượng tin bài giả đang gia tăng nghiêm trọng trên mạng xã hội của mình đã quyết đinh phải kiểm duyệt tất cả tin tức bài đăng, nhưng do số lượng tin bài nhận được mỗi ngày quá nhiều, bộ phận kiểm tra gồm 100 nhân viên làm ngày làm đêm vẫn không thể hoàn thành được nhiệm vụ. Một cuộc họp khẩn giữa 12 giờ đêm đã được diễn ra, cuối cùng cũng đưa ra một phương pháp có thể giải quyết được vấn đề trên.

*Giải pháp:*

* Sử dụng máy tính để làm việc kiểm duyệt thay cho con người.
* Xây dựng bộ dữ liệu lớn để máy phân loại được các bài viết theo tiêu chí các bài sử dụng các từ tương đồng nhau thường có mục đích như nhau.
* Để tối ưu chi phí và nguồn lực hiện có, tạm thời chỉ kiểm tra với tiêu đề của bài viết.

## Dữ liệu - Phân tích - Hướng giải quyết:

### Bộ dữ liệu

Đề tài sử dụng bộ dữ liệu tiêu đề tin tức ở nguồn: [*https://www.kaggle.com/rmisra/news-headlines-dataset-for-sarcasm-detection*](https://www.kaggle.com/rmisra/news-headlines-dataset-for-sarcasm-detection)

Thông tin bộ dữ liệu:

* Dữ liệu được thu thập ở 2 trang: *theonion* chuyên viết các bài châm biếm, mỉa mai và *huffingtonpost* chuyên viết tin chính thống.
* Gồm 26709 dòng dữ liệu kiểu JSON, có các thông tin: link bài viết, tiêu đề và nhãn (có phải tin bài mỉa mai)
* Các thông tin tiêu đề đều bằng tiếng Anh.

### Hướng giải quyết

Bài toán trở thành một bài toán phân lớp dữ liệu ngôn ngữ tự nhiên. Máy tính chỉ có thể xử lý với các con số, nên cần có bước đầu tiên là bước chuyển dữ liệu chữ sang số. Có một số kỹ thuật để giải quyết vấn đề này nhưng ở đây, chúng ta sẽ chọn cách đơn giản là Bag of Words. Kỹ thuật này sẽ xây dựng một bộ dữ liệu tất cả các từ được sử dụng rồi từ đó sẽ tạo ra các vector với số chiều là số lượng từ tương ứng với mỗi chuỗi đầu vào.

Tuy nhiên, trước đó dữ liệu văn bản cần một bước làm sạch với các quá trình chuẩn hóa từ và loại bỏ các từ không cần thiết. Trong phần lớn các ngôn ngữ, có một số từ được thêm vào câu với mục đích tăng sắc thái biểu đạt hoặc cảm xúc, những từ này cần được loại bỏ để giảm số lượng chiều của vector đầu vào, tăng hiệu quả mô hình.

Sau khi đã chuyển dữ liệu sang dạng số, các vector ở đây là vector thưa có chiều rất lớn, vì vậy để tăng hiệu quả quá trình học và giảm thời gian chạy cần có biện pháp giảm số chiều dữ liệu nhưng vẫn phải cố gắng giữ được tương quan giữa các dữ liệu với nhau.

Sau khi xong phần chuẩn bị dữ liệu và tiền xử lý, cần lựa chọn thuật toán phù hợp. Với dạng bài toán binary classification, có khá nhiều phương pháp để xử lý như regression, knn, decision tree, svm,… Ở đây, chúng ta dùng naive bayes.

## Tiền xử lý

*Code tiền xử lý nằm trong file make\_vector.py*

Đây là giai đoạn cực kỳ quan trọng của quá trình học máy, thường chiếm một phần lớn thời gian của cả quá trình. Với bài toán trên, sẽ có những bước tiền xử lý sau:

### Chuẩn hóa dữ liệu và tách từ:

Dữ liệu hiện tại đã được chuẩn hóa về nội dung, cấu trúc, không có trùng lặp, vô lý,…

Tùy thuộc vào loại dữ liệu, loại ngôn ngữ và mục đích sử dụng, có một số cách để chia nhỏ nội dung với các đặc điểm sau:

* Tùy theo loại ngôn ngữ mà cấu trúc của phần con được chia ra cho phù hợp nghĩa, ví dụ trong tiếng Anh, mỗi câu gồm nhiều từ, một từ thường cách nhau theo khoảng trắng, nhưng trong tiếng Việt, câu được tạo thành từ nhiều tiếng, một từ có thể phải được kết hợp từ nhiều tiếng nên việc tách câu theo khoảng trằng là không đảm bảo nội dung. Bên cạnh đó, tùy thuộc yêu cầu bài toán mà phải tách từ hoặc cụm từ, cụm cấu trúc (chủ ngữ, vị ngữ),…
* Có nhiều kiểu tách từ trong tiếng Anh, tuy nhiên khó có cách nào có thể tách trọn vẹn, hoàn chỉnh nội dung của các câu bởi số lượng từ vựng và cụm từ lớn, các dấu câu mang các ý nghĩa khác nhau tùy thuộc cách sử dụng.

Với dữ liệu ở đây, cách đơn giản nhất được sử dụng là loại bỏ tất cả dấu và ký tự đặc biệt tách từ đơn theo khoảng trắng.

### Hoàn nguyên từ:

Với một số ngôn ngữ trong đó có tiếng Anh, các từ trong quá trình sử dụng có sự biến đổi thành các dạng khác nhau nhưng phần lớn tạo ra ý nghĩa khá là gần nhau.

ví dụ: learn → learns → learning

Trong máy học với công nghệ hiện tại, các sự biến đổi này là không cần thiết, nó chỉ làm tăng thêm độ phức tạp mà máy cần phải xử lý vì vậy cần có một bước để đưa các từ về với dạng gốc ban đầu.

Có 2 phương pháp đưa từ về dạng gốc:

* stemming là một cách hoàn nguyên từ theo kiểu đánh đồng, các từ với các kết cấu giống nhau sẽ được đưa về dạng giống nhau phần lớn theo dạng bỏ bớt ký tự. ví dụ: learning → learn, machines→ machin, studies → studi
* lemmatizing là một phiên bản nâng cấp của stemming khi các từ tạo ra phải qua quá trình kiểm tra lại với từ điển để đảm bảo sự chính xác. Tất nhiên cái giá phải trả là thời gian và quá trình xử lý. ví dụ: machines → machine, studies → study.

Cả 2 phương pháp đều có thể được sử dụng tùy theo mục đích và yêu cầu. Ở bài toán này, chúng tôi chọn lemmatizing.

Do lemmatizing yêu cầu một bộ từ điển để xác thực từ được tạo ra nên chúng tôi sử dụng bộ wordnet của thư viện nltk.

### Loại bỏ stop words

Trong các câu văn, có những từ chứa ít dữ liệu cho quá trình xử lý phân lớp sẽ gây nhiễu rất lớn đối với mô hình BoW.

Các stop words trong tiếng anh thường là giới từ, từ nối,... như *the, there, an, anyway, hereby,...*

Ví dụ trong tiếng Anh:

*I hope that, when I've built up my savings, I'll be able to travel to Mexico.*

Câu này nếu được rút gọn lại với các từ: hope, built, up, savings, able, travel, Mexico. thì cơ bản nghĩa vẫn sẽ được dữ lại. Quá trình học máy chỉ cần có thế thôi.

### Tạo vector từ dữ liệu:

Sau các quá trình chuẩn hóa trên, các câu đã được tách thành một danh sách các từ phù hợp, với kỹ thuật Bag of Words ta tiến hành gom tất cả các từ tạo thành một bộ danh sách các từ được sử dụng sau đó chuyển các câu đầu vào thành các vector biểu thị số lần xuất hiện của từ trong câu. Với mô hình này, máy sẽ bỏ qua tất cả ngữ nghĩa hay cú pháp của câu.

Các vector này này sẽ có số chiều rất lớn và phần lớn sẽ có giá trị là 0 nên được gọi là sparse vector.

### Giảm chiều dữ liệu và trực quan hóa:

Với dữ liệu có chiều lớn như trên, sẽ có những phương pháp để giảm đi số chiều mà vẫn cố gắng dữ tối đa tương quan giữa các điểm. Một số nguyên tắc của giảm chiều dữ liệu là:

* Dimension reduction là một kỹ thuật máy học giúp giảm số lượng thuộc tính đồng thời cố gắng dữ lại nhiều thông tin nhất có thể. Nó thường được thực hiện bằng cách tạo ra được bộ thuộc tính mới có thể thay thế bộ thuộc tính cũ.
* Việc giảm chiều dữ liệu vữa dễ trực quan hóa vừa giảm sự dư thừa tối ưu quá trình tính toán.
* Phần lớn các= phương pháp giảm chiều dữ liệu là các thuật toán học máy không giám sát bởi vì các nhãn (nếu có) không được sử dụng trong quá trình này.
* Một số phương pháp nổi tiếng như: PCA, NMF,…

Ở đây, dùng t-SNE (t-Distributed Stochastic Neighbor Embedding):

* Thuật toán này nhúng dữ liệu chiều cao vào không gian chiều thấp hơn trong đó sự tương đồng giữa các mẫu dữ liệu lân cận là được bảo toàn.
* Ở đây dùng hàm TSNE dựng sẵn của thư viện sklearn.

Trực quan hóa dữ liệu bằng đồ thị:

* Sau khi giảm chiều dữ liệu thành công, dữ liệu đầu ra này sẽ có thể được biểu diễn trên đồ thị dễ dàng.
* Ở đây dùng thư viện matplotlib để vẽ.

### Hiện thực với python:

* Chuẩn hóa và chuyển sang vector với dữ liệu đầu vào

## Huấn luyện với naive bayes

### Lý do chọn naive bayes

Với việc trực quan hóa, ta có thể thấy rõ rằng những phương pháp như regression hoặc svm có thể được sử dụng để giải quyết bài toán trên nhưng ở đây chúng tôi chọn naive bayes.

Với những bài toán phân lớp với dữ liệu ngôn ngữ tự nhiên với mô hình bag of words, thường thuật toán naive bayes sẽ đưa ra kết quả tốt:

* BoW coi các thuộc tính là độc lập, không có liên quan với nhau tương tự với thuật toán naive bayes.
* Với bài toán có dữ liệu đầu vào nhiều chiều như thế này, các thuật toán phân lớp khác như KNN khó hiệu quả do đòi hỏi rất nhiều việc tính toán.
* Decision tree sẽ tỏ ra không hiệu quả với sparse vector do dễ dàng gây ra overfit.

### Thuật toán naive bayes:

**Ý tưởng:** Với mỗi thuộc tính của dữ liệu cần dự đoán, xét xác suất của tất cả các nhãn có thể xảy ra và sẽ chọn nhãn có xác suất cao hơn.

**Thuật toán[1]:**

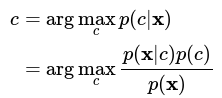
* Xét bài toán phân lớp với C lớp, xác suất một điểm dữ liệu X thuộc vào lớp c là xác suất có điều kiện của c với điều kiện X. Chúng ta cần tìm c để cho xác suất đó lớn nhất:



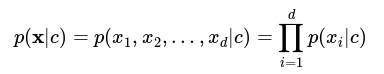
* Theo công thức bayes:



* Rút ra được:

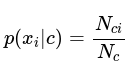


* p(c) có thể dễ dàng tính được với dữ liệu đầu vào, p(X) không phụ thuộc vào giá trị của c nên có thể loại bỏ. Vì vậy nhiệm vụ là phải tính được p(X|c).
* Quá trình tính p(X|c) sẽ rất khó khăn do X có thể có nhiều thuộc tính vì vậy để thuận tiện trong tính toán, chúng ta sẽ coi các thuộc tính trong X là các biến ngẫu nhiên độc lập và sẽ có công thức tính p(X|c như sau)

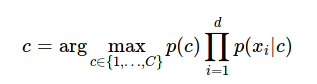


**Giải thuật:**

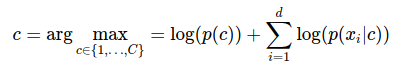
* Tính xác suất của các nhãn p(c) từ dữ liệu đầu vào
* Với dữ liệu đầu vào danh sách các từ xuất hiện trong câu và thuộc tính là một loạt các từ, vậy suy ra, xác suất để một câu có từ đó và có nhãn tương ứng là tỉ lệ số từ đó xuất hiện trên tổng số từ. Đây gọi là mô hình multinomial naive bayes.



* Với Nci là số lần xuất hiện của từ thứ i.
* Nc là tổng số từ xuất hiện.
* Tính ra xác suất điều kiện cần tìm với mỗi thuộc tính xuất hiện trong dữ liệut test theo công thức:

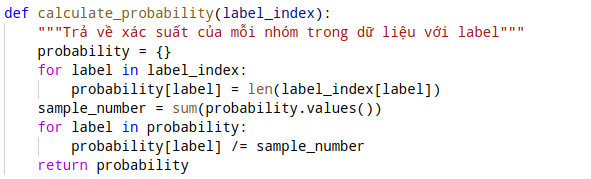


* Tuy nhiên, do các giá trị p(xi|C) tính ra có thể rất nhỏ nên khi nhân lại rất dễ gây tràn bộ nhớ, vì vậy người ta thường sử dụng log, công thức thay thế:

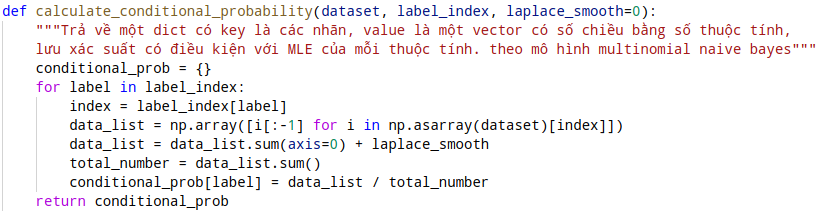


### Hiện thực với python

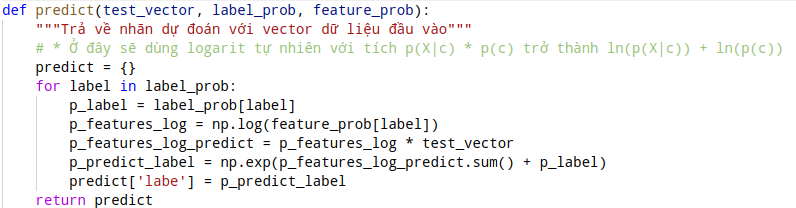
* Hàm tính xác suất của các nhãn p(c)



* Hàm tính xác suất của các thuộc tính với điều kiện là nhãn

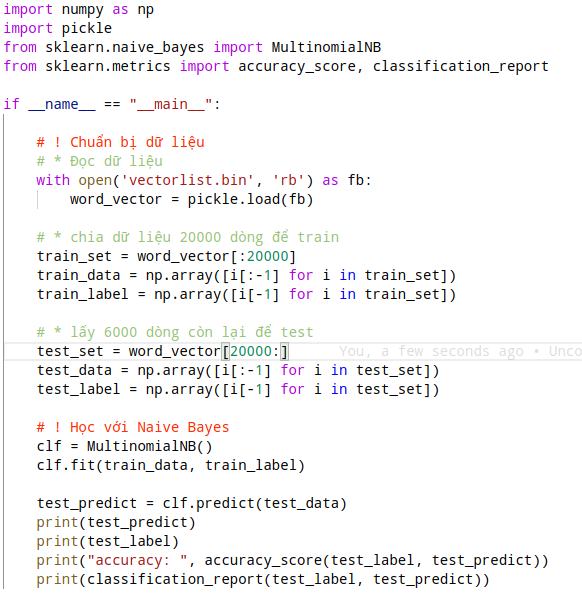


* Dựa vào các kết quả xác suất, tìm ra dự đoán:



### Code với naive bayes của thư viện sklearn:

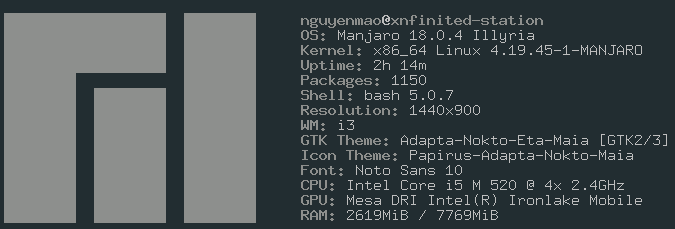
* Sklearn là một thư viện máy học đồ sộ đã có xây dựng sẵn rất nhiều mô hình thuật toán cho các bài toán máy học.



## Kết quả, đánh giá

### Các lần học và đánh giá mô hình:

* Cấu hình máy tính:



CPU: core i5 M520 4 x 2.4 GHz

RAM: 8 GiB

SWAP: 4GiB SSD

HDH: Manjaro 18.0.4 64 bit

* Quá trình tiền xử lý và tạo vector: *python make\_vector.py*

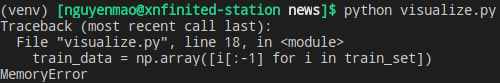
**

Thời gian chạy quá trình tiền xử lý là khoảng 228s, Quá trình ghi vector ra file là 20s tổng thời chạy là 248s

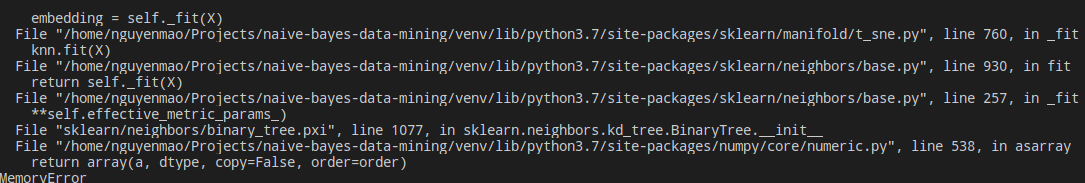
Sau quá trình trên, dữ liệu vector được ghi ra file *vectorlist.bin* để có thể sử dụng trong quá trình xử lý sau này.

* Giảm chiều dữ liệu và trực quan hóa dữ liệu: *python visualize.py*

**Lần 1:** Với 20000 dòng dữ liệu đầu tiên và lấy tất cả thuộc tính để train, máy báo lỗi tràn bộ nhớ

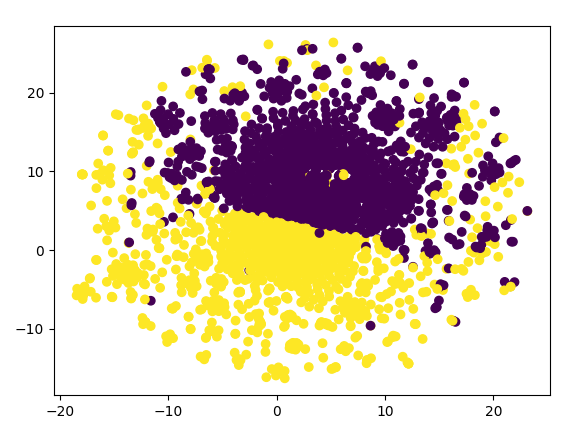


**Lần 2:** Với 20000 dòng dữ liệu đầu tiên làm tập train và chọn 10000 thuộc tính xuất hiện nhiều nhất, máy tiếp tục báo lỗi tràn bộ nhớ

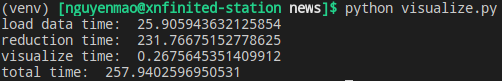


**Lần 3:** Chỉ lấy 2000 dòng dữ liệu và 2000 thuộc tính xuất hiện nhiều nhất.

Kết quả biểu đồ thu được:



Thời gian chạy:



Thời gian đọc dữ liệu 26s, quá trình giảm chiều với t-SNE mất 232s, vẽ biểu đồ mất 0.3s, tổng thời gian là khoảng 258s.

→ Các nhóm dữ liệu xếp khá rõ ràng và không có nhiều điểm chồng chéo lên nhau nên có thể sử dụng một vài thuật toán khác để phân lớp như regression, svm.

* Chạy với code tự viết: *python naive\_bayes.py*

**Lần 1:** Chạy với tất cả dữ liệu và tất cả các thuộc tính, lỗi tràn bộ nhớ.

**Lần 2:** Chạy với 10000 dòng và 10000 thuộc tính xuất hiện nhiều nhất.

**Lần 3:** 5000 dòng với 10000 thuộc tính xuất hiện nhiều nhất.

* Chạy với thư viện:

**Lần 1:** Chạy với tất cả dữ liệu và tất cả các thuộc tính, lỗi tràn bộ nhớ.

**Lần 2:** Chạy với 10000 dòng và 10000 thuộc tính xuất hiện nhiều nhất.

**Lần 3:** 5000 dòng với 10000 thuộc tính xuất hiện nhiều nhất.

### Phân tích kết quả

* Naive bayes cho ra kết quả rất tốt với độ chính xác cao.

### Kết luận

## Hướng phát triển

### Tối ưu phương pháp

Trước hết, có thể có các biện pháp tối ưu quá trình tiền xử lý dữ liệu đầu vào như:

* Sử dụng phương pháp tách từ hiệu quả, toàn diện hơn kiểm tra các ký tự đặc biệt, rút ra quy luật dùng các dấu câu, chú ý đến từ loại của từ.
* Tìm hiểu sâu hơn về cấu trúc ngữ nghĩa của câu để có thể loại bỏ các từ như tên riêng, địa điểm, thời gian,… để giảm số lượng thuộc tính dữ liệu.
* Bags of Words là một phương pháp đơn giản nhưng có nhiều thiếu sót, có thể thay thế phương pháp trên bằng các phương pháp có phụ thuộc vào ngữ nghĩa của các từ để tăng độ chính xác dự đoán. ví dụ như word2vector, Glove, LSI,...

Tiếp theo về mô hình thuật toán:

* Naive bayes thường không được đánh giá cao về mặt lý thuyết nhưng trên thực tế nó lại cho ra kết quả rất tốt, có thể tiếp tục sử dụng naive bayes tùy vào mô hình dữ liệu vector đầu vào.
* Thay đổi thuật toán sang dùng Neuron network để phân lớp.

### Phát triển bài toán

Bài toán trên có rất nhiều ứng dụng trong thực tế, bằng việc thu thập nhiều dữ liệu tin tức trên các trang mạng xã hội để xây dựng ra bộ dữ liệu training càng lớn càng tốt, sau đó xây dựng model phân lớp từ các thuộc tính là từ trong cả bài tin tức có thể phân biệt được tin bài sai sự thật,…

# Tài liệu tham khảo

[1] https://machinelearningcoban.com/2017/08/08/nbc/