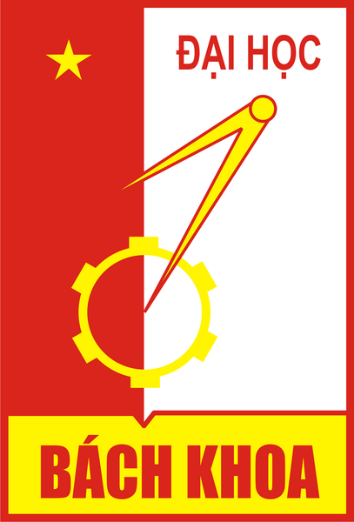
**ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**----------------------------oOo----------------------------**



**BÁO CÁO PROJECT III**

**Đề tài:**

TÌM HIỂU THUẬT TOÁN MACHINE LEARNING

PHÂN LOẠI BÀI BÁO THÀNH CÁC CHỦ ĐỀ CHO TRƯỚC

Giảng viên hướng dẫn: **Trịnh Văn Loan**

Sinh viên thực hiện: **Nguyễn Xuân Tân**

Mã số sinh viên: **20194370**

*Hà Nội* *- 2023*

# MỤC LỤC

[MỤC LỤC 1](#_Toc130810403)

[DANH MỤC HÌNH VẼ 2](#_Toc130810404)

[MỞ ĐẦU 3](#_Toc130810405)

[CHƯƠNG 1. MÔ HÌNH HỌC MÁY ĐỀ XUẤT 4](#_Toc130810406)

[1.1. K-NEAREST NEIGHBORS 4](#_Toc130810407)

[1.1.1. Cơ sở lý thuyết 4](#_Toc130810408)

[1.1.2. Tăng tốc KNN 5](#_Toc130810409)

[1.1.3. Vì sao chọn KNN cho bài toán 6](#_Toc130810410)

[1.2. Naïve Bayes 7](#_Toc130810411)

[1.2.1. Cơ sở lý thuyết 7](#_Toc130810412)

[1.2.2. Vì sao chọn Naïve Bayes cho bài toán 8](#_Toc130810413)

[1.3. Support vector machine – SVM 9](#_Toc130810414)

[1.3.1. Cơ sở lý thuyết 9](#_Toc130810415)

[1.3.2. Vì sao chọn SVM cho bài toán 12](#_Toc130810416)

[CHƯƠNG 2. CHUẨN BỊ DỮ LIỆU 13](#_Toc130810417)

[2.1. Dữ liệu 13](#_Toc130810418)

[2.2. Tiền xử lý dữ liệu 13](#_Toc130810419)

[2.3. Chuẩn hóa dữ liệu 14](#_Toc130810420)

[2.3.1. TF-IDF( Term frequency–inverse document frequency) 14](#_Toc130810421)

[2.3.2. SVD( Singular value decomposition) 15](#_Toc130810422)

[CHƯƠNG 3. KẾT QUẢ THỰC NGHIỆM 17](#_Toc130810423)

[3.1. K-NEAREST NEIGHBORS 18](#_Toc130810424)

[3.2. Naïve Bayes 19](#_Toc130810425)

[3.3. Support vector machine – SVM 20](#_Toc130810426)

[3.4. Kết luận 21](#_Toc130810427)

[KẾT LUẬN 22](#_Toc130810428)

# DANH MỤC HÌNH VẼ

[Figure 1. Mô tả ví dụ về bài toán phân lớp KNN( Nguồn: k-Nearest Neighbors (KNN) in Python - JC Chouinard) 5](#_Toc130808728)

[Figure 2. Hai lớp dữ liệu vuông và tròn là tách biệt tuyến tính.( Nguồn: GitHub - tiepvupsu/ebookMLCB: ebook Machine Learning cơ bản) 9](#_Toc130808729)

[Figure 3. Mô tả ví dụ về chia cách dữ liệu 2 lớp ( Nguồn: Giới thiệu về Support Vector Machine trong Machine Learning (noron.vn)) 10](#_Toc130808730)

[Figure 4. Mô tả ví dụ về chia cách dữ liệu 2 lớp ( Nguồn: Giới thiệu về Support Vector Machine trong Machine Learning (noron.vn)) 10](#_Toc130808731)

[Figure 5. Mô tả ví dụ về chia cách dữ liệu 2 lớp ảnh hưởng tham số C nhỏ( Nguồn: Giới thiệu về Support Vector Machine trong Machine Learning (noron.vn)) 11](#_Toc130808732)

[Figure 6. Mô tả ví dụ về chia cách dữ liệu 2 lớp ảnh hưởng tham số C lớn Nguồn: Giới thiệu về Support Vector Machine trong Machine Learning (noron.vn)) 12](#_Toc130808733)

[Figure 7. File bài báo chủ đề Du lịch. 13](#_Toc130808734)

[Figure 8. Ví dụ 1 file sau tiền xử lý dữ liệu 14](#_Toc130808735)

[Figure 9. Kết quả đánh giá mô hình KNN 18](#_Toc130808736)

[Figure 10. Confusion Matrix mô hình KNN 18](#_Toc130808737)

[Figure 11. Kết quả đánh giá mô hình Naïve Bayes 19](#_Toc130808738)

[Figure 12. Confusion Matrix mô hình Naïve Bayes 19](#_Toc130808739)

[Figure 13. Kết quả đánh giá mô hình SVM 20](#_Toc130808740)

[Figure 14. Confusion Matrix mô hình SVM 20](#_Toc130808741)

# MỞ ĐẦU

Học máy là môn khoa học nhằm phát triển những thuật toán và mô hình thống kê mà các hệ thống máy tính sử dụng để thực hiện các tác vụ dựa vào khuôn mẫu và suy luận mà không cần hướng dẫn cụ thể. Các hệ thống máy tính sử dụng thuật toán máy học để xử lý khối lượng lớn dữ liệu trong quá khứ và xác định các khuônmẫu dữ liệu. Việc này cho phép chúng dự đoán kết quả chính xác hơn từ cùng một tập dữ liệu đầu vào cho trước.

Một trong những bài toán phổ biến nhất của học máy là bài toán phân lớp. Với mục đích tìm hiểu, so sánh các mô hình và phân loại các lớp khác nhau, nhóm đã chọn đề tài “Phân loại bài báo thành các chủ đề cho trước” dựa trên bộ dữ liệu ̣thu thập từ các trang web tin tức. Các mô hình trên sẽ được áp dụng để kiểm tra bài toán này và kiểm tra độ hiệu quả thông qua các thực nghiệm.

Ứng dụng của bài toán phổ biến, chúng ta có thể sử dụng mô hình này để phân loại văn bản tin tức tiếng Việt cho một trang báo điện tử. Mỗi khi một bài báo được đăng, chương trình giúp tự động xác định được bài báo đó nằm trong chủ đề nào. Các chủ đề gồm có: thời sự, đời sống, giải trí, khoa học, kinh doanh, pháp luật, sức khỏe, thể thao, du lịch, giáo dục (10 chủ đề).

Quá trình gồm các giai đoạn:

Xử lý dữ liệu: Đây được coi là giai đoạn quan trọng nhất . Dữ liệu là đầu vào của các mô hình học máy để giải quyết bài toán. Tuy nhiên, những dữ liệu ban đầu khi mới thu thập chưa thể đưa vào mô hình do những thuộc tính dư thừa hay định dạng của một số thuộc tính không phù hợp với mô hình. Do đó, dữ liệu cần phải được chọn ra những đặc trưng tốt, xử lý những thông tin bị thiếu hoặc thay đổi định dạng của dữ liệu. Trong quá trình này, chúng ta cũng cần phải thực hiện phân chia dữ liệu thành các tập khác nhau bao gồm: tập huấn luyện (training set), tập xác thực (validation set) và tập kiểm thử (test set) với mục đích đo đạc, chọn ra mô hình với thông số phù hợp và kiểm tra độ chính xác của mô hình trong thực tế.

Xây dựng mô hình và điều chỉnh tham số: Mục đích của bước này là xây dựng các mô hình cho việc huấn luyện. Các tham số của các mô hình sẽ được điều chỉnh phù hợp dựa trên kết quả học của tập huấn luyện và tập xác thực. Sau đó, các mô hình tốt nhất đại diện cho mỗi thuật toán sẽ được đưa vào đánh giá hiệu quả trên tập kiểm thử.

Các quá trình trên sẽ được trình bày rõ hơn trong các phần sau.

# MÔ HÌNH HỌC MÁY ĐỀ XUẤT

## K-NEAREST NEIGHBORS

### Cơ sở lý thuyết

#### Một số thuộc tính quan trọng trong KNN

- N\_neighbors: Số điểm lân cận được xét trong KNN

- Weights: Cách đánh trọng số của các điểm dữ liệu, có 2 cách phổ biến :

- Uniform: Tất cả các điểm trong mỗi vùng lân cận đều có trọng số như nhau

- Distance: Trọng số bằng nghịch đảo của khoảng cách của chúng. trong trường hợp này, các hàng xóm gần điểm truy vấn hơn sẽ có ảnh hưởng lớn hơn các hàng xóm ở xa hơn. Ngoài ra chúng ta có thể tự tạo 1 hàm tính trọng số của riêng mình

- Metrics : Cách tính khoảng cách

+ Minkowski:

+ Manhattan:

+ Euclidian: Là trường hợp đặc biệt của Minkowski khi p=2

+ Chebyshev:

|  |  |
| --- | --- |
| **Distance** | **Equation** |
| Euclidean | d = |
| City Block | d = |
| Chebyshev | d = |
| Cosine | d = 1 - |
| Corelation | d = 1 - |

Có 1 lưu ý rất quan trọng đó là KNN không có hàm mất mát và hàm tối ưu, vậy

nên mô hình hoạt động ra sao phụ thuộc tất cả vào các tham số ở trên.

#### KNN cho bài toán Classfication

* Biểu diễn dữ liệu:
  + - * Mỗi đối tượng quan sát được thể hiện bởi một vector n chiều, mỗi chiều đại diện cho một thuộc tính đối tượng. Ví dụ
* Có tập C là tập các nhãn đã được gán trước
  + Giai đoạn học: Lưu tập dữ liệu huấn luyện D và các nhãn của nó
  + Dự đoán đầu vào mới z
* Mỗi dữ liệu x trong D, tính khoảng cách tương đồng của nó với z
* Tìm tập NB(z) là tập các hàng xóm gần với z nhất
* Sử dụng các nhãn của các dữ liệu trong NB(z) để dự đoán z

Hình là mô tả ví dụ về bài toán phân lớp. Trong đó, cần phải xác định điểm dữ liệu “sao” là màu xanh lá cây hay xanh dương

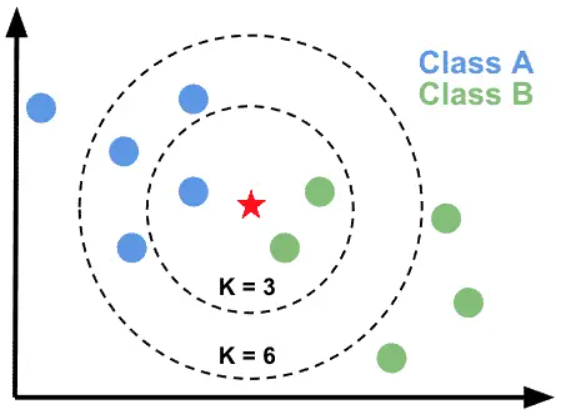


Figure . Mô tả ví dụ về bài toán phân lớp KNN( Nguồn: [k-Nearest Neighbors (KNN) in Python - JC Chouinard](https://www.jcchouinard.com/k-nearest-neighbors/))

### Tăng tốc KNN

Ngoài việc tính toán khoảng cách từ một điểm test data đến tất cả các điểm trong traing set (Brute Force), có một số thuật toán khác giúp tăng tốc việc tìm kiếm này.

#### Kd – tree (K-dimensional tree)

Mỗi cấp độ của cây Kd chia tất cả con dọc theo một chiều cụ thể bằng một siêu mặt phẳng vuông góc với trục tương ứng. Ở gốc của cây, tất cả các con sẽ được phân chia dựa trên chiều thứ nhất: nếu tọa độ chiều thứ nhất nhỏ hơn gốc, nó sẽ ở trong cây phụ bên trái và nếu nó lớn hơn gốc thì rõ ràng nó sẽ ở trong cây phụ bên phải. Mỗi cấp độ trong cây phân chia theo chiều không gian tiếp theo, trở về chiều không gian đầu tiên khi tất cả các chiều khác đã được xem xét.

Từ tree ta có thể tìm neighbour của một điểm cho trước như sau: Bắt đầu root node, move qua các nhánh một cách đệ quy. Trong khi di chuyển nó sẽ check xem, với điểm current best ở nhánh trái thì distance best (khoảng cách từ target point tới current best) có ngắn hơn khoảng cách từ target point tới bờ phân chia hay không, nếu ngắn hơn tức là bên nhánh trái đã cho kết quả tốt nhất và ta không cần tìm tiếp bên phải, nếu dài hơn tức là có lẽ sẽ có 1 điểm nào đó bên phải cho khoảng cách tới target point tốt hơn nên ta phải tiếp tục lặp qua các nhánh bên phải.

#### Ball - tree

Cây bóng là một cây nhị phân trong đó mỗi nút xác định một quả bóng chiều D chứa một tập hợp con của các điểm cần tìm kiếm. Mỗi nút bên trong của cây phân vùng các điểm dữ liệu thành hai tập hợp rời rạc được liên kết với các quả bóng khác nhau. Trong khi các quả bóng có thể giao nhau, mỗi điểm được gán cho một hoặc quả bóng khác trong phân vùng theo khoảng cách của nó từ tâm của quả bóng. Mỗi nút lá trong cây xác định một quả bóng và liệt kê tất cả các điểm dữ liệu bên trong quả bóng đó.

Mỗi nút trong cây xác định quả bóng nhỏ nhất chứa tất cả các điểm dữ liệu trong cây con của nó. Điều này làm phát sinh tính chất hữu ích rằng, đối với một điểm kiểm tra nhất định t bên ngoài quả bóng, khoảng cách đến bất kỳ điểm nào trong quả bóng B trên cây lớn hơn hoặc bằng khoảng cách từ t đến bề mặt của quả bóng.

Cây bóng đã được chứng minh là hoạt động khá tốt vấn đề tìm kiếm hàng xóm gần nhất, đặc biệt là khi số lượng kích thước của chúng tăng lên. Tuy nhiên, cấu trúc dữ liệu lân cận gần nhất tốt nhất cho một ứng dụng nhất định sẽ phụ thuộc vào tính thứ nguyên, số lượng điểm dữ liệu và cấu trúc cơ bản của dữ liệu.

### Vì sao chọn KNN cho bài toán

Thứ nhất, với một bài toán Classfication, KNN hoạt động rất đơn giản vì nó chỉ là việc tìm kiếm các điểm dữ liệu gần nhất trong tập dữ liệu huấn luyện và sử dụng chúng để phân loại điểm dữ liệu mới, ngoài ra không yêu cầu độ dốc tuyến tính và không cần tính toán đạo hàm.

Thứ hai, đây là một thuật toán có khả năng định vị dữ liệu rất tốt thông qua các vector và tọa độ trên một không gian n-chiều, chính vì vậy chúng ta có thể tìm ra được dữ liệu của những mẫu nào có sự tương quan với nhau nhiều và sự tương quan ít hơn. Vì vậy, KNN có thể là một lựa chọn hiệu quả trong các bài toán phân loại cơ bản với tập dữ liệu nhỏ và đơn giản.

## Naïve Bayes

### Cơ sở lý thuyết

Xét các bài toán phân loại với C nhãn khác nhau. Thay vì tìm ra chính xác nhãn của mỗi điểm dữ liệu x ∈ , ta có thể đi tìm xác suất để kết quả rơi vào mỗi nhãn: p(y = c|x), hoặc viết gọn thành p(c|x). Biểu thức này được hiểu là xác suất để đầu ra là nhãn c biết rằng đầu vào là vector x. Nếu tính được biểu thức này, ta có thể giúp xác định nhãn của mỗi điểm dữ liệu bằng cách chọn ra nhãn có xác suất rơi vào cao nhất:

c =

Nhìn chung, khó có cách tính trực tiếp p(c|x). Thay vào đó, quy tắc Bayes thường được sử dụng:

c = =

Tiếp tục quan sát, p(c) có thể được hiểu là xác suất để một điểm bất kỳ rơi vào nhãn c. Nếu tập huấn luyện lớn, p(c) có thể được xác định bằng phương pháp ước lượng hợp lý cực đại (MLE) – là tỉ lệ giữa số điểm thuộc nhãn c và số điểm trong tập huấn luyện. Nếu tập huấn luyện nhỏ, giá trị này có thể được xác định bằng phương pháp ước lượng hậu nghiệm cực đại (MAP)

Thành phần còn lại p(x|c) là phân phối của các điểm dữ liệu trong nhãn c. Thành phần này thường rất khó tính toán vì x là một biến ngẫu nhiên nhiều chiều. Để có thể ước lượng được phân phối đó, tập huấn luyện phải rất lớn. Nhằm đơn giản hoá việc tính toán, người ta thường giả sử rằng các thành phần của biến ngẫu nhiên x độc lập với nhau khi đã biết c:

**Ở bước huấn luyện**, các phân phối p(c) và p(|c), i = 1, . . . , d được xác định dựa vào dữ liệu huấn luyện. Việc xác định các giá trị này có thể được thực hiện bằng MLE hoặc MAP.

**Ở bước kiểm tra**, nhãn của một điểm dữ liệu mới x được xác đinh bởi:

c =

Khi d lớn và các xác suất nhỏ, biểu thức ở vế phải là một số rất nhỏ, khi tính toán có thể gặp sai số. Để giải quyết việc này, công thức thường được viết lại dưới dạng tương đương bằng cách lấy log của vế phải:

c =

Phân phối thường dùng trong NBC

**Multinomial naïve Bayes**

Mô hình này chủ yếu được sử dụng trong bài toán phân loại văn bản mà vector đặc trưng được xây dựng dựa trên ý tưởng bag of words (BoW). Lúc này, mỗi văn bản được biểu diễn bởi một vector có độ dài d là số từ trong từ điển. Giá trị của thành phần thứ i trong mỗi vector là số lần từ thứ i xuất hiện trong văn bản đó. Khi đó, tỉ lệ với tần suất từ thứ i (hay đặc trưng thứ i trong trường hợp tổng quát) xuất hiện trong các văn bản có nhãn c. Giá trị này có thể được tính bởi

Trong đó:

• là tổng số lần từ thứ i xuất hiện trong các văn bản của nhãn c. Nó chính là tổng tất cả thành phần thứ i của các vector đặc trưng ứng với nhãn c.

• là tổng số từ, kể cả lặp, xuất hiện trong nhãn c. Nói cách khác, Nc là tổng độ dài của tất cả các văn bản thuộc nhãn c. Có thể suy ra rằng , từ đó = 1

Cách tính này có một hạn chế là nếu có một từ mới chưa bao giờ xuất hiện trong

nhãn c thì biểu thức (11.7) sẽ bằng không, dẫn đến vế phải của (11.4) bằng không

bất kể các giá trị còn lại lớn thế nào (xem thêm ví dụ ở mục sau). Để giải quyết

việc này, một kỹ thuật được gọi là làm mềm Laplace (Laplace smoothing) được

áp dụng:

với α là một số dương, thường bằng 1, để tránh trường hợp tử số bằng không.

Mẫu số được cộng với dα để đảm bảo tổng xác suất . Như vậy, mỗi

nhãn c được mô tả bởi một bộ các số dương có tổng bằng 1: ={}

### Vì sao chọn Naïve Bayes cho bài toán

Sự khác nhau của các mô hình phân loại navie Bayes chủ yếu dựa trên các giả định liên quan đến xác suất của . Mặc dù những giả định được đơn giản hóa quá mức, nhưng phân lạo navie Bayes vẫn hoạt động khá tốt trong nhiều tình huống thực tế, điển hình là phân loại tài liệu và đọc thư rác. Cả quá trình huấn luyện và kiểm tra của NBC đều cực kỳ nhanh so với các phương pháp phân loại phức tạp khác. Việc giả sử các thành phần trong dữ liệu là độc lập với nhau khiến cho việc tính toán mỗi phân phối trở nên đơn giản.

## Support vector machine – SVM

### Cơ sở lý thuyết

Một máy vectơ hỗ trợ xây dựng một siêu phẳng hoặc một tập hợp các siêu phẳng trong một không gian nhiều chiều hoặc vô hạn chiều, có thể được sử dụng cho phân loại, hồi quy, hoặc các nhiệm vụ khác. Một cách trực giác, để phân loại tốt nhất thì các siêu phẳng nằm ở càng xa các điểm dữ liệu của tất cả các lớp (gọi là hàm Biên) càng tốt, vì nói chung Biên càng lớn thì sai số tổng quát hóa của thuật toán phân loại càng bé.

Ví dụ, ta có các điểm dữ liệu như hình dưới đây với mỗi điểm thuộc 1 trong 2 lớp cho trước:

Diagram

Description automatically generated

Figure . Hai lớp dữ liệu vuông và tròn là tách biệt tuyến tính.( Nguồn: [GitHub - tiepvupsu/ebookMLCB: ebook Machine Learning cơ bản](https://github.com/tiepvupsu/ebookMLCB))

Có vô số đường thằng có thể phân loại chính xác hai lớp dữ liệu này.

Ta có, khoảng cách từ một điểm (vector) có toạ độ () tới siêu phẳng = 0 được xác định bởi

Ví dụ trên có các điểm dữ liệu thuộc 2 lớp có thể phân cách trực tiếp bằng nhiều đường thẳng. Tuy nhiên, không phải dạng dữ liệu nào cũng đơn giản được như vậy. Ví dụ với dạng dữ liệu như hình dưới:

Chart, icon, scatter chart

Description automatically generated

Figure . Mô tả ví dụ về chia cách dữ liệu 2 lớp ( Nguồn: [Giới thiệu về Support Vector Machine trong Machine Learning (noron.vn)](https://www.noron.vn/post/gioi-thieu-ve-support-vector-machine-trong-machine-learning-40dxtjcmrdye))

Với các điểm dữ liệu thế này thì không thể phân cách chỉ bằng 1 đường thẳng. Các điểm dữ liệu của lớp hình tròn nằm tập trung ở xung quanh gốc tọa độ và để phân cách chúng với các điểm dữ liệu của lớp hình vuông thì phải sử dụng đường cong. Trong trường hợp này, đường cong đơn giản nhất chúng ta có thể nghĩ đến là hình tròn. Ta áp dụng điều này bằng cách thêm 1 chiều không gian

z = . Nếu các bạn để ý thì sẽ thấy z chính là công thức của 1 hình tròn có tâm tại gốc tọa độ. Ta sẽ thử vẽ lại đồ thị của các điểm dữ liệu trên không gian zy:

Chart, scatter chart

Description automatically generated

Figure . Mô tả ví dụ về chia cách dữ liệu 2 lớp ( Nguồn: [Giới thiệu về Support Vector Machine trong Machine Learning (noron.vn)](https://www.noron.vn/post/gioi-thieu-ve-support-vector-machine-trong-machine-learning-40dxtjcmrdye))

Tại không gian này, các điểm dữ liệu đã có thể phân cách bằng 1 đường thẳng. Nói cách khác, với chiều không gian mới z, việc phân loại đã trở nên đơn giản hơn rất nhiều. Việc sử dụng thủ thuật biến đổi đại số và thêm chiều không gian z được gọi là “phương pháp kernels”. Với các tập dữ liệu khác nhau, bạn có thể sử dụng các kernels có sẵn hoặc sẽ phải sáng tạo ra 1 kernel mới để phù hợp với bài toán của mình.

**Điều chinh các tham số: Kernel, Regularization, Gamma và Margin**

* **Kernel**

Quá trình xây dựng siêu phẳng phân cách trong SVM được thực hiện qua các phép biển đổi đại số. Với kernel dạng tuyến tính (linear kernel) công thức dùng để dự đoán những điểm dữ liệu mới là: thực hiện tích vô hướng giữa đầu vào (x) với mỗi support vector () như sau: f(x) = B(0) + sum(ai (x, )). Các hệ số B0 và ai (cho mỗi đầu vào) phải được ước tính từ dữ liệu học. Với kernel dạng đa thức (polynomial kernel) có thể được viết dưới dạng:

K(x, ) = 1 +.

Còn với kernel dạng lũy thừa (exponential kernel) có dạng:

K(x, ) = exp(-gamma \* sum((x - )).

Kernel dạng đa thức và dạng lũy thừa tính toán đường phân cách ở những chiều không gian cao hơn và được gọi là kernel trick.

* **Regularization**

Tham số Regularization ( được nhắc đến trong thư viên sklearn là tham số C) điều chỉnh việc có nên bỏ qua các điểm dữ liệu bất thường trong quá trình tối ưu mô hình SVM. Nếu tham số này có giá trị lớn, quá trình tối ưu sẽ chọn một siêu phẳng sao cho siêu phẳng này phân cách tất cả các điểm dữ liệu một cách tốt nhất, từ đó khoảng cách giữa siêu phẳng tới các điểm dữ liệu của 2 lớp sẽ có giá trị nhỏ (small-margin). Ngược lại, khi tham số này có giá trị nhỏ, siêu phẳng sẽ được xây dựng sao cho khoảng cách với các điểm dữ liệu của 2 lớp có giá trị lớn (large-margin), kể cả khi siêu phẳng này sẽ phân loại sai nhiều điểm dữ liệu hơn.

Dưới đây là các ví dụ về 2 trường hợp chọn tham số C:

Tham số C có giá trị nhỏ

Chart, scatter chart

Description automatically generated

Figure . Mô tả ví dụ về chia cách dữ liệu 2 lớp ảnh hưởng tham số C nhỏ( Nguồn: [Giới thiệu về Support Vector Machine trong Machine Learning (noron.vn)](https://www.noron.vn/post/gioi-thieu-ve-support-vector-machine-trong-machine-learning-40dxtjcmrdye))

Tham số C có giá trị lớn

Chart, scatter chart

Description automatically generated

Figure . Mô tả ví dụ về chia cách dữ liệu 2 lớp ảnh hưởng tham số C lớn Nguồn: [Giới thiệu về Support Vector Machine trong Machine Learning (noron.vn)](https://www.noron.vn/post/gioi-thieu-ve-support-vector-machine-trong-machine-learning-40dxtjcmrdye))

* **Gamma**

Tham số gamma xác định việc sử dụng bao nhiêu điểm dữ liệu cho việc xây dựng siêu phẳng phân cách. Với giá trị gamma nhỏ, các điểm dữ liệu nằm xa đường phân cách sẽ được sử dụng trong việc tính toán đường phân cách. Ngược lại, với giá trị gamma lớn, chỉ những điểm nằm gần đường phân cách mới được sử dụng để tính toán.

* **Margin**

Margin trong SVM là khoảng cách giữa siêu phẳng phân cách với các điểm dữ liệu gần nó nhất. Khoảng cách này đối với các điểm dữ liệu gần nhất của cả 2 lớp càng lớn thì mô hình càng phân loại chính xác. Các ví dụ về margin:

SVM có margin tốt : khoảng cách lớn và cân bằng giữa siêu phẳng và các điểm dữ liệu của 2 lớp

SVM có margin tồi: khoảng cách không cân bằng & nghiêng hẳn về 1 phía

### Vì sao chọn SVM cho bài toán

Là một kĩ thuật phân lớp khá phổ biến, SVM thể hiện được nhiều ưu điểm trong số đó có việc tính toán hiệu quả trên các tập dữ liệu lớn.

* 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.

# CHUẨN BỊ DỮ LIỆU

## Dữ liệu

Dữ liệu các bài báo được lấy từ trang [Báo Tuổi Trẻ - Tin tức mới nhất, tin nhanh, tin nóng 24h (tuoitre.vn)](https://tuoitre.vn/) với nội dung từ 10 chủ đề: Du lịch, Giải trí, Giáo dục, Kinh doanh, Pháp luật, Sức khỏe, Thế giới, Thể thao, Văn hóa, Xe được lưu trữ dưới dạng file text.

Đã lưu trữ được:

* 2556 bài báo chủ đề Du lịch
* 4151 bài báo chủ đề Giải trí
* 3626 bài báo chủ đề Giáo dục
* 3329 bài báo chủ đề Kinh doanh
* 3723 bài báo chủ đề Pháp luật
* 4002 bài báo chủ đề Sức khỏe
* 3178 bài báo chủ đề Thế giới
* 4308 bài báo chủ đề Thể thao
* 3253 bài báo chủ đề Văn hóa
* 3222 bài báo chủ đề Xe

Ví dụ 1 file crawl được:

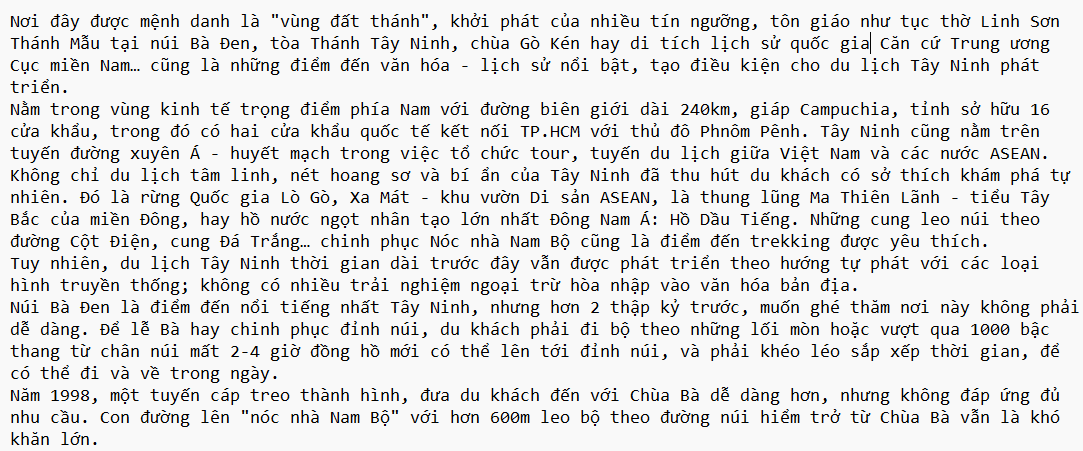


Figure . File bài báo chủ đề Du lịch.

## Tiền xử lý dữ liệu

* Vấn đề chính bây giờ là dữ liệu đang ở định dạng văn bản (string). Thuật toán phân loại (classification) chỉ hoạt động dưới dạng dữ liệu số hóa (numeric), từ đó yêu cầu phải chuyển đổi dữ liệu đầu vào trên (chuỗi kí tự) thành vector (chuỗi số) để các thuật toán học máy có thể hiểu được
* Tách từ trong câu thành từ đơn, từ ghép nhằm tăng ý nghĩa của từ, giảm features. Dùng thư viện underthesea
* Chuyển các số phần trăm (vd 20%, 10%, 0.5%...) thành 'percents' nhằm giảm features
* Xóa các kí tự
* Chuyển các số thành 'numbers', ngày tháng thành 'days' nhằm giảm features
* Loại bỏ stopword

Ví dụ 1 file sau khi được xử lý:

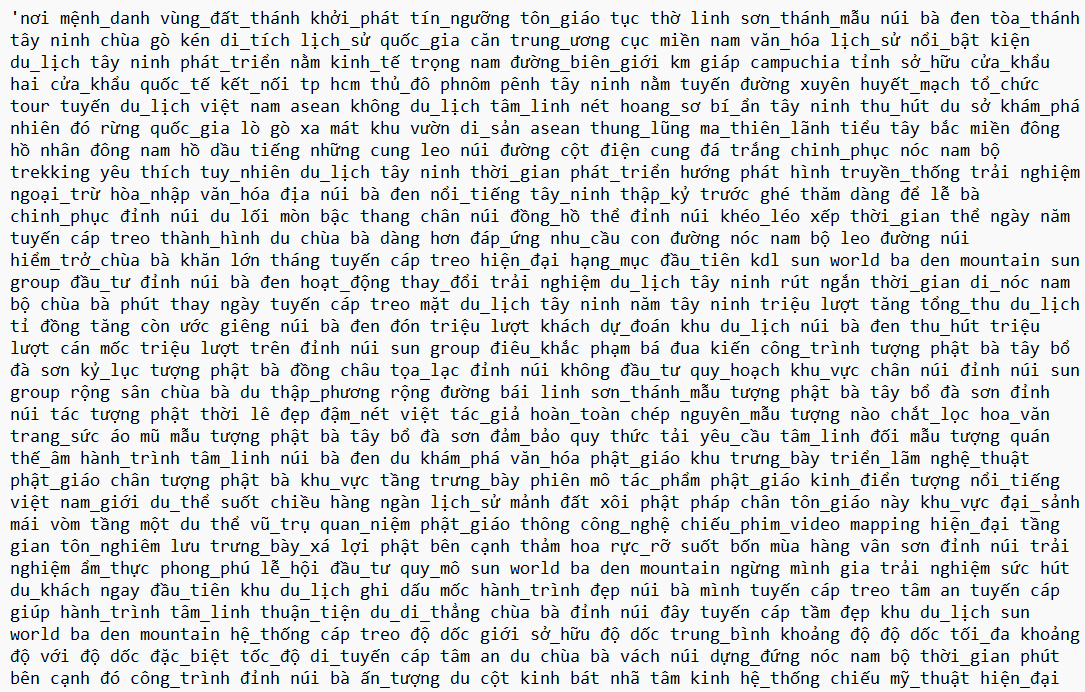


Figure . Ví dụ 1 file sau tiền xử lý dữ liệu

## Chuẩn hóa dữ liệu

### TF-IDF( Term frequency–inverse document frequency)

Bây giờ các văn bản đã được mã hóa, chúng ta cần chuyển đổi chúng thành các vector để các thuật toán học máy trong SciKit Learn có thể làm việc được.

Trong truy hồi thông tin, TF-IDF (term frequency–inverse document frequency) là một thống kê số học nhằm phản ánh tầm quan trọng của một từ đối với một văn bản trong một tập hợp hay một ngữ liệu văn bản. TF-IDF thường dùng dưới dạng là một trọng số trong tìm kiếm truy xuất thông tin, khai thác văn bản, và mô hình hóa người dùng.

Giá trị TF-IDF tăng tỉ lệ thuận với số lần xuất hiện của một từ trong tài liệu và được bù đắp bởi số lượng tài liệu trong kho ngữ liệu có chứa từ, giúp điều chỉnh thực tế là một số từ xuất hiện nói chung thường xuyên hơn. TF-IDF là một trong những lược đồ (scheme) tính trọng số phổ biến nhất hiện nay.

1. TF-term frequency - tần số xuất hiện của 1 từ trong 1 văn bản. Cách tính như sau:

tf(t, d) =

* Kết quả thuộc khoảng [0,1], thuơng của số lần xuất hiện 1 từ trong văn bản và số lần xuất hiện nhiều nhất của 1 từ trong văn bản đó
* f(t, d): Tần suất của từ t trong văn bản
* max( f(w, d) : w ϵ d): số lần xuất hiện nhiều nhất của 1 từ trong d

1. IDF - Inverse document frequency - Tần số nghịch của một từ trong tập văn bản (corpus). Tính IDF để giảm giá trị của những từ phổ biến. Mỗi từ chỉ có 1 IDF duy nhất trong tập văn bản. Cách tính như sau:

idf(t, D) = log

* |D|: Tổng số văn bản trong D
* |{d ϵ D: t ϵ d}|: Số văn bản chứa từ nhất định, với điều kiện xuất hiện trong văn bản d

Cơ số logarit không thay đổi giá trị của 1 từ mà chỉ thu hẹp khoảng giá trị của từ đó (thay đổi cơ số không ảnh hưởng tới tỷ lệ giữa các IDF). Tuy nhiên thay đổi khoảng giá trị sẽ giúp tỷ lệ IDF và TF tương đồng để dùng cho công thức TF-IDF như bên dưới.



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 (từ khoá của văn bản đó).

**Ứng dụng**

IDF có ứng dụng phổ biến nhất trong tìm kiếm. Ví dụ như chúng ta muốn mua hàng ở shopee, chúng ta thực hiện truy vấn: "Những món quà đẹp tặng người yêu ngày valentine". Sau khi tách từ, những từ "người yêu" và "valentine" chắc chắn là những có idf cao nhất trong văn bản trên, từ đó hệ thống sẽ coi những từ trên là từ khóa, lấy tất cả các văn bản có chưa từ "người yêu" và "valentine" sau đó thực hiện đánh giá và so sánh dựa trên bộ truy vấn

### SVD( Singular value decomposition)

Sau khi thực hiện TF-IDF, chúng ta dễ dàng nhận thấy rằng, ma trận mà chúng ta thu được có kích thước rất lớn, và việc xử lý tính toán với ma trận này đòi hỏi thời gian và bộ nhớ khá tốn kém.

Để xử lý vấn đề này, chúng ta sẽ sử dụng thuật toán SVD (singular value decomposition) nhằm mục đích giảm chiều dữ liệu của ma trận mà chúng ta thu được, mà vẫn giữ nguyên được các thuộc tính của ma trận gốc ban đầu.

SVD (Singular Value Decomposition) là một phương pháp để phân tích một ma trận bất kỳ thành tích của ba ma trận thành phần.

Một ma trận (có m hàng và n cột) bất kỳ đều có thể phân tích thành dạng:

Trong đó:

* là ma trận trực giao cấp m
* là ma trận trực giao cấp n
* là ma trận đường chéo không vuông với các phần tử trên đường chéo σ1≥σ2≥⋯≥σr≥0 với r là rank của ma trận A. Trong đó σ1, σ2,…, σr là các trị riêng của ma trận A.

**Truncated SVD**

Chú ý rằng trong ma trận , các giá trị trên đường chéo là không âm và giảm dần σ1≥σ2≥⋯≥σr≥0. Thông thường, chỉ một lượng nhỏ các σi mang giá trị lớn, các giá trị còn lại thường nhỏ và gần 0. Khi đó ta có thể xấp xỉ ma trận

A bằng tổng của k < r ma trận có rank 1:

A ≈ =

Ta có định lý:

=

Định lý này nói rằng sai số do cách xấp xỉ trên chính là căn bậc hai của tổng bình phương của các singular values mà ta đã bỏ qua ở phần cuối của S. Như vậy, sai số do xấp xỉ càng nhỏ nếu phần singular values bị truncated có giá trị càng nhỏ so với phần singular values được giữ lại.

# KẾT QUẢ THỰC NGHIỆM

Khi xây dựng một mô hình Machine Learning, chúng ta cần một phép đánh giá để xem mô hình sử dụng có hiệu quả không và để so sánh khả năng của các mô hình. Em sẽ sử dụng các phương pháp đánh giá sau:

**Accuracy( độ chính xác) :**

Tỉ lệ giữa số điểm được dự đoán đúng và tổng số điểm trong tập dữ liệu kiểm thử.

**Confusion matrix:**

Một ma trận chỉ ra được cụ thể mỗi loại được phân loại như thế nào, lớp nào được phân loại đúng nhiều nhất, và dữ liệu thuộc lớp nào thường bị phân loại nhầm vào lớp khác.

**True/False Positive/Negative:**

* True Positive (TP): số lượng điểm của lớp positive được phân loại đúng là positive.
* True Negative (TN): số lượng điểm của lớp negative được phân loại đúng là negative.
* False Positive (FP): số lượng điểm của lớp negative bị phân loại nhầm thành positive.
* False Negative (FN): số lượng điểm của lớp positiv bị phân loại nhầm thành negative

**Precision và Recall:**

* Precision =
* Recall =
* Precision cao đồng nghĩa với việc độ chính xác của các điểm tìm được là cao. Recall cao đồng nghĩa với việc True Positive Rate cao, tức tỉ lệ bỏ sót các điểm thực sự positive là thấp.

**-score:**

-score có giá trị nằm trong nửa khoảng (0, 1]. càng cao, bộ phân lớp càng tốt. Khi cả recall và precision đều bằng 1 (tốt nhất có thể), . Khi cả recall và precision đều thấp, ví dụ bằng 0.1, 0.1

**Macro-average:**

Macro-average precision là trung bình cộng của các precision theo class, tương tự với Macro-average recall.

## K-NEAREST NEIGHBORS

Sau các quá trình tiền xử lý và véc-tơ hóa, tập dữ liệu các bài báo được huấn luyện với giải thuật phân loại văn bản tự động K-NEAREST NEIGHBORS ta thu được như sau:

Table

Description automatically generated with low confidence

Figure . Kết quả đánh giá mô hình KNN

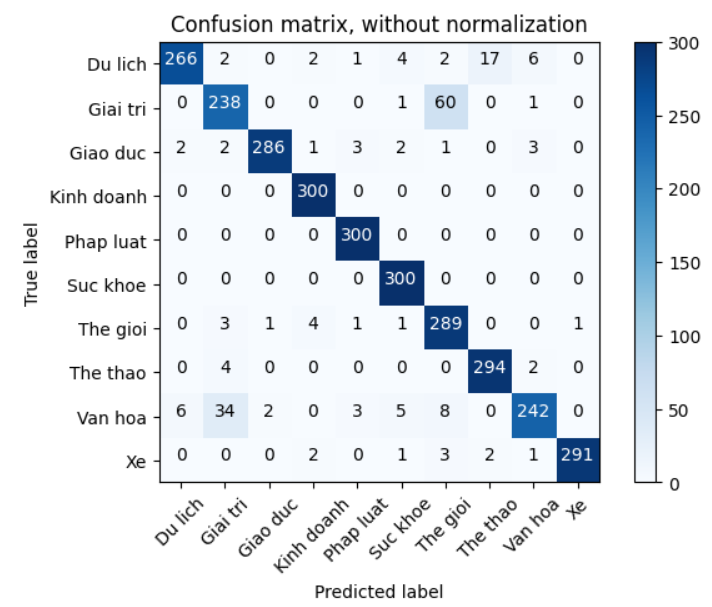


Figure . Confusion Matrix mô hình KNN

## Naïve Bayes

Sau các quá trình tiền xử lý và véc-tơ hóa, tập dữ liệu các bài báo được huấn luyện với giải thuật phân loại văn bản tự động Naïve Bayes ta thu được như sau:

Calendar

Description automatically generated with low confidence

Figure . Kết quả đánh giá mô hình Naïve Bayes

Chart, scatter chart

Description automatically generated

Figure . Confusion Matrix mô hình Naïve Bayes

## Support vector machine – SVM

Sau các quá trình tiền xử lý và véc-tơ hóa, tập dữ liệu các bài báo được huấn luyện với giải thuật phân loại văn bản tự động Support vector machine – SVM ta thu được như sau:

Calendar

Description automatically generated

Figure . Kết quả đánh giá mô hình SVM

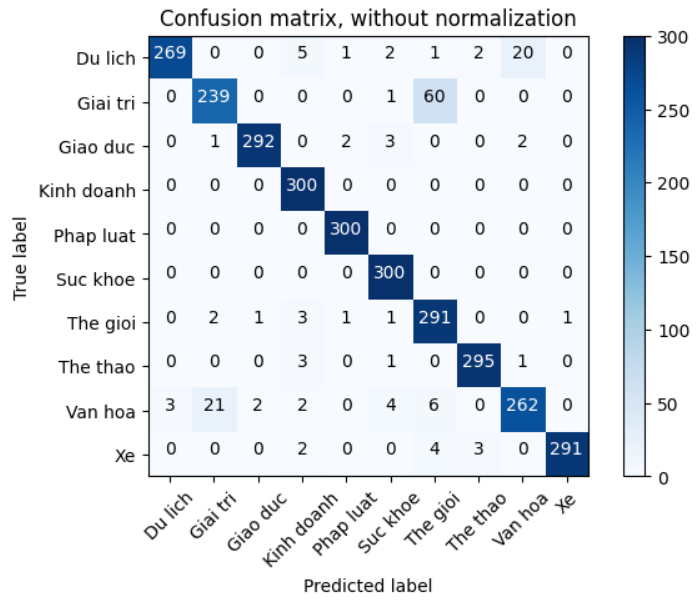


Figure . Confusion Matrix mô hình SVM

## Kết luận

Kết quả thực nghiệm cho thấy hiệu quả phân loại của các giải thuật là tương đối tốt. Trong đó, giải thuật SVM có kết quả phân loại tốt nhất, cho độ chính xác > 95%, rất khả thi cho việc xây dựng hệ thống tự động phân loại bài báo; góp phần giúp cho quá trình phân loại bài báo của tác giả và ban biên tập được nhanh và chính xác hơn.

# KẾT LUẬN

Bài báo cáo đã hoàn thành được các mục tiêu đặt ra là nghiên cứu và tìm hiểu về

Trong quá trình hoàn thành báo cáo này, em có cơ hội tìm hiểu được nhiều kiến thức mới bổ ích, phát triển một số kĩ năng nghiên cứu các bài báo.

Cuối cùng, em xin chân thành cảm ơn thầy Trịnh Văn Loan đã giúp đỡ, nhận xét, đánh giá và đưa ra lời khuyên để em có thể hoàn thiện báo cáo một cách tốt nhất.

TÀI LIỆ [1]U THAM KHẢo

|  |  |
| --- | --- |
| [1] | V.H.Tiệp,"K-nearest neighbors".*https://machinelearningcoban.com/2017/01/08/knn/.* |
| [2] | V. H. Tiệp, Machine Learning cơ bản, 2020. |
| [3] | https://viblo.asia/p/phan-loai-van-ban-tu-dong-bang-machine-learning-nhu-the-nao-4P856Pa1ZY3. |
| [4] | N. T. Hậu, "Phân loại văn bản tự động bằng Machine Learning như thế nào?".*https://viblo.asia/p/phan-loai-van-ban-tu-dong-bang-machine-learning-nhu-the-nao-4P856Pa1ZY3.* |
| [5] | N. V. Hùng, "Trích chọn thuộc tính trong đoạn văn bản với TF-IDF".*https://viblo.asia/p/trich-chon-thuoc-tinh-trong-doan-van-ban-voi-tf-idf-Az45bAOqlxY.* |
| [6] | T.Q.Khoát,"MachineLearningandData Mining".*https://bkai.ai/course/machine-learning-and-data-mining/?fbclid=IwAR2uXsJ78BwzW2A-hUuIGAKcRQxShvbpa7dbFG7qByacahcDLddR2fqsKNI.* |
| [7] | N. H. Nam, "Giới thiệu về Support Vector Machine trong Machine Learning".*https://www.noron.vn/post/gioi-thieu-ve-support-vector-machine-trong-machine-learning-40dxtjcmrdye.* |
| [8] | Trần Thanh Điện, Thái Nhựt Thanh, Nguyễn Thái Nghe, "GIẢI PHÁP PHÂN LOẠI BÀI BÁO KHOA HỌC BẰNG KỸ THUẬT MÁY HỌC". |