Báo cáo Xử lý ngôn ngữ tự nhiên

công nghệ thông tin (Trường Đại học Ngoại ngữ Tin học Thành phố Hồ Chí Minh)

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÁO CÁO BÀI TẬP LỚN
XỬ LÝ NGÔN NGỮ TỰ NHIÊN

ĐỀ TÀI:

Phân loại văn bản sử dụng NaiveBayes và SVM

GIẢNG VIÊN HƯỚNG DẪN:
PGS.TS Lê Thanh Hương
SINH VIÊN THỰC HIỆN:
Ngô Vân Anh - 20150084
Tô Hương Giang - 20151110
Nguyễn Hoàng Giang - 20151094

Hà Nội, tháng 5 năm 2019

MỤC LỤC

LỜI NÓI ĐẦU

Ngày nay, với sự phát triển vượt bậc của khoa học và công nghệ, đặc biệt là sự bùng nổ của Internet với các phương tiện truyền thông xã hội, thương mại điện tử,... đã cho phép mọi người tìm chia sẻ, tìm kiếm thông tin. Vì vậy mà Internet đã trở lên vô cùng quan trọng và là nguồn cung cấp một lượng thông tin vô cùng lớn và quan trọng.

Phân loại văn bản là một vấn đề quan trọng trong lĩnh vực xử lý ngôn ngữ. Nhiệm vụ của bài toán này là gán các tài liệu văn bản nào đó vào nhóm các chủ đề cho trước. Đây là một bài toán thường gặp trong thực tế điển hình như một chuyên gia phân tích thị trường chứng khoán, anh ta cần phải tổng hợp rất nhiều tài liệu, bài viết về thị trường chứng khoán để đọc và đưa ra phán đoán của mình. Tuy nhiên anh ta không thể đọc tất cả các tài liệu, bài báo để rồi phân loại chúng đâu là tài liệu chứng khoán mà anh ta cần. Lý do của vấn đề này là bởi vì số lượng bài báo, bài viết hiện nay rất nhiều, đặc biệt là trên internet, nếu đọc được hết tất cả các tài liệu thì sẽ mất rất nhiều thời gian. Một ví dụ khác trong thực tế là việc phân loại spam mail. Khi một mail được gửi đến hộp thư, nếu để người dùng phải đọc tất cả các mail thì sẽ tốn thời gian vì spam mail rất nhiều. Vì vậy cần phải phân loại đâu là spam mail và đâu là mail tốt.

Để giải bài toán này đã có rất nhiều phương pháp được đưa ra như: thuật toán NaiveBayes, SVM, Neural Network, Convolutional Neural Network,... Mỗi phương pháp đều cho kết quả khá tốt cho bài toán này, tuy nhiên phương pháp phân loại bằng NaiveBayes và SVM được sử dụng khá phổ biến và dễ cài đặt hơn cả. Chính vì vậy nhóm em đã chọn đề tài: "**Phân loại văn bản bằng Naïve Bayes và SVM**".

1. Giới thiệu bài toán

1.1. Mục đích của bài toán

Hệ thống phân loại văn bản dùng được xây dựng với mục đích phân loại văn bản thành 10 nhóm: Chính trị xã hội, Đời sống, Khoa học, Kinh doanh, Pháp luật, Sức khỏe, Thế giới, Thể thao, Văn hóa, Vi tính. Đầu vào và đầu ra được mô tả như sau:

- Input: Đoạn văn bản hoặc đường dẫn đến một bài báo trên internet
- Output: Chủ đề của đoạn văn bản

1.2. Phương pháp giải quyết bài toán

Như đã nói ở phần đầu, đây là một bài toán phân loại văn bản, vì vậy cách giải quyết bài toán này là sử dụng các mô hình học máy. Các phương pháp có thể thực hiện như SVM, Naïve Bayes, Neural Network, Convolutional Neural Network,...

Như vậy, để giải quyết được bài toán này thì ta phải giải quyết được những vấn đề sau:

- Chuẩn bị dữ liệu
- Tiền xử lý dữ liệu
- Trích chọn đặc trưng
- Sử dụng các mô hình học máy cho bài toán phân lớp

1.3. Kịch bản hệ thống

Kịch bản hoạt động của hệ thống được thực hiện như sau:

- Người dùng nhập một đoạn văn bản hoặc đường link dẫn tới bài báo trên internet
- Hệ thống tiến hành tiền xử lý dữ liệu
- Load lại model đã thu được ở bước train hệ thống
- Hệ thống tiến hành gán nhãn cho đoạn văn bản

1.4 . Triển khai

Ngôn ngữ: Python 3.5

Thư viện: numpy, gensim, os, pickle, sklearn, pyvi

2. Chuẩn bị dữ liệu

2.1. Đặc điểm của dữ liệu

Bộ dữ liệu sử dụng dựa trên nguồn tài nguyên chính là 4 trang báo điện tử của Việt Nam: VnExpress (http://www.vnexpress.net/), Tuổi trẻ Online (http://www.tuoitre.com.vn/), Thanh niên Online (http://www.thanhnien.com.vn/), Người lao động Online (http://www.nld.com.vn/). Dữ liệu đã được qua một quá trình tiền xử lý tự động trên máy tính như gỡ bỏ các tag HTML, chuẩn hóa chính tả,…

Dữ liệu gồm 10 chủ đề lớn, chia thành: Chính trị xã hội, Đời sống, Khoa học, Kinh doanh, Pháp luật, Sức khỏe, Thế giới, Thể thao, Văn hóa, Vi tính. Trong đó chứa khoảng 33759 tài liệu dùng cho tập huấn luyện và 50373 tài liệu dùng cho tập kiểm nghiệm.

Tên chủ đề	Số file huấn luyện	Số file kiểm chứng
Chính trị xã hội	5219	7567
Đời sống	3159	2036
Khoa học	1820	2096
Kinh doanh	2552	5276
Pháp luật	3868	3788
Sức khỏe	3384	5417
Thế giới	2898	6716
Thể thao	5298	6667
Văn hóa	3080	6250
Vi tính	2481	4560
Tổng cộng	33759	50373

2.2. Tiền xử lý dữ liệu

Dựa vào đặc điểm của dữ liệu được nêu ở phía trên, việc tiền xử lý dữ liệu bao gồm các công đoạn:

- Loại bỏ những ký tự đặc biệt trong văn bản ban đầu như dấu chấm, dấu phẩy, dấu mở đóng ngoặc,...
- Tách từ tiếng Việt. Một điểm đặc biệt trong văn bản tiếng Việt đó là một từ có thể được kết hợp bởi nhiều tiếng khác nhau, ví dụ như: sử_dụng, bắt_đầu,... khác với tiếng Anh và một số ngôn ngữ khác, các từ được phân cách nhau bằng khoảng trắng: use some examples, i love you... Vì vậy chúng ta cần tách từ để có thể đảm bảo ý nghĩa của từ được toàn vẹn. Sử dụng bộ tách từ có sẵn Pyvi, ví dụ: trường đại học bách khoa hà nội. → trường đại_học bách_khoa hà_nội .
- Sau khi thực hiện tuần tự và đầy đủ theo quy trình trên, ta thu được bộ dữ liệu sạch cho pha tiếp theo của mô hình.

2.3. Trích chọn đặc trưng

Tf-Idf(Term Frequency – Inverse Document Frequency) là một trong những phương pháp được sử dụng để khắc phục nhược điểm của mô hình Bag of word. Đó là trọng số của một từ trong văn bản thu được qua thống kê thể hiện mức độ quan trọng của từ này trong văn bản đó, mà bản thân văn bản đang xét nằm trong một tập hợp các văn bản.

Cách tính trọng số tf-idf:

- Tf (Term Frequency): tần số xuất hiện của một từ trong văn bản

$$\text{tf}(t, d) = \frac{f(t,d)}{\max\{f(w,d):w \epsilon d\}}$$

Trong đó:

- f(t,d): số lần xuất hiện từ t trong văn bản d
- max {f(w,d):w∈d} : số lần xuất hiện nhiều nhất của một từ bất kỳ trong văn bản d. Tuy nhiên, đối với mỗi văn bản thì có độ dài khác nhau, vì thế số lần xuất hiện của từ có thể nhiều hơn. Do đó số lần xuất hiện của từ sẽ được chia độ dài của văn bản (tổng số từ trong văn bản đó). Khi đó công thức tính Tf được tính như sau:

$$\text{tf}(t, d) = \frac{\textit{số lần từ t xuất hiện trong văn bản d}}{\textit{tổng số từ trong văn bản d}}$$

- Idf (Inverse document frequency): tần số nghịch của một từ trong tập văn bản. Idf được dùng để ước lượng mức độ quan trọng của từ đó như thế nào. 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 sử dụng nhiều nhưng không quan trọng để thể hiện ý nghĩa của đoạn văn ví dụ: giới từ, từ chỉ định,…

$$\text{idf}(t, D) = \log \frac{|D|}{|\{d \epsilon D : t \epsilon d\}|}$$

Trong đó:

- |D|: tổng số văn bản trong tập D
- |{d∈ *D*:t∈d}|: Số văn bản trong d có chứa từ t. Nếu từ đó không xuất hiện ở bất cứ văn bản nào trong tập D thì mẫu số sẽ bằng 0. Phép chia cho 0 không hợp lệ nên người ta thường cộng thêm 1 vào biểu thức dưới mẫu.

- Giá trị tf-idf:

$$\text{tf-idf}(t,d,D) = \text{tf}(t,d) * \text{idf}(t,D)$$

3. Giải quyết bài toán

3.1. Naïve Bayes

- Naïve Bayes là một thuật toán phân loại cho các vấn đề phân loại nhị phân (hai lớp) hoặc nhiều lớp.
- Xét bài toán phân loại với C lớp 1, 2, …, C. Giả sử có một điểm dữ liệu x $\in R^d$. Tính xác suất để điểm dữ liệu này rơi vào class c. Nói cách khác cần tính:

$P(y = c|x)$ hay $P(c|x)$ (1): xác suất để đầu ra là class c biết rằng đầu vào là vector x

- Biểu thức trên nếu tính được sẽ giúp xác định được xác suất để điểm dữ liệu rơi vào mỗi class. Từ đó có thể giúp xác định được class của điểm dữ liệu đó bằng cách chọn ra class có xác suất cao nhất: $c = \arg\max p(c|x)$ (2)
- Sử dụng quy tắc Bayes:

$$c = \arg\max_c p(c|\mathbf{x}) \quad (3)$$
$$= \arg\max_c \frac{p(\mathbf{x}|c)p(c)}{p(\mathbf{x})} \quad 4)$$
$$= \arg\max_c p(\mathbf{x}|c)p(c) \quad (5)$$

Trong công thức (5), $p(c)$ là xác suất để một điểm rơi vào class c. Giá trị này có thể tính bằng MLE, tức tỷ lệ số điểm dữ liệu trong tập training rơi vào class này chia cho tổng số lượng dữ liệu trong tập training, hoặc cũng có thể đánh giá bằng MAP estimation. Trường hợp thứ nhất thường được sử dụng nhiều hơn.
Thành phần còn lại $p(x|c)$ là phân phối của các điểm dữ liệu trong class c, thường khó tính toán vì x là một biến ngẫu nhiên nhiều chiều, cần rất nhiều dữ liệu training để có thể xây dựng được phân phối này. Để giúp tính toán đơn giản, người ta thường giả sử một cách đơn giản nhất rằng các thành phần của biến ngẫu nhiên x là độc lập với nhau, tức là:

$$p(\mathbf{x}|c) = p(x_1, x_2, \ldots, x_d|c) = \prod_{i=1}^{d} p(x_i|c) \quad (6)$$

Naïve Bayes nhờ vào việc giả sử các chiều của dữ liệu nên có tốc độ training và test rất nhanh.

$$c = \arg\max_{c\in\{1,\ldots,C\}} p(c) \prod_{i=1}^{d} p(x_i|c) \quad (7)$$

Ở bước training, các phân phối $p(c)$ và $p(x_i|c)$, i= 1,…,d sẽ được xác định dựa vào training data.
Ở bước test, với một điểm dữ liệu mới x, class của nó sẽ được xác định bởi:

Khi d lớn và các xác suất nhỏ, biểu thức ở vế phải của (7) sẽ là một số rất nhỏ, khi tính toán có thể gặp sai số. Để giải quyết, (7) thường được viết lại dưới dạng tương đương bằng cách lấy log của vế phải:

$$c = \arg \max_{c \in \{1,\dots,C\}} = \log(p(c)) + \sum_{i=1}^{d} \log(p(x_i|c)) \quad (7.1)$$

Việc này không ảnh hưởng tới kết quả vì log là một hàm đồng biến trên tập các số dương.

- Mặc dù giả thiết mà Naïve Bayes sử dụng là phi thực tế nhưng chúng vẫn hiệu quả trong nhiều bài toán thực tế, đặc biệt trong các bài toán phân loại văn bản.
- Mỗi giá trị $p(c)$ có thể được xác định như là tần suất xuất hiện của class c trong training data. Việc tính toán $p(x_i|c)$ phụ thuộc vào loại dữ liệu, có ba loại được sử dụng phổ biến là: Gaussian Naïve Bayes, Multinomial Naïve Bayes và Bernoulli Naïve.

- Mutinomial Naïve Bayes:
  - Mô hình này chủ yếu được sử dụng trong phân loại văn bản. Lúc này 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 i xuất hiện trong văn bản đó.
  - Khi đó, $p(x_i|c)$ tỷ lệ với tần suất từ thứ i xuất hiện trong các văn bản của class c. Giá trị này có thể tính được bằng cách:

$$\lambda_{ci} = p(x_i|c) = \frac{N_{ci}}{N_c}$$

Trong đó:

- N_{ci} là tổng số lần từ thứ i xuất hiện trong các văn bản của class c, nó được tính là tổng của tất cả các thành phần thứ i của các feature vectors ứng với class c.
- N_c là tổng số từ (kể cả lặp) xuất hiện trong class c. Nói cách khác, nó bằng tổng độ dài của toàn bộ các văn bản thuộc vào class c. Có thể suy ra rằng $N_c = \sum_{i=1}^{d} N_{ci}$, $\sum_{i=1}^{d} \lambda_{ci} = 1$.

  - Cách tính này có một hạn chế là nếu có một từ mới chưa bao giờ xuất hiện trong class c thì biểu thức sẽ bằng 0, dẫn đến kết quả không chính xác. Để giải quyết việc này, người ta áp dụng kỹ thuật Laplace smoothing:

$$\hat{\lambda}_{ci} = \frac{N_{ci} + \alpha}{N_c + d\alpha}$$

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

$$\sum_{i=1}^{d} \hat{\lambda}_{ci} = 1.$$

Như vậy, mỗi class c sẽ được mô tả bởi bộ các số dương có tổng bằng 1: $\hat{\lambda}_c = \{\hat{\lambda}_{c1}, \ldots, \hat{\lambda}_{cd}\}$.

3.2. SVM

Máy vector hỗ trợ (SVM) được đề cử bởi V. Vapnik và các đồng nghiệp của ông vào những năm 1970s tại Nga và sau đó trở nên nổi tiếng và phổ biến vào những năm 1990s.

SVM là một trong những thuật toán *classification* phổ biến nhất trong nhóm các giải thuật học máy có giám sát (*Supervise learning*). SVM dạng chuẩn nhận các điểm dữ liệu đầu vào và phân loại chúng vào hai lớp khác nhau (*binary classifiers*). Với một bộ các ví dụ huấn luyện thuộc 2 thể loại cho trước, thuật toán SVM sẽ tiến hành xây dựng một siêu phẳng (*hyperplane*) phân chia 2 thể loại đó thành 2 lớp tương ứng sao cho khoảng cách từ điểm gần nhất của mỗi lớp tới siêu phẳng này là lớn nhất. Khoảng cách đó được gọi là *margin*.

SVM là một phương pháp phân lớp tuyến tính với mục đích xác định một siêu phẳng để phân tách hai lớp của dữ liệu. SVM là một phương pháp phù hợp đối với những bài toán phân lớp có không gian rất nhiều chiều.

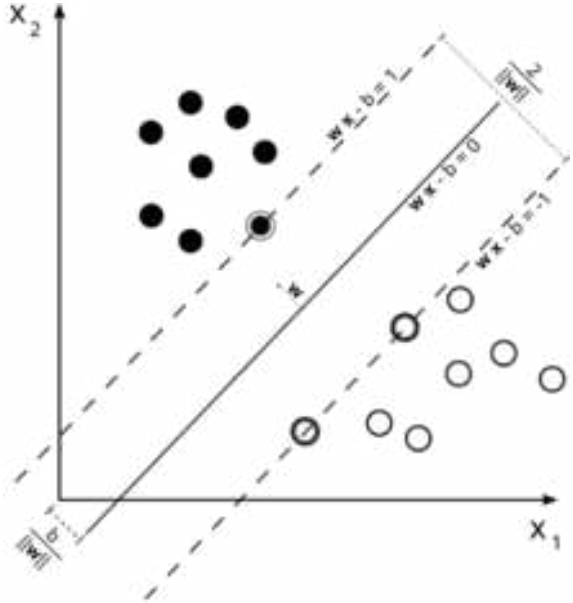
Một máy vectơ hỗ trợ xây dựng một siêu phẳng hoặc một tập hợp các siêu phẳng trong một không gian nhiều chiều hoặc vô hạn chiều, có thể được sử dụng cho phân loại, hồi quy, hoặc các nhiệm vụ khác.Giả sử ta có bài toán có một tập huấn luyện D gồm n điểm có dạng:

$$\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathbb{R}^p,\ y_i \in \{-1, 1\}\}_{i=1}^{n}$$

với y_i mang giá trị 1 hoặc −1, xác định lớp của điểm x_i. Mỗi x_i là một vectơ thực p-chiều.

Siêu mặt phẳng phân tách các quan sát thuộc lớp dương và các quan sát thuộc lớp âm là

<W*X>+b=0



Nếu dữ liệu huấn luyện có thể được chia tách một cách tuyến tính, thì ta có thể chọn siêu phẳng có lề sao cho không có điểm nào ở giữa chúng và sau đó

tăng khoảng cách giữa chúng đến tối đa có thể. Bằng hình học, ta tìm được khoảng cách giữa hai siêu phẳng lề là $\frac{2}{||W||}$.

Từ đó tương đương với việc giải quyết bài toán tối ưu bậc hai sau:

Tìm W và b sao cho : margin=$\frac{2}{||W||}$ đạt cực đại với điều kiện:

<W*X_i> + b >= 1, if Y_i =1 ,

<W*X_i> + b <= 1, if Y_i =-1.

Hay bài toán:

Tìm W và b sao cho : margin=$\frac{||W||}{2}$ đạt cực tiểu với điều kiện:

Y_i *(<W*X_i> + b) >= 1, với mọi i= 1,2,...,n

Từ đó cần giải được bài toán cực tiểu hóa có ràng buộc bất đẳng thức:

Cực tiểu hóa f(x), với các điều kiện $g_i(x)$ <= 0

$$\begin{cases} \dfrac{\partial}{\partial \mathbf{x}}\left(f(\mathbf{x}) + \sum_{i=1}^{r} \alpha_i g_i(\mathbf{x}) \right)\Big|_{\mathbf{x}=\mathbf{x}_0} = 0; & \text{với } \alpha_i \geq 0 \\ g_i(\mathbf{x}) \leq 0 \end{cases}$$

với α_i là một hệ số nhân Lagrange.

Biểu thức Langrange:

$$L_P(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\langle \mathbf{w} \cdot \mathbf{w} \rangle - \sum_{i=1}^{r} \alpha_i [y_i(\langle \mathbf{w} \cdot \mathbf{x}_i \rangle + b) - 1]$$

với α_i là các hệ số nhân Lagrange.

Lý thuyết tối ưu chỉ ra rằng một lời giải tối ưu cho hàm Lagrange trên phải thỏa mãn các điều kiện Karush-Kuhn-Tucker. Đối với SVM, các bài toán cực tiểu khóa có hàm mục tiêu lồi và các ràng buộc tuyến tính thì các điều kiện Karush-Kuhn-Tucker là cần và đủ đối với một lời giải tối ưu.

Hard Margin SVM: được áp dụng khi dữ liệu của cả 2 lớp là phân biệt tuyến tính (linearly separable). Giả sử ta có một tập huấn luyện D gồm N điểm dữ liệu như sau: (x_1, y_1), (x_2, y_2),…,(x_n, y_n),$x_i \epsilon R^T$, $y_i \epsilon \{1, -1\}$, d là số chiều của các điểm dữ liệu, yi là nhãn của điểm dữ liệu thứ i, y_i = 1 (class 1), y_i = -1 (class 2). Siêu phẳng phân chia 2 class có thể viết dưới dạng: w.x + b = 0

Khi đó, với cặp dữ liệu (x_n, y_n) bất kỳ, ta có khoảng cách từ điểm đó tới mặt chia là: $\frac{y_n.(w^t.x_n+b)}{||w||_2}$ Suy ra, ta có lề margin được tính bằng:

$$\text{margin}=min_n \frac{y_n.(w^t.x_n+b)}{||w||_2}$$

Ta cần chọn w và b sao cho margin là lớn nhất.

Khi đó, ta giả sử: $y_n(w^t.x_n+ b) = 1$ với những điểm nằm gần mặt phân chia nhất. Như vậy, $y_n(w^t.x_n+ b) \geq 1$ với mọi n. Bài toán tối ưu SVM trở thành: (w, b) = arg $min_{w,b} \frac{1}{2} ||w||_2^2$ với điều kiện: 1 - $y_n(w^t.x_n+ b) \leq 0$ với mọi n = 1, 2, …, N Xác định class cho một điểm dữ liệu mới: class(x) = sgn(w^t + b)

Soft Margin SVM: Hard Margin SVM làm việc không hiệu quả trong 2 trường hợp:

- Xuất hiện các điểm "nhiễu" trong bộ dữ liệu, mặc dù chúng vẫn phân biệt tuyến tính.
- Dữ liệu không phân biệt tuyến tính nhưng gần phân biệt tuyến tính.

Thuật toán Soft Margin SVM là một biến thể của Hard Margin SVM, thường được sử dụng để giải quyết các trường hợp ngoại lệ nêu trên. Trong bài toán này, ta chấp nhận "hy sinh" một số điểm dữ liệu được coi là nhiễu để có thể tìm được siêu phẳng cho kết quả tốt nhất. Tuy nhiên, chúng ta cần phải hạn chế sự "hy sinh" này. Do đó, hàm mục tiêu của bài toán vừa nhằm mục đích tối đa hóa giá trị margin, vừa nhằm tối thiểu hóa "sự hy sinh" các điểm dữ liệu. Với mỗi điểm x_n, ta sử dụng một biến mới ξ_n là để đo sự hy sinh tương ứng với điểm dữ liệu đó. Với những điểm x_n nằm trong vùng an toàn (vùng được phân lớp đúng) thì = 0. Với những điểm nằm trong vùng không an toàn thì có > 0. Nếu yi = ±1 là nhãn của xi trong vùng không an toàn thì = $|w^t.x_i + b - y_i|$. Hàm mục tiêu:

$$\frac{1}{2}||w||_2^2 + C.\sum_{n=1}^{N}\xi_n$$

trong đó : C là hằng số dương được dùng để điều chỉnh giữa margin và sự hy sinh dữ liệu.

⇒ Bài toán tối ưu của Soft Margin SVM là:

$(w, b, \xi) = \arg\min_{w,b,\xi} \frac{1}{2}||w||_2^2 + C.\sum_{n=1}^{N}\xi_n$ với điều kiện: $1 - \xi_n - y_n(w^T.x_n + b) \leq 0$, $-\xi_n \leq 0$ với mọi n = 1, 2,..., N

⇒ Hard Margin SVM là một trường hợp đặc biệt của Soft Margin SVM. = 0 tương ứng với những điểm nằm trong vùng an toàn, $0 < \leq 1$ tương ứng với những điểm nằm trong vùng không an toàn nhưng vẫn được phân lớp đúng, > 1 tương ứng với những điểm bị phân loại sai.

Kernel SVM: được áp dụng đối với các bài toán có dữ liệu hoàn toàn không phân biệt tuyến tính. Ý tưởng cơ bản của Kernel SVM và các phương pháp kernel nói chung là tìm một phép biến đổi sao cho từ bộ dữ liệu ban đầu là không phân biệt tuyến tính được biến đổi sang một không gian mới mà ở đó, dữ liệu trở nên phân biệt tuyến tính. Khi đó, ta có thể dùng các thuật toán phân lớp tuyến tính thông thường như PLA, Logistic Regression, hay Hard/Soft Margin SVM để tìm ra mặt phân chia các điểm dữ liệu. Ở đây, chúng ta sẽ tiếp cận theo hướng là sử dụng các kernel functions để mô tả mối quan hệ giữa hai điểm dữ liệu bất kỳ trong không gian mới.

Một số kernel functions thông dụng:

Tên	Công thức	Thiết lập hệ số
linear	$x^T.z$	Không có hệ số
polynomial	$(r + \gamma x^T z)^d$	d: dgree (> 0, bậc của đa thức), γ: gamma, r: coef0
sigmoid	$\tanh(\gamma x^T z + r)$	γ: gamma, r: coef0
rbf	$\exp(-\gamma \|\|x - z\|\|_2^2)$	$\gamma > 0$: gamma

Ưu, nhược điểm của SVM:

- Ưu điểm:

- Xử lý dữ liệu với số chiều cao: SVM là một công cụ tính toán hiệu quả trong không gian có số chiều cao, trong đó đặc biệt hữu ích khi áp dụngcho các bài toán phân loại văn bản và phân tích quan điểm người dùng, đó là các bài toán có số 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 cao: Trong thực tế, các bài toán 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 suấtphân loại lớn hơn.

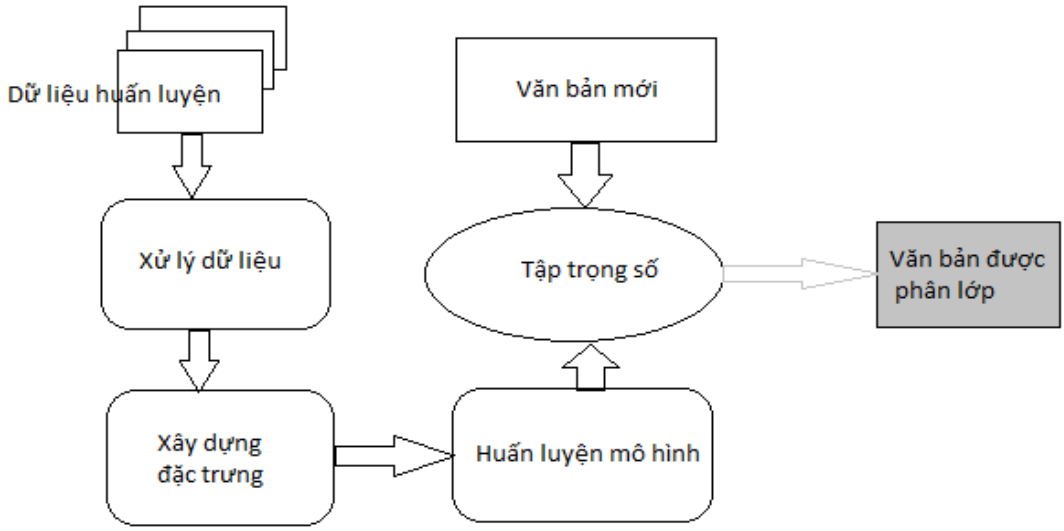
- Nhược điểm:

- Bài toán có số chiều cao: Trong trường hợp số lượng thuộc tính của tậpdữ liệu lớn hơn rất nhiều so với số lượng dữ liệu thì SVM cho kết quả khá tồi.

- Các bài toán SVM thuần (Hard Margin SVM) hoạt động không hiệu quả khi có nhiễu ở gần biên hoặc thậm chí khi dữ liệu giữa hai lớp gần phân biệt tuyến tính.

3.3. Áp dụng Naïve Bayes và SVM vào bài toán phân loại văn bản

Mô hình phân loại:



3.3.1. Áp dụng Naïve Bayes vào bài toán phân loại văn bản

Ta áp dụng phương pháp Naïve Bayes vào chương trình phân loại với cách tiếp cận NaiveBayes là sử dụng xác suất có điều kiện giữa từ và chủ đề để dự đoán xác suất chủ đề của một văn bản cần phân loại. Điểm quan trọng của phương pháp này chính là ở chỗ giả định rằng sự xuất hiện của các từ trong văn bản độc lập với nhau. Giả định đó làm cho việc tính toán NB hiệu quả và nhanh chóng hơn các phương pháp khác vì không sử dụng kết hợp các từ để đưa ra phán đoán chủ đề.

Áp dụng trong bài toán phân loại văn bản, các dữ kiện gồm có:

- D: tập dữ liệu huấn luyện đã được vector hóa
- C_i: phân lớp i, với i= {1,2,…10} (10 lớp)
- Các thuộc tính độc lập điều kiện đôi một với nhau

Các bước thực hiện thuật toán Naïve Bayes:

Bước 1: Huấn luyện Naïve Bayes(dựa vào tập dữ liệu), tính $P(C_i)$ và $P(x_k|C_i)$

Bước 2: Phân lớp $X^{new} = (x_1, x_2, \ldots, x_n)$, ta cần tính xác suất thuộc từng phân lớp khi đã biết trước X^{new}. X^{new} được gán vào lớp có xác suất lớn nhất theo công thức:

$$\max_{C_i \in C} \left(P(C_i) \prod_{k=1}^{n} P(x_k|C_i) \right)$$

3.3.2. Áp dụng SVM vào bài toán phân loại văn bản

- Bài toán phân loại văn bản là một bài toán thuộc dạng *multi-class classification*. Cụ thể, đối với mỗi điểm dữ liệu đầu vào (1 đoạn văn bản), bài toán cần xác định xem điểm dữ liệu đó thuộc vào lớp (chủ đề) nào trong số 10 lớp. Các phương pháp Support Vector Machine đã đề cập (Hard Margin, Soft Margin, Kernel) đều được xây dựng nhằm giải quyết các bài toán *binary classification* (hay còn gọi là *binary classifiers*), tức các bài toán phân lớp với chỉ với hai classes. Để giải quyết bài toán này, chúng em đã tiến hành sử dụng thư viện *sklearn.svm.SVC* được hỗ trợ bởi

thư viện scikit-learn. Ở đây, mô hình phân loại đa lớp được xử lý theo phương pháp sử dụng nhiều binary classifiers kết hợp với kỹ thuật onevs-one

- One-vs-one: Xây dựng rất nhiều bộ binary classifiers cho từng cặp classes. Bộ thứ nhất phân biệt class 1 và class 2, bộ thứ hai phân biệt class 1 và class 3,… Khi có một dữ liệu mới vào, ta đưa nó vào toàn bộ các bộ binary classifiers trên. Kết quả cuối cùng có thể được xác định bằng cách xem class nào mà điểm dữ liệu đó được phân vào nhiều nhất (major voting). Như vậy, nếu có *C* classes thì tổng số binary classifiers phải dùng là Số lớn, kém hiệu quả trong tính toán.
- Trong bài toán này, chúng em tiến hành thử nghiệm thuật toán trên 3 mô hình ứng với 3 kernel functions khác nhau, đó là: linear, poly và rbf

4. Đánh giá

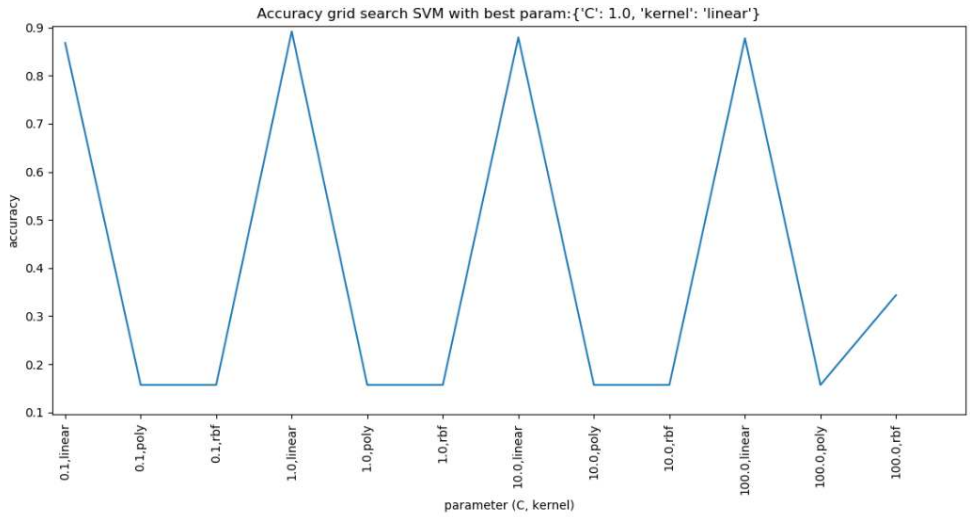
4.1. Naïve Bayes

Thực hiện đưa dữ liệu dạng văn bản đã được xử lý về dạng vector thuộc tính có dạng số học. Sử dụng phương pháp tf-idf để đưa dữ liệu văn bản dạng text về dạng số. Sau khi thực hiện sẽ thu được một ma trận mà trong đó, mỗi hàng đại diện cho một văn bản, mỗi cột đại diện cho một từ có trong từ điển và mỗi ô sẽ chứa giá trị tf-idf của từ trong văn bản tương ứng.
Từ đó, chúng ta thực hiện huấn luyện mô hình NaiveBayes.
Kết quả kiểm tra trên tập test đạt độ chính xác: **86.49%**

4.2. SVM

Thực hiện đưa dữ liệu dạng văn bản đã được xử lý về dạng vector thuộc tính có dạng số học. Sử dụng phương pháp tf-idf để đưa dữ liệu văn bản dạng text về dạng số. Sau khi thực hiện sẽ thu được một ma trận mà trong đó, mỗi hàng đại diện cho một văn bản, mỗi cột đại diện cho một từ có trong từ điển và mỗi ô sẽ chứa giá trị tf-idf của từ trong văn bản tương ứng.
Từ đó, chúng ta thực hiện huấn luyện mô hình SVM.
Thực hiện tối ưu tham số của mô hình SVM:

- C : 0.1, 1.0, 10, 100
- kernels (hàm nhân): linear, poly, rbf



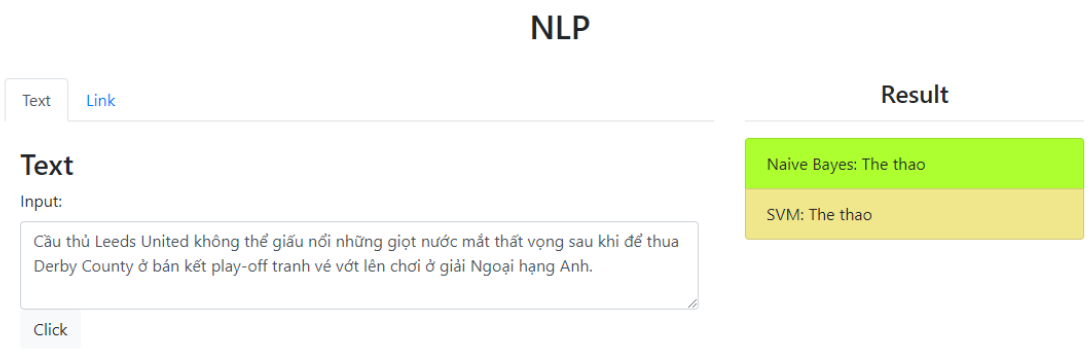
Bộ tham số tốt nhất: **C=1, kernel=linear**
Kết quả kiểm tra trên tập test đạt độ chính xác: **92.75%**

4.3. So sánh các mô hình đề xuất

Mô hình	Accuracy of test
Naïve Bayes	86.49%
SVM	92.75%

Naïve Bayes ưu điểm hơn so với SVM ở việc học nhanh chóng hơn do giả định rằng sự xuất hiện của tất cả các từ trong văn bản độc lập với nhau. Tuy nhiên, SVM lại có độ chính xác cao hơn Naïve Bayes.

5. Demo



PHÂN CÔNG CÔNG VIỆC

Ngô Vân Anh	Tìm hiểu, thực hiện SVM
Tô Hương Giang	Tìm hiểu, thực hiện Naïve Bayes
Nguyễn Hoàng Giang	Tiền xử lý dữ liệu, trích chọn đặc trưng

TÀI LIỆU THAM KHẢO

[1] Diễn đàn Machine learning cơ bản
https://machinelearningcoban.com/
[2] Dataset: https://github.com/duyvuleo/VNTC
[3] Slide bài giảng Xử lý ngôn ngữ tự nhiên – PGS.TS Lê Thanh Hương