ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA TOÁN - CƠ - TIN HỌC

BÁO CÁO
LOGISTIC REGRESSION
VÀ ỨNG DỤNG CHO BÀI TOÁN
PHÂN TÍCH SẮC THÁI BÌNH LUẬN

Nhóm 5
Đỗ Tất Thành
Cao Thọ Hiếu

Mục lục

I Logistic Regression - Hồi quy Logistic

Logistic Regression là một công cụ phân tích quan trọng trong khoa học tự nhiên và khoa học xã hội. Trong lĩnh vực xử lý ngôn ngữ tự nhiên, Logistic Regression đóng vai trò là thuật toán học máy có giám sát, và có quan hệ gần gũi với Neural Networks. Một mạng Neural Networks có thể xem như một chuỗi các Logistic Regression xếp chồng lên nhau. Logistic Regression có thể sử dụng để phân loại hai hay nhiều lớp (binary/multinomial logistic regression), nhưng trước hết, ta sẽ tìm hiểu một số khái niệm về học máy và logistic regression.

I.1 Generative Classifiers và Discriminative Classifiers:

Là hai framework khác nhau trong việc xây dựng mô hình học máy. *Generative model* đặt mục tiêu *hiểu* được đối tương đầu vào và xây dựng mô hình *định nghĩa* đối tượng đó. Trong khi đó, *Discriminative model* không tìm cách *hiểu* nhóm đối tượng mà cố gắng tìm cách phân biệt giữa các nhóm đối tượng. Generative model *tập trung vào đối tượng*, còn Discriminative Classifiers thì hướng đến *sự khác biệt giữa các nhóm đối tượng*. Ví dụ, ta muốn xây dựng một mô hình học máy phân biệt ảnh của con chó với ảnh của con mèo, mô hình Generative Classifiers đặt mục tiêu hiểu được chó trông như thế nào và mèo trông ra sao. Còn Discriminative model sẽ học cách phân biệt giữa hai nhóm dữ liệu trong quá trình huấn luyện.

Nếu bộ dữ liệu bao gồm những con chó đeo vòng cổ, trong khi những con mèo thì không, mô hình có khả năng xác định rằng, những ảnh có vòng cổ là ảnh của chó, và ngược lại. Khi phân loại một ảnh mới, nếu ảnh có vòng cổ, mô hình sẽ cho rằng đó là ảnh của chó, kể cả khi đó là một con mèo đeo vòng cổ.

I.2 Các thành phần của một thuật toán học máy phân loại dựa trên xác suất:

Logistic Regression là một thuật toán học máy có giám sát giúp phân loại đối tượng dựa trên xác suất. Giả thiết tập huấn luyện (training) bao gồm M mẫu (x^i, y^i). Một hệ thống học máy cho phân loại sẽ bao gồm 4 thành phần:

feature representation / biểu diễn các đặc trưng của đầu vào, thường là một vector các đặc trưng của mẫu dữ liệu đầu vào $x^{(i)}$: $[x^1, x^2, ..., x^n]$.

Một *hàm phân loại* có nhiệm vụ tính ŷ, lớp của đối tượng mà mô hình dự đoán. Hai hàm phân loại phổ biến là *sigmoid* và *softmax*.

Một *hàm mục tiêu* cho việc học, thường dẫn đến việc tối thiểu hóa lỗi trong phân loại của mô hình ở bước huấn luyện. Hàm mục tiêu được sử dụng ở đây là *cross-entropy loss function*.

Một *thuật toán giúp tối ưu hàm mục tiêu*. Ở đây, thuật toán được sử dụng là *SGD-stochastic gradient descent*.

I.3 Các pha của quá trình huấn luyện:

training: huấn luyện hệ thống (các trọng số w và bias b), sử dụng SGD và cross-entropy loss function.

Validation: hiệu chỉnh hệ thống thông qua tập validation.

Testing: phân loại nhãn cho tập kiểm tra, đánh giá độ chính xác của mô hình được xây dựng.

I.4 Quá trình học của Logistic Regression:

I.4.1 *Trích chọn đặc trưng*:

Đặc trưng được trích chọn thông qua quá trình khảo sát dữ liệu huấn luyện bằng những kiến thức ngôn ngữ học và chuyên ngành. Đặc trưng cũng có thể được trích chọn thông qua quá trình phân tích lỗi trong lúc huấn luyện, hoặc trong quá trình chạy thử nghiệm hệ thống ở những phiên bản đầu. Có thể xây dựng những đặc trưng phức tạp từ quá trình kết hợp những đặc trưng cơ bản.

Cho những công việc cần đến một số lượng lớn các đặc trưng, khi đó, việc trích chọn đặc trưng một cách thủ công trở nên không khả thi. Để giải quyết vấn đề, người ta tiến hành xây dựng *feature templates*, mô tả trừu tượng của các đặc trưng cần trích chọn và tiến hành chọn một cách tự động dựa trên mô tả. *Bigram templates* (hay *n-gram template*) là một template chọn đặc trưng tự động như vậy. Và để tránh sự tốn kém về công sức khi trích chọn thủ công, nghiên cứu NLP có một nhánh tập trung vào *representation learning,* các cách thức để học tự động các đặc trưng một cách phi giám sát từ dữ liệu đầu vào.

I.4.2 *Phân lớp dữ liệu:*

Cho vector thể hiện các đặc trưng $[x_1, x_2, ..., x_n]$ của mẫu x, nhãn phân loại là 1 (x là thành viên của lớp được xét) hoặc 0 (x không phải là thành viên của lớp được xét), và chúng ta muốn biết xác suất $P(y = 1|x)$ mẫu đã cho thuộc về lớp được xét. LG giải quyết bài toán này bằng cách học từ tập huấn luyện một vector của *các trọng số w và b*. Mỗi trọng số w^i là một số thực, tương ứng với mỗi đặc trưng x^i. Trọng số w^i cho biết mức độ quan trọng của đặc trưng x^i khi đóng góp vào việc dự đoán lớp mà x thuộc về, có thể đạt giá trị dương (nghĩa là đặc trưng tương ứng với lớp được xét),

hoặc âm (đặc trưng không tương ứng với lớp được xét), hoặc bằng 0 (đặc trưng không đóng góp vào quá trình quyết định của mô hình).

I.4.2.1 *Hàm phân lớp nhị phân Sigmoid*:

Mục tiêu của LG cho hai lớp (binary LG) là huấn luyện một hàm phân loại nhị phân, giúp chỉ ra một mẫu dữ liệu thuộc lớp nào trong hai lớp (hoặc có thể hiểu là thuộc hay không thuộc một lớp nào đó). Và *hàm Sigmoid* thường được sử dụng để làm công việc đó.

Để xác định nhãn cho mẫu sau khi huấn luyện xong, mô hình nhân mỗi x^i với w^i tương ứng, cộng chúng lại, sau đó cộng thêm trọng số *bias* b, được kết quả z, thể hiện giá trị dự đoán (mỗi đặc trưng cùng với trọng số ảnh hưởng của nó đến dự đoán):

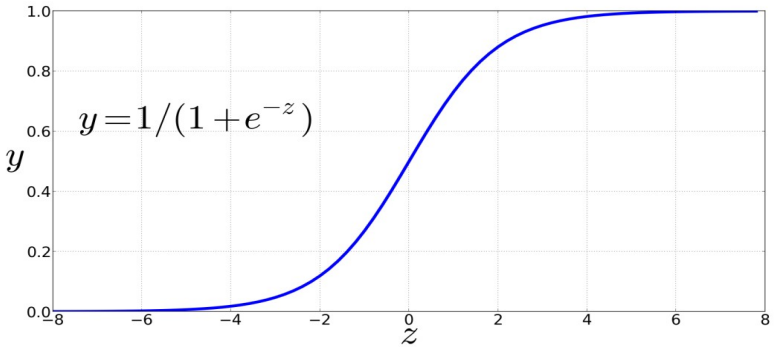
$$z = \left(\sum_{i=1}^{n} w_i x_i \right) + b$$

Hay có thể biểu diễn gọn hơn bằng phép nhân vô hướng vector (*dot product*):

$$z = w \cdot x + b$$

với w, x lần lượt là các vector trọng số và vector đặc trưng của mẫu. Vấn đề tiếp theo là, vì w, x là các vector số thực, có thể âm, có thể dương, do đó, z là một số thực tùy ý. Trong khi đó, điều chúng ta cần là một giá trị thể hiện xác suất của nhãn mà mô hình dự đoán, giá trị cần tìm cần trong đoạn [0, 1]. Để tạo ra giá trị xác xuất đó, ta xử lý giá trị z bằng một hàm “ép” giá trị thực tùy ý về đoạn [0, 1], và tổng xác suất của các nhãn bằng 1. Nếu mô hình phân loại nhị phân (có 2 lớp), mô hình sử dụng hàm sigmoid để phân loại. Còn nếu mô hình phân loại nhiều lớp hơn (multinomial LG), ta sử dụng hàm phân loại là *softmax*.

Đối với mô hình phân lớp nhị phân, ta sử dụng hàm sigmoid:



Do đó, chúng ta tính được xác suất P(y|x) cho mỗi nhãn 0 hoặc 1 của x:

$$\begin{aligned} P(y=1) &= \sigma(w \cdot x + b) \\ &= \frac{1}{1+e^{-(w \cdot x+b)}} \\ P(y=0) &= 1-\sigma(w \cdot x+b) \\ &= 1-\frac{1}{1+e^{-(w \cdot x+b)}} \\ &= \frac{e^{-(w \cdot x+b)}}{1+e^{-(w \cdot x+b)}} \end{aligned}$$

Sau khi tính được xác suất có thể cho mỗi nhãn, nhãn được dự đoán sẽ là nhãn có xác suất lớn hơn, tương đương với nhãn đó có xác suất lớn hơn 0.5:

$$\hat{y} = \begin{cases} 1 & \text{if } P(y=1|x) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

I.4.2.2 *Hàm phân lớp đa lớp Softmax*:

Đối với mô hình phân loại đa lớp - Multinomial LG, ta sử dụng hàm softmax để xác định xác suất nếu nhãn y = c (P(y=c|x), tương đương x thuộc lớp c). Hàm softmax nhận vào một vector thực z = [z_1, z_2, ..., z_k] có k giá trị z i tương ứng với mỗi trọng số w cho mỗi lớp, và ánh xạ chúng vào không gian phân phối xác suất (đoạn [0, 1]). Tổng các z_i có giá trị bằng 1.

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \quad 1 \le i \le k$$

$$\text{softmax}(z) = \left[\frac{e^{z_1}}{\sum_{i=1}^{k} e^{z_i}}, \frac{e^{z_2}}{\sum_{i=1}^{k} e^{z_i}}, \ldots, \frac{e^{z_k}}{\sum_{i=1}^{k} e^{z_i}} \right]$$

Từ đó tính được xác suất của mỗi nhãn:

$$p(y = c|x) = \frac{e^{w_c \cdot x + b_c}}{\sum_{j=1}^{k} e^{w_j \cdot x + b_j}}$$

Việc xây dựng các mô hình LG cho phân loại đối tượng tương ứng với việc học các trọng số w và bias b sao cho, khi sử dụng w và b đó cho thuật toán LG, nhãn ŷ mà mô hình dự đoán cho đối tượng gần sát nhất với nhãn thực (gold-label) y. Để làm được điều đó, chúng ta cần hai thành phần. Một thành phần là một độ đo cho sự khác nhau giữa ŷ và y, được gọi là *loss / cost function*, mà độ đo thường được sử dụng là *cross-entropy loss*. Thành phần thứ hai là một thuật toán tối ưu cho phép tự động cập nhật trọng số sao cho mô hình từ trọng số đó tối thiểu hóa hàm mục tiêu. Thuật toán chuẩn là GD, thuật toán được giới thiệu trong báo cáo là SGD.

I.4.3 *Cross-Entropy Loss Function:*

Loss Function là hàm cho phép thể hiện sự khác biệt giữa nhãn do mô hình phân loại (được tính bằng *(ŷ = σ (w · x + b))*) với nhãn chính xác của mẫu (y).

L(ŷ, y) = Sự khác biệt giữa ŷ và y

Để làm được vậy. chúng ta chọn các tham số w, b giúp tối ưu log xác suất của nhãn thực y trong dữ liệu huấn luyện. Hàm để thực hiện điều đó được gọi là cross-entropy loss. Đối với mô hình phân loại nhị phân (2 lớp, nhãn được đánh dấu 0 hoặc 1), loss function được tính bằng:

$$L_{CE}(\hat{y}, y) = -\log p(y|x) = -[y \log \hat{y} + (1 - y) \log(1 - \hat{y})] \quad (1)$$

Công thức này được suy ra từ mục đích tối ưu hóa xác suất của nhãn được gán chính xác p(y|x):

$$p(y|x) = \hat{y}^{\,y} (1 - \hat{y})^{\,1-y} \quad (2)$$

Lấy log của nó:

$$\log p(y|x) = \log[\hat{y}^{y}(1-\hat{y})^{1-y}] = y \log \hat{y} + (1-y)\log(1-\hat{y}) \qquad (3)$$

Từ (2) và (3) suy ra (1). Thế w, b trong công thức tính nhãn của mẫu dữ liệu x là $\hat{y} = \sigma(w \cdot x + b)$, ta có:

$$L_{CE}(w, b) = -[y \log \sigma(w \cdot x + b) + (1-y) \log(1 - \sigma(w \cdot x + b))]$$

Còn đối với mô hình phân lớp đa lớp, ta có hàm Loss Function được tính là:

$$\begin{aligned} L_{CE}(\hat{y}, y) &= -\sum_{k=1}^{K} 1\{y = k\} \log p(y = k|x) \\ &= -\sum_{k=1}^{K} 1\{y = k\} \log \frac{e^{w_k \cdot x + b_k}}{\sum_{j=1}^{K} e^{w_j \cdot x + b_j}} \end{aligned}$$

hàm 1{} có giá trị bằng 1 nếu điều kiện trong ngoặc là đúng và bằng 0 nếu ngược lại.

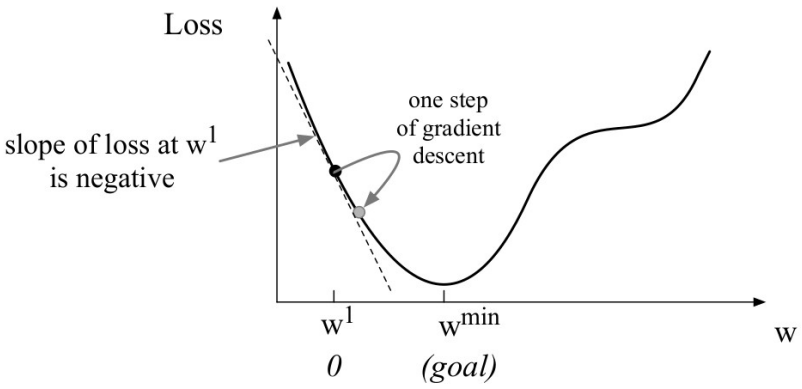
I.4.4 Gradient Descent:

Chúng ta sử dụng GD trong mô hình học máy nhằm mục đích tìm trọng số tối ưu, tối thiểu hóa loss function mà chúng ta đã định nghĩa trong mô hình:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \frac{1}{m} \sum_{i=1}^{m} L_{CE}(y^{(i)}, x^{(i)}; \theta)$$

Trong đó θ-mũ là tham số tối ưu cho mô hình, θ là tham số (w, b) của mô hình. $L_{CE}(x^{i}, y^{i}; \theta)$ là hàm loss-function tướng ứng, và m là số lượng mẫu trong tập training. Phương pháp GD (giảm gradient) tìm cực tiểu của hàm loss bằng cách xem xét (trong không gian của tham số θ) chiều hướng tăng nhanh nhất của hàm loss, sau đó điều chỉnh tham số để đi theo chiều ngược lại. Đối với LG, hàm loss là một hàm lồi, do đó chắc chắn có duy nhất cực tiểu và không có cực tiểu địa phương. Do đó, dù bắt đầu ở bất cứ điểm nào, GD cũng tìm được tới cực tiểu toàn cục.

Dưới đây là ví dụ mô tả cho ý tưởng của GD, không gian tham số θ chỉ có một chiều. Trục hoành thể hiện giá trị của tham số, trong khi trục tung là giá trị của hàm loss function:

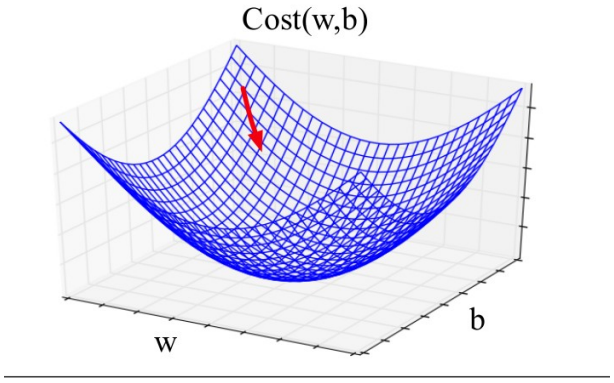


Điểm đen là điểm xuất phát tại w^1, đường đứt đoạn là gradient của hàm loss tại w^1. Ta thấy, chiều thay đổi của w theo hướng giảm giá trị hàm loss. Theo đó, w sẽ thay đổi để đạt được cực tiểu tại w^{min}. Lượng thay đổi của w trong GD được tính bằng giá trị của vi phân hàm loss theo w (df (x; w) / dw) được gán trọng số η gọi là *learning rate*. Learning rate lớn tương đương với việc w di chuyển nhanh hơn (thay đổi nhiều hơn) trong không gian tham số. Giá trị mới của tham số được cập nhật theo:

$$w^{t+1} = w^t - \eta \frac{d}{dw} f(x; w)$$

cho trường hợp trong ví dụ trên.

Tiếp theo là một ví dụ khác trong không gian tham số nhiều chiều (2 chiều). Hàm loss function vẫn là hàm lồi, và mũi tên màu đỏ cho biết hướng đi của tham số trong GD :



Gradient trong trường hợp không gian tham số nhiều chiều được tính bằng:

$$\nabla_\theta L(f(x;\theta),y)) \;=\; \begin{bmatrix} \frac{\partial}{\partial w_1} L(f(x;\theta),y) \\ \frac{\partial}{\partial w_2} L(f(x;\theta),y) \\ \vdots \\ \frac{\partial}{\partial w_n} L(f(x;\theta),y) \end{bmatrix}$$

với L là loss function, f(x;θ) = $\hat{y}$ là nhãn của x mà mô hình dự đoán.

Công thức tổng quát cho cập nhật tham số:

$$\theta_{t+1} \;=\; \theta_t - \eta \nabla L(f(x;\theta),y)$$

Từ đó suy ra các công thức tương ứng với trường hợp cụ thể binary LG hay multinomial LG.

Gradient cho binary LG:

$$\frac{\partial L_{CE}(w,b)}{\partial w_j} \;=\; [\sigma(w\cdot x+b)-y]x_j$$

Gradient cho thành phần w k trong multinomial GD:

$$\begin{aligned} \frac{\partial L_{CE}}{\partial w_k} &= -(1\{y=k\} - p(y=k|x))x_k \\ &= -\left(1\{y=k\} - \frac{e^{w_k\cdot x+b_k}}{\sum_{j=1}^{K} e^{w_j\cdot x+b_j}}\right)x_k \end{aligned}$$

Thuật toán thường được sử dụng là SGD, tối thiểu hóa hàm loss và cập nhật trọng số bằng tính toán gradient của nó sau mỗi mẫu huấn luyện. Thuật toán SGD:

function STOCHASTIC GRADIENT DESCENT($L()$, $f()$, x, y) **returns** θ

```
    # where: L is the loss function
    #        f is a function parameterized by θ
    #        x is the set of training inputs x^(1), x^(2),..., x^(n)
    #        y is the set of training outputs (labels) y^(1), y^(2),..., y^(n)

θ ← 0
repeat til done    # see caption
  For each training tuple (x^(i), y^(i)) (in random order)
    1. Optional (for reporting):        # How are we doing on this tuple?
       Compute ŷ^(i) = f(x^(i); θ)      # What is our estimated output ŷ?
       Compute the loss L(ŷ^(i), y^(i)) # How far off is ŷ^(i)) from the true output y^(i)?
    2. g ← ∇_θ L(f(x^(i); θ), y^(i))    # How should we move θ to maximize loss?
    3. θ ← θ − η g                      # Go the other way instead
return θ
```

Stochastic GD là phương pháp điều chỉnh trọng số dựa trên một mẫu ngẫu nhiên trong tập huấn luyện. Vì lý do đó, sự thay đổi của trọng số cũng ngẫu nhiên thay vì hội tụ trực tiếp tới điểm cực tiểu. Một phương pháp khác giúp hội tụ trực tiếp hơn là xem xét toàn bộ tập huấn luyện để tính toán gradient và điều chỉnh trọng số dựa trên việc xem xét toàn bộ mẫu, gọi là *batch training*. Phương pháp này cho một ước lượng tốt về hướng thay đổi của trọng số, với cái giá phải trả là tốn nhiều thời gian xử lý và tài nguyên tính toán.

Cân bằng giữa hai phương pháp trên là phương pháp *mini-batch* training, vẫn xem xét nhiều mẫu dữ liệu cùng lúc, nhưng không phải toàn bộ tập huấn luyện như *batch training* . Mỗi lần *mini-batch* sẽ huấn luyện một nhóm m mẫu dữ liệu, m nhỏ hơn toàn bộ số mẫu của tập huấn luyện. *Mini-batch* đạt ưu thế về hiệu năng tính toán, khi có thể dễ dàng vector hóa, chọn lựa kích thước khối dữ liệu cho phù hợp với khả năng tính toán của máy, và xử lý song song trên nhiều máy.

Với trường hợp sử dụng *mini-batch*, hàm loss-function và gradient từng phần sẽ là:

$$\begin{aligned} Cost(w,b) &= \frac{1}{m}\sum_{i=1}^{m} L_{CE}(\hat{y}^{(i)}, y^{(i)}) \\ &= -\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \log \sigma(w \cdot x^{(i)} + b) + (1 - y^{(i)}) \log \left(1 - \sigma(w \cdot x^{(i)} + b)\right) \end{aligned} \tag{5.21}$$

$$\frac{\partial L_{CE}(w,b)}{\partial w_j} = [\sigma(w \cdot x + b) - y] x_j$$

I.4.5 Hiệu chỉnh tham số:

Khi huấn luyện mô hình học máy, một vấn đề thường xảy ra là mô hình được huấn luyện có thể cho độ chính xác rất cao với dữ liệu trong tập train, nhưng lại tồi khi dự đoán những mẫu dữ liệu chưa từng thấy. Trường hợp này gọi là *overfitting*, khi mà mô hình chịu ảnh hưởng bởi nhiễu trong tập huấn luyện. Một mô hình tốt cần là một mô hình có khả năng khái quát hóa cho cả những dữ liệu nó chưa từng biết đến, và để tránh overfitting, cùng với tăng khả năng khái quát hóa của mô hình, cần hiệu chỉnh nó. Một *hàm hiệu chỉnh R(θ)* được thêm vào loss function để hiệu chỉnh trọng số, không để mô hình trở nên overfitting.

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{m} \log P(y^{(i)}|x^{(i)}) - \alpha R(\theta)$$

Hai hàm hiệu chỉnh được sử dụng phổ biến là *L2 regularization* hay chính là *khoảng cách Euclide*, được tính bằng:

$$R(\theta) = ||\theta||_2^2 = \sum_{j=1}^{n} \theta_j^2$$

Khi đó, tham số θ-mũ được tính là:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \left[\sum_{1=i}^{m} \log P(y^{(i)}|x^{(i)}) \right] - \alpha \sum_{j=1}^{n} \theta_j^2$$

Một hàm hiệu chỉnh phổ biến khác là *L1 regularization* hay *khoảng cách Mahattan:*

$$R(\theta) = ||\theta||_1 = \sum_{i=1}^{n} |\theta_i|$$

Và tham số θ-mũ trong trường hợp này được tính là:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \left[\sum_{1=i}^{m} \log P(y^{(i)}|x^{(i)}) \right] - \alpha \sum_{j=1}^{n} |\theta_j|$$

II Bài toán phân tích sắc thái bình luận:

II.1 Tổng quan về Sentiment Analysis

Phân tích tình cảm là quá trình tự động phát hiện các tình cảm tích cực, trung tính và tiêu cực trong văn bản bằng thuật toán học máy. Với các công cụ phân tích tình cảm trực tuyến, doanh nghiệp có thể theo dõi tình cảm của người tiêu dùng và hiểu nhu cầu của khách hàng bằng cách tự động gán các nhãn tình cảm cho dữ liệu của họ.

Phân tích tình cảm là quá trình tự động phân tích dữ liệu văn bản và phân loại ý kiến là tiêu cực, tích cực hoặc trung tính. Thông thường, bên cạnh việc xác định ý kiến, các hệ thống này trích xuất các thuộc tính của biểu thức, ví dụ:

Phân cực: nếu người nói bày tỏ ý kiến tích cực hoặc tiêu cực,
Chủ đề: điều đang được nói đến,
Người giữ ý kiến: người, hoặc thực thể thể hiện ý kiến.

Hiện nay, phân tích tình cảm là một chủ đề rất được quan tâm và phát triển vì nó có nhiều ứng dụng thực tế. Các công ty sử dụng phân tích tình cảm để tự động phân tích phản hồi khảo sát, đánh giá sản phẩm, nhận xét trên phương tiện truyền thông xã hội và muốn có được những hiểu biết có giá trị về thương hiệu, sản phẩm và dịch vụ của họ.

II.2 Thực hiện một bài toán Sentiment Analysis

II.2.1 Phương pháp và thuật toán phân tích tình cảm

Có nhiều phương pháp và thuật toán để thực hiện các hệ thống phân tích tình cảm, có thể được phân loại là:

Các hệ thống dựa trên quy tắc thực hiện phân tích tình cảm dựa trên một bộ quy tắc được tạo thủ công.

Hệ thống tự động dựa trên các kỹ thuật học máy để học từ dữ liệu.

Các hệ thống kết hợp cả hai cách tiếp cận dựa trên quy tắc và tự động.

II.2.1.1 Phương pháp tiếp cận dựa trên quy tắc:

Thông thường, các cách tiếp cận dựa trên quy tắc xác định một tập hợp các quy tắc trong một số loại ngôn ngữ kịch bản xác định tính chủ quan, tính phân cực hoặc chủ đề của ý kiến.

Các quy tắc có thể sử dụng nhiều loại đầu vào, chẳng hạn như:

Các kỹ thuật NLP cổ điển như *stemming*, *tokenization*, *part of speech tagging* và *parsing*.

Các tài nguyên khác, chẳng hạn như từ vựng (lexicons - danh sách các từ, thành ngữ...).

Một ví dụ cơ bản về việc thực hiện dựa trên quy tắc:

Xác định hai danh sách các từ phân cực (ví dụ: các từ phủ định như xấu, tệ nhất, xấu, v.v. và các từ tích cực như tốt, tốt nhất, đẹp, v.v.).
Đưa ra một văn bản.
Đếm số lượng từ tích cực xuất hiện trong văn bản.
Đếm số lượng từ phủ định xuất hiện trong văn bản.
Nếu số lần xuất hiện từ tích cực lớn hơn số lần xuất hiện từ phủ định sẽ trả lại một tình cảm tích cực, ngược lại, trả lại một tình cảm tiêu cực. Nếu không, trả lại trung tính.

Hệ thống này rất ngây thơ vì nó không tính đến cách các từ được kết hợp thành một chuỗi. Một quá trình xử lý tiên tiến hơn có thể được thực hiện, nhưng các hệ thống này trở nên rất phức tạp một cách nhanh chóng. Chúng có thể rất khó để duy trì vì các quy tắc mới có thể cần thiết để thêm hỗ trợ cho các biểu thức và từ vựng mới. Ngoài ra, việc thêm các quy tắc mới có thể có kết quả không mong muốn do kết quả của sự tương tác với các quy tắc trước đó. Do đó, các hệ thống này đòi hỏi các khoản đầu tư quan trọng trong việc điều chỉnh thủ công và duy trì các quy tắc.

II.2.1.2 Phương pháp tự động.

Các phương thức tự động, trái với các hệ thống dựa trên quy tắc, không dựa vào các quy tắc được tạo thủ công, mà dựa trên các kỹ thuật học máy. Nhiệm vụ phân tích tình cảm thường được mô hình hóa như một vấn đề phân loại trong đó mô hình phân loại được cung cấp đầu vào là văn bản và trả về nhãn tương ứng, ví dụ: tích cực, tiêu cực hoặc trung tính.

II.2.1.3 Phương pháp lai.

Sử dụng kết hợp của hai phương pháp trên. Thông thường, bằng cách kết hợp cả hai phương pháp, các phương pháp có thể cải thiện độ chính xác và độ chính xác.

II.2.2 Quy trình đào tạo và dự đoán

Mô hình học cách liên kết một đầu vào cụ thể (một văn bản) với đầu ra (nhãn) tương ứng dựa trên các mẫu dữ liệu được sử dụng cho đào tạo. Tiền xử lý (trích chọn đặc trưng) chuyển đầu vào văn bản thành một vectơ đặc trưng. Các cặp vectơ đặc trưng và nhãn (ví dụ: dương, âm hoặc trung tính) được đưa vào thuật toán học máy để tạo mô hình.

II.2.2.1 Trích chọn đặc trưng từ văn bản

Bước đầu tiên trong trình phân loại văn bản máy học là chuyển đổi văn bản thành biểu diễn số, thường là một vectơ. Thông thường, mỗi thành phần của vectơ biểu thị tần số của một từ hoặc cụm từ trong từ điển được xác định trước (ví dụ: từ vựng của các từ phân cực). Quá trình này được gọi là trích chọn đặc trưng hoặc vector hóa văn bản và cách tiếp cận cổ điển là *Bag-Of-Words* với tần suất của từ.

II.2.2.2 Thuật toán phân loại

Bước phân loại thường liên quan đến một mô hình thống kê như *Naïve Bayes*, *Logistic Regression*, *Support Vector Machines* hoặc *Neural Networks*:

Naïve Bayes: một nhóm các thuật toán xác suất sử dụng Định lý Bayes để dự đoán thể loại của một văn bản.

Logistic Regression: một thuật toán thông dụng trong thống kê được sử dụng để dự đoán một số giá trị (Y) được cung cấp một tập hợp các đặc trưng (X).

Support Vector Machines: một mô hình phi xác suất sử dụng biểu diễn các văn bản mẫu như điểm trong một không gian đa chiều. Các mẫu này được ánh xạ sao cho các mẫu thuộc về các *categories (hay sentiments)* khác nhau thuộc về các vùng riêng biệt của không gian đó . Sau đó, các văn bản mới được ánh xạ vào cùng một không gian đó và được dự đoán thuộc về một loại dựa trên khu vực mà chúng rơi vào.

Deep Learning: một bộ thuật toán đa dạng cố gắng bắt chước cách thức hoạt động của bộ não con người bằng cách sử dụng các mạng thần kinh nhân tạo để xử lý dữ liệu.

II.2.2.3 Phân tích và đánh giá cho Sentiment Analysis

Có nhiều cách để đánh giá hiệu suất của mô hình cho Sentiment Analysis và để hiểu mô hình Sentiment Analysis chính xác đến mức nào. Một trong những phương pháp được sử dụng thường xuyên nhất là *cross-validation*.

Cross-validation chia bộ dữ liệu training thành k lớp thông thường lấy k-1 lớp dữ liệu cho huấn luyện và 1 lớp dữ liệu cho *validation*, sử dụng các lớp training để training bộ phân loại, và kiểm tra nó dựa trên lớp *validation* để có được số liệu về hiệu suất của mô hình. Quá trình được lặp lại nhiều lần và tính trung bình cho mỗi số liệu được tính toán.

Nếu tập validation luôn giống nhau, có thể dẫn đến overfitting cho bộ dữ liệu đó, có nghĩa là mô hình được điều chỉnh để dự đoán tốt nhất cho bộ dữ liệu đang có, nhưng sẽ thất bại trong việc phân tích một dữ liệu hoàn toàn mới. *Cross-validation* giúp ngăn chặn điều đó.

II.2.3 Những điểm khó khăn trong bài toán Sentiment Analysis

II.2.3.1 Tính chủ quan và ngữ điệu:

Xác định tính chủ quan hoặc khách quan của văn bản là yếu tố quan trọng trong phân tích ngữ điệu mà văn bản thể hiện. Về cơ bản, văn bản có tính khách quan thì thể hiện thông tin mà không thể hiện tình cảm, hay tính tình cảm của nó là trung tính.

II.2.3.2 Ngữ cảnh và tính phân cực:

Tất cả mọi phát ngôn đều có ngữ cảnh của phát ngôn, và ngữ cảnh đó ảnh hưởng đến ý nghĩa của phát ngôn. Cùng một phát ngôn, đặt trong ngữ cảnh khác nhau có thể dẫn đến ý nghĩa khác nhau. Phân tích tình cảm mà không có bối cảnh trở nên khá khó khăn. Tuy nhiên, máy móc không thể tìm hiểu về bối cảnh nếu chúng không được đề cập rõ ràng. Một trong những vấn đề nảy sinh từ bối cảnh là những thay đổi về tính phân cực. Ví dụ, trong một cuộc khảo sát, câu trả lời nhận được là:

Mọi thứ.

Không gì cả.

Nếu câu hỏi là: *Bạn thích điều gì trong sự kiện này?,* phản hồi đầu tiên sẽ là tích cực và phản hồi thứ hai sẽ là tiêu cực. Trong khi đó, nếu câu hỏi là: Bạn không thích điều gì trong sự kiện này? *Sự* tiêu cực trong câu hỏi sẽ làm cho phân tích tình cảm thay đổi hoàn toàn.

II.2.3.3 Châm biếm và mỉa mai

Sự khác biệt giữa nghĩa đen và nghĩa bóng khi phát ngôn có tính mỉa mai hay châm biếm thường thay đổi tình cảm tích cực thành tiêu cực trong khi tình cảm tiêu cực hoặc trung tính có thể được thay đổi thành tích cực. Tuy nhiên, việc phát hiện sự mỉa mai hoặc châm biếm cần có sự phân tích tốt về bối cảnh mà các văn bản được tạo ra và do đó, thực sự rất khó để phát hiện tự động.

II.2.3.4 Biểu tượng cảm xúc (Emojis)

Có hai loại biểu tượng cảm xúc. Biểu tượng cảm xúc phương Tây (ví dụ: D) được mã hóa chỉ trong một ký tự hoặc kết hợp một vài trong số chúng trong khi biểu tượng cảm xúc phương Đông (ví dụ \ _ (ツ) _ /) là một sự kết hợp dài hơn của các ký tự theo hàng ngang. Trong các văn bản mạng xã hội, biểu tượng cảm xúc đóng một vai trò trong thể hiện tính tình cảm của văn bản.

Các emojis đòi hỏi nhiều về tiền xử lý cho văn bản, và có thể được tính như một đặc trưng tốt cho thể hiện cảm xúc của băn bản.

II.2.4 **Ý nghĩa:**

Tại sao phân tích tình cảm là quan trọng?

Khoảng 80% dữ liệu của thế giới là không có cấu trúc và không được tổ chức theo cách được xác định trước. Hầu hết điều này đến từ dữ liệu văn bản, như email, vé hỗ trợ, trò chuyện, phương tiện truyền thông xã hội, khảo sát, bài viết và tài liệu. Những văn bản này thường khó, tốn thời gian và tốn kém để phân tích, hiểu và sắp xếp.

Hệ thống phân tích tình cảm cho phép các công ty hiểu được biển văn bản phi cấu trúc này bằng cách tự động hóa các quy trình kinh doanh, hiểu biết sâu sắc và tiết kiệm hàng giờ xử lý dữ liệu thủ công, nói cách khác, bằng cách làm cho các nhóm hiệu quả hơn.

Một số ưu điểm của phân tích tình cảm bao gồm:

Khả năng mở rộng quy mô:

Phân tích tình cảm cho phép xử lý dữ liệu theo quy mô một cách hiệu quả và tiết kiệm chi phí so với xử lý thủ công.

Phân tích thời gian thực:

Chúng ta có thể sử dụng phân tích tình cảm để xác định thông tin quan trọng cho phép nhận thức tình huống trong các tìnhhuống cụ thể trong thời gian thực, ví dụ như một cuộc khủng hoảng truyền thông của một doanh nghiệp lớn. Một hệ thống phân tích tình cảm có thể giúp xác định ngay các loại tình huống này và hành động.

Tiêu chí thống nhất:

Con người không có các tiêu chí rõ ràng để đánh giá tình cảm của một đoạn văn bản. Đánh giá văn bản là một công việc chủ quan chịu ảnh hưởng lớn từ kinh nghiệm, suy nghĩ và niềm tin cá nhân. Bằng cách sử dụng một hệ thống phân tích tình cảm tập trung, các công ty có thể áp dụng các tiêu chí tương tự cho tất cả dữ liệu của họ. Điều này giúp giảm lỗi và cải thiện tính nhất quán dữ liệu.

Phân tích tình cảm là một công việc khó khăn ngay cả đối với con người. Vậy nên, phân tích tình cảm tự động có thể không chính xác như các loại phân loại khác. Tuy nhiên, cho dù đôi khi các dự đoán phân tích tình cảm sẽ bị sai, nhưng bằng cách sử dụng phân tích tình cảm, có xác suất 70-80% số văn bản được phân tích đúng trong một thời gian và chi phí nhỏ hơn rất nhiều so với phân tích thủ công.

III Course’s Project: Chương trình phân tích tình cảm cho bình luận về sản phẩm

Project thực hiện phân tích tình cảm cho hai bộ dữ liệu bình luận thuộc hai ngôn ngữ: tiếng Anh và tiếng Việt. Hai bộ dữ liệu tương ứng là:

Bộ dữ liệu tiếng Anh:

Bộ dữ liệu aclImdb - Large Movie Review Dataset[1]: Bộ dữ liệu cho phân tích tình cảm nhị phân, ở đó, dữ liệu được gán nhãn tích cực hoặc tiêu cực. Dữ liệu bao gồm 25 000 mẫu đánh giá phim đã phân loại dùng cho training, và 25 000 mẫu khác dành cho testing mô hình.

Bộ dữ liệu tiếng Việt:

Bộ dữ liệu Sentiment Analysis VLSP 2016:

Chỉ chứa các đánh giá có quan điểm cá nhân.

1 http://ai.stanford.edu/~amaas/data/sentiment/

Dữ liệu thường là những bình luận ngắn, chứa quan điểm về một đối tượng, không giới hạn về số khía cạnh của đối tượng được đề cập đến.

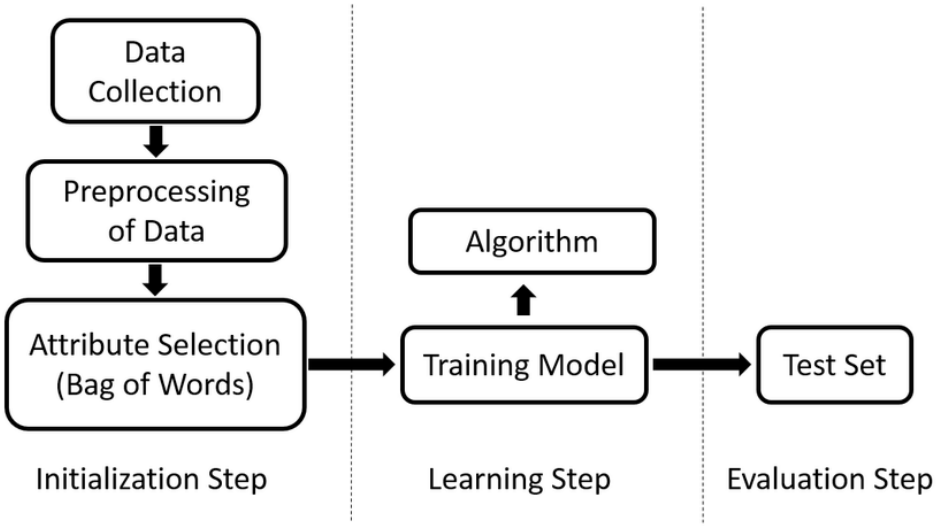
: Mọi dữ liệu đều được gán nhãn đầy đủ (nhãn gồm: tích cực, tiêu cực, trung tính)

Bộ dữ liệu chỉ chứa các dữ liệu thực được thu thập từ mạng xã hội, không phải dữ liệu tự tạo.

Một dữ liệu được gán nhãn trung tính có thể vì không thể xác định rõ ràng tính tích cực hay tiêu cực.

Dữ liệu trung tính có thể được tính cho bình luận có chứa cả tính tích cực và tiêu cực, nhưng khi gộp lại, chúng thể hiện sự trung tính.

Project nhằm mục tiêu áp dụng mô hình học máy Logistic Regression cho bài toán phân tích tình cảm cho bình luận về sản phẩm. Về cơ bản, quy trình giải quyết bài toán tuân theo sơ đồ dưới:



Trong đó:

Quá trình Data Collection, thay vì crawling dữ liệu trực tiếp từ website, chương trình sử dụng các bộ dữ liệu mở sẵn có, là hai bộ dữ liệu vừa được đề cập phía trên.

Quá trình Preprocess Data bao gồm:

Tách từ. Đối với dữ liệu tiếng Anh, việc tách từ khá đơn giản, thông qua các dấu câu và dấu cách. Điều đó trở nên phức tạp hơn nhiều khi làm việc với dữ liệu tiếng Việt. Để tách từ cho tiếng Việt, chương trình sử dụng bộ công cụ VnCoreNLP, là một bộ công cụ mã nguồn mở chuyên dụng cho xử lý ngôn ngữ tiếng Việt, cho phép tách từ tiếng Việt, gán nhãn từ loại, nhận dạng thực thể có tên, phân

tích phụ thuộc… với độ chính xác cao, nhanh, dễ sử dụng. Thư viện được viết bằng java, bản cài đặt trong project là phần vỏ python giúp giao tiếp với chương trình java.

Lọc các ký tự đặc biệt không phải chữ cái, chữ số, dấu !, dấu ? Đặc điểm của hai bộ dữ liệu được sử dụng là chúng là dữ liệu đã được làm "sạch", gần với văn bản thông thường, không có quá nhiều bất quy tắc, phá cách trong văn bản. Do đó, quá trình tiền xử lý tương đối đơn giản và không tiêu tốn nhiều công sức. Với những văn bản mạng xã hội đặc thù, việc sử dụng nhiều kí tự đặc biệt, bất quy tắc, phá cách, biểu tượng cảm xúc (emojis) có thể làm cho quá trình tiền xử lý văn bản trở nên phức tạp hơn.

Chia bộ dataset ngẫu nhiên thành hai phần, training data và testing data, với tỉ lệ 2 : 1.

Attribute Selection: Trích chọn đặc trưng và vector hóa bình luận sử dụng phương pháp Bag-Of-Words, có thể sử dụng tần suất của từ trong quá trình vector hóa, hoặc sử dụng phương pháp TF-IDF.

Training Model: Thuật toán học máy được áp dụng ở đây là thuật toán Logistic Regression, áp dụng cross-validation với 5-folds để đánh giá mức khái quát trong độ chính xác của mô hình. Thuật toán sử dụng hàm loss-function là Cross-Entropy và tối ưu bằng Stochastic Gradient Descent, hiệu chỉnh bằng L2 Regularization.

Kết quả:

Đối với chương trình phân tích tình cảm cho tập dữ liệu tiếng Anh, chương trình đạt độ chính xác 88% trên tập kiểm tra, với BoW là tập toàn bộ các từ trong kho ngữ liệu (corpus). Với chương trình phân tích cho tiếng Việt, thông qua nhiều phương án trích chọn đặc trưng, xây dựng BagOfWords khác nhau, cho kết quả sau:

Phương pháp tiếp cận	Độ chính xác
BoW = Tất cả các từ khác nhau trong văn bản. Vector bình luận là vector nhị phân (chỉ xét đến sự hiện diện của từ)	65%
BoW = Tất cả các từ khác nhau trong văn bản. Vector bình luận là vector tần suất của từ xuất hiện trong văn bản	63-64%
BoW = Các tính từ trong VnEmoLex Vector bình luận là vector tần suất của từ xuất hiện trong văn bản	60-62%
BoW = Tất cả các từ khác nhau trong văn bản. Vector bình luận là vector tần suất của từ xuất hiện trong văn bản, tăng tần suất lên N lần cho các tính từ trong VnEmoLex	62-64%
TF-IDF	NaN

Độ chính xác của phương án I và II gần tương đương nhau. Ở phương án III, khi chỉ sử dụng các tính từ làm đặc trưng để vector hóa bình luận, độ chính xác giảm xuống. Vấn đề này là do các từ khác trong bình luận (không phải tính từ mang tính tích cực, tiêu cực) cũng có đóng góp, dù nhỏ hơn, vào quá trình dự đoán nhãn của bình luận. Ở phương án IV, độ chính xác tăng lên khi ta gia tăng giá trị của đặc trưng tính từ thể hiện độ tích cực, tiêu cực trong vector bình luận. Đó là do các tính từ này đóng góp nhiều vào quá trình mô hình quyết định nhãn của từ. Phương án sử dụng TF-IDF để xử lý vector bình luận, nhằm giảm giá trị của các từ phổ biến trong toàn ngữ liệu, và gia tăng độ quan trọng của các từ đặc trưng trong mỗi lớp văn bản, có tiềm năng gia tăng độ chính xác.

IV Phụ lục:

IV.1 Tài liệu tham khảo:

Chapter 5 - Speech and Language Processing 3rd edittion - Daniel Jurafsky & James H. Martin

Large Movie Review Dataset - Stanford AI Lab

http://vlsp.org.vn/ - 2019

https://github.com/vncorenlp/VnCoreNLP – 2019

VnEmoLex: A Vietnamese emotion lexicon for sentiment intensity analysis – KTLab

IV.2 Các thư viện, gói chương trình sử dụng trong chương trình phân tích:

VNCoreNLP (cho words tokenize).

Scikit Learn (cho Logistic Regression).

Numpy/Scipy (cho Sparse Matrix).

VnEmoLex (bộ dữ liệu về tính từ thể hiện sắc thái)

Pandas