TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN

BÀI TIỂU LUẬN

XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Đề tài:

Phân loại tin tức tiếng Việt theo chủ đề

Sinh viên thực hiện: **Thân Trung Sơn**
Lớp: **CNTT K21CLC**
Giảng viên hướng dẫn: **TS. Trần Văn Khánh**

Thái Nguyên, năm 2025

Mục lục

LỜI MỞ ĐẦU

Trong bối cảnh bùng nổ thông tin hiện nay, lượng tin tức được sản xuất và lan truyền với tốc độ nhanh chóng trên nhiều nền tảng khác nhau. Việc phân loại tin tức một cách tự động không chỉ giúp tổ chức và truy xuất thông tin dễ dàng hơn mà còn hỗ trợ các ứng dụng như lọc tin theo chủ đề, phát hiện tin giả, hoặc đề xuất nội dung phù hợp cho người dùng. Mô hình học máy đóng vai trò quan trọng trong việc tự động hóa quá trình phân loại tin tức. Thay vì dựa vào các quy tắc thủ công tốn kém và kém linh hoạt, các thuật toán học máy có thể học từ dữ liệu và cải thiện độ chính xác theo thời gian. Trong bài viết này, chúng tôi trình bày quy trình xây dựng mô hình học máy để phân loại tin tức thành các danh mục khác nhau, từ thu thập dữ liệu, tiền xử lý văn bản, lựa chọn mô hình đến đánh giá hiệu suất. Hy vọng rằng nghiên cứu này sẽ mang lại cái nhìn tổng quan về cách áp dụng học máy trong xử lý ngôn ngữ tự nhiên (NLP) và mở ra hướng phát triển các hệ thống phân loại thông minh hơn trong tương lai.

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1. Machine learning

Machine Learning là một tập con của trí tuệ nhân tạo. Machine Learning là một lĩnh vực nhỏ trong khoa học máy tính, có khả năng tự học hỏi dựa trên dữ liệu được đưa vào mà không cần phải được lập trình cụ thể (Machine Learning is the subfield of computer science, that "gives computers the ability to learn without being explicitly programmed" – Wikipedia)

Hình 1.1. Mối quan hệ giữa học sâu, học máy và trí tuệ nhân tạo

Machine learning sử dụng các thuật toán để máy tính có thể học từ tập dữ liệu được cung cấp, từ đó, máy tính sẽ có thể thực hiện được các công việc theo yêu cầu của mỗi bài toán chi tiết dựa trên những gì đã học được. Vì vậy, máy tính có khả năng cải thiện chính nó dựa vào các tập dữ liệu mẫu được đưa vào mẫu (training data) hoặc dựa vào kinh nghiệm (những gì đã được học). Các mô hình tự học có khả năng dự đoán kết quả và phân loại thông tin mà không cần sự can thiệp của con người.

Cả thuật toán học máy sử dụng mạng lưới thần kinh để "học" từ lượng dữ liệu khổng lồ. Các mạng lưới thần kinh này là các cấu trúc có lập trình được mô phỏng theo quá trình ra quyết định của bộ não con người. Chúng bao gồm các lớp nút được kết nối với nhau để trích xuất các đặc điểm từ dữ liệu và đưa ra dự đoán về những gì dữ liệu thể hiện. Các thuật toán học máy cổ điển sử dụng mạng thần kinh với lớp đầu vào, một hoặc hai lớp 'ẩn' và lớp đầu ra.

Machine learning có mối quan hệ rất mật thiết đối với statistics (thống kê). Machine learning sử dụng các mô hình thống kê để "ghi nhớ" lại sự phân bố của dữ liệu. Tuy nhiên, không đơn thuần là ghi nhớ, machine learning phải có khả năng tổng quát hóa những gì đã được nhìn thấy và đưa ra dự đoán cho những trường hợp chưa được nhìn thấy. Đỉnh cao của machine learning sẽ là mô phỏng được khả năng tổng quát hóa và suy luận của con người để từ đó, đưa ra kết quả cho những trường hợp chưa có trong tập dữ liệu mẫu.

Từ đó, ta nhận thấy rằng khả năng phán đoán chính xác của các mô hình học máy phụ thuộc khá nhiều và tập dữ liệu mẫu dùng để training. Từ một tập dữ liệu đơn giản, mô hình học máy đã có thể học và từ đó phán đoán một giá trị mới bất kỳ. Tuy nhiên, khi ta cho mô hình máy học một tập dữ liệu lớn hơn và chi tiết hơn về các trường hợp có thể xảy ra của dữ liệu thì chắc chắn rằng khả năng học hỏi và phán đoán kết quả của mô hình sẽ chính xác hơn rất nhiều so với tập dữ liệu nhỏ. Bên cạnh việc chuẩn bị tập dữ liệu phù hợp và đủ lớn thì một phần quan trọng không thể thiếu chính là thuật toán mà mô hình học máy đó sử dụng. Với một thuật toán tốt, có thể biểu diễn tốt hơn tập dữ liệu học thì chắc chắn rằng mô hình học máy được đào tạo ra có thể thực hiện tốt hơn nhiệm vụ phán đoán kết quả của một giá trị mới.

1.1.1. Phân loại machine learning

1.1.1.1. Học có giám sát

Học có giám sát là một loại học máy sử dụng các tập dữ liệu được gắn nhãn để huấn luyện các thuật toán nhằm dự đoán kết quả và nhận ra các mẫu. Không giống như học không giám sát , các thuật toán học có giám sát được đào tạo có gắn nhãn để tìm hiểu mối quan hệ giữa đầu vào và đầu ra.

Dữ liệu được sử dụng trong học có giám sát được gắn nhãn - nghĩa là nó chứa các ví dụ về cả đầu vào (được gọi là đặc trưng) và đầu ra chính xác (nhãn). Các thuật toán phân tích một tập dữ liệu lớn gồm các cặp huấn luyện này để suy ra giá trị đầu ra mong muốn khi được yêu cầu đưa ra dự đoán về dữ liệu mới. Sau khi mô hình đã được huấn luyện và thử nghiệm, bạn có thể sử dụng nó để đưa ra dự đoán về dữ liệu chưa biết dựa trên kiến thức đã học trước đó.

Học có giám sát trong học máy thường được chia thành hai loại: classification (phân loại) và regression (hồi quy).

Hồi quy (Regression): Khi kết quả đầu ra là một giá trị liên tục, chúng ta sử dụng thuật toán hồi quy. Ví dụ: dự đoán giá nhà, dự đoán doanh số.

Phân loại (Classification): Khi kết quả đầu ra là một giá trị rời rạc, chúng ta sử dụng thuật toán phân loại. Ví dụ: phân loại văn bản, nhận dạng khuôn mặt.

1.1.1.2. Học không giám sát

Học tập không giám sát trong trí tuệ nhân tạo là một loại học máy học từ dữ liệu mà dữ liệu đầu vào không có nhãn. Thuật toán unsupervised learning sẽ dựa vào cấu trúc của dữ liệu để thực hiện một công việc nào đó, ví dụ như phân nhóm (clustering) hoặc giảm số chiều của dữ liệu (dimension reduction) để thuận tiện trong việc lưu trữ và tính toán.

Có ba loại học máy không giám sát: phân cụm, quy tắc kết hợp và giảm kích thước.

Phân cụm (Clustering): Phân cụm là một kỹ thuật để khám phá dữ liệu thô, chưa được gắn nhãn và chia nó thành các nhóm (hoặc cụm) dựa trên những điểm tương đồng hoặc khác biệt.

Liên kết (Association): Khai thác quy tắc kết hợp là một cách tiếp cận dựa trên quy tắc để khám phá mối quan hệ thú vị giữa các điểm dữ liệu trong bộ dữ liệu lớn. Các thuật toán học không giám sát tìm kiếm các liên kết nếu – thì thường xuyên còn gọi là quy tắc để khám phá các mối tương quan và sự xuất hiện đồng thời trong dữ liệu cũng như các kết nối khác nhau giữa các đối tượng dữ liệu.

Giảm kích thước (Dimensionality reduction): Giảm kích thước là một khía cạnh thiết yếu của học máy vì nó giúp mang lại kết quả chính xác hơn cho các tập dữ liệu lớn bằng cách giúp giảm số lượng tính năng được xem xét kỹ lưỡng, giúp dữ liệu dễ quản lý hơn mà không cần loại bỏ bất kỳ phần không thể thiếu nào. Điều này giúp tránh được vấn đề về chiều và tạo ra một mô hình dự đoán phù hợp hơn.

1.1.1.3. Học tăng cường

Học tăng cường (RL) là kỹ thuật máy học (ML) giúp đào tạo phần mềm đưa ra quyết định nhằm thu về kết quả tối ưu nhất. Kỹ thuật này bắt chước quy trình học thử và sai mà con người sử dụng để đạt được mục tiêu đã đặt ra. RL giúp phần mềm tăng cường các hành động hướng tới mục tiêu, đồng thời bỏ qua các hành động làm xao lãng mục tiêu.

Học tăng cường khác với học có giám sát ở chỗ trong học có giám sát, dữ liệu huấn luyện có khóa trả lời nên mô hình được huấn luyện với chính câu trả lời đúng trong khi học tăng cường, không có câu trả lời nhưng tác nhân tăng cường quyết định phải làm gì thực hiện nhiệm vụ được giao. Trong trường hợp không có tập dữ liệu huấn luyện, nó buộc phải học hỏi từ kinh nghiệm của mình.

1.1.2. Một số thuật toán học máy cơ bản

1.1.2.1. Hồi quy

Hồi quy, một phương pháp thống kê, phân tích mối quan hệ giữa các biến phụ thuộc và biến độc lập, cho phép dự đoán thông qua các mô hình hồi quy khác nhau.

Regression hay Hồi quy là một phương pháp thống kê được sử dụng để phân tích mối quan hệ giữa một biến phụ thuộc (biến mục tiêu) và một hoặc nhiều biến độc lập (biến dự đoán). Mục tiêu là xác định hàm phù hợp nhất mô tả mối liên hệ giữa các biến này.

Đây là một kỹ thuật học máy có giám sát, được sử dụng để dự đoán giá trị của biến phụ thuộc cho dữ liệu mới, chưa được nhìn thấy. Nó mô hình hóa mối quan hệ giữa các tính năng đầu vào và biến mục tiêu, cho phép ước tính hoặc dự đoán các giá trị số.

a) Linear Regression

Hồi quy tuyến tính là một trong những mô hình thống kê đơn giản và được sử dụng rộng rãi nhất. Điều này giả định rằng có mối quan hệ tuyến tính giữa các biến độc lập và biến phụ thuộc. Điều này có nghĩa là sự thay đổi của biến phụ thuộc tỷ lệ thuận với sự thay đổi của biến độc lập.

b) Logistic Regression

Hồi quy Logistic là một mô hình thống kê được sử dụng để phân loại nhị phân, tức dự đoán một đối tượng thuộc vào một trong hai nhóm. Hồi quy Logistic làm việc dựa trên nguyên tắc của hàm sigmoid – một hàm phi tuyến tự chuyển đầu vào của nó thành xác suất thuộc về một trong hai lớp nhị phân.

c) Stepwise regression

Phương pháp hồi quy từng bước (stepwise regression) là một phương pháp phân tích hồi quy được sử dụng để xác định các biến độc lập quan trọng nhất trong việc dự đoán biến phụ thuộc. Phương pháp này tiến hành bổ sung các biến một cách tuần tự vào mô hình hồi quy, đánh giá ảnh hưởng và khả năng giải thích của từng biến được bổ sung. Quá trình hồi quy từng bước bắt đầu với một mô hình hồi quy đơn giản chứa một biến độc lập duy nhất. Sau đó, các biến khác được bổ sung vào mô hình dựa trên các tiêu chí nhất định, như mức độ quan trọng của biến đối với biến phụ thuộc, sự cải thiện của mô hình sau khi thêm biến, và các chỉ số thống kê như giá trị p. Phương pháp hồi quy từng bước (stepwise regression) là một kỹ thuật thống kê được sử dụng để xây dựng một mô hình hồi quy tuyến tính bằng cách chọn ra tập hợp các biến độc lập quan trọng nhất để dự đoán biến phụ thuộc. Phương pháp này thường được áp dụng trong việc lựa chọn biến trong mô hình hồi quy khi có nhiều biến độc lập có thể ảnh hưởng đến biến phụ thuộc.

Quá trình tiến hành theo hai hướng: tiến và lùi. Trong hướng tiến, một biến độc lập được bổ sung vào mô hình ở mỗi bước và kiểm tra xem biến đó có cải thiện khả năng giải thích của mô hình hay không. Nếu biến đó đạt được ngưỡng quy định, nó sẽ được giữ lại trong mô hình. Trong hướng lùi, các biến độc lập được loại bỏ khỏi mô hình một cách tuần tự để kiểm tra xem loại bỏ biến đó có làm giảm khả năng giải thích của mô hình hay không. Phương pháp hồi quy từng bước cho phép chúng ta tìm ra một mô hình hồi quy tối ưu, chỉ chứa các biến quan trọng nhất và đóng góp đáng kể vào việc dự đoán biến phụ thuộc. Nó giúp giảm chiều của mô hình và loại bỏ các biến không cần thiết, từ đó tăng tính hiệu quả và khả năng giải thích của mô hình hồi quy.

Tuy nhiên, cần lưu ý rằng phương pháp hồi quy từng bước cũng có nhược điểm, bao gồm khả năng tạo ra mô hình quá đơn giản hoặc quá phức tạp, vấn đề về đa cộng tuyến (multicollinearity) giữa các biến độc lập, và nguy cơ xảy ra sai sót thống kê. Do đó, việc áp dụng phương pháp này cần cân nhắc kỹ lưỡng và kết hợp với các phương pháp khác để đánh giá mô hình hồi quy một cách toàn diện.

1.1.2.2. Classification

Classification là một quá trình phân loại dữ liệu hoặc đối tượng thành các lớp hoặc danh mục được xác định trước dựa trên các tính năng hoặc thuộc tính của chúng.

Phân loại là một loại kỹ thuật học có giám sát trong đó thuật toán được đào tạo trên tập dữ liệu được gắn nhãn để dự đoán lớp hoặc danh mục dữ liệu mới, chưa được nhìn thấy.

Mục tiêu chính của học máy phân loại là xây dựng một mô hình có thể gán nhãn hoặc danh mục chính xác cho một quan sát mới dựa trên các tính năng của nó.

a) Linear Classification.

Linear Classifier là một loại mô hình học máy được sử dụng trong bài toán phân loại. Mô hình này

hoạt động dựa trên nguyên tắc của hàm tuyến tính để tạo ra một ranh giới phẳng (hyperplane) trong không gian đặc trưng, phân chia các điểm dữ liệu thành các lớp khác nhau.

Cụ thể, trong một không gian đặc trưng nhiều chiều, mỗi điểm dữ liệu được biểu diễn dưới dạng một vector đặc trưng. Một linear classifier sẽ tính toán một tổ hợp tuyến tính của các đặc trưng này và sử dụng kết quả để quyết định lớp mà điểm dữ liệu thuộc về. Thường thì, một hàm kích hoạt (ví dụ: hàm softmax cho bài toán phân loại nhiều lớp hoặc hàm sigmoid cho bài toán phân loại nhị phân) được sử dụng để chuyển đổi đầu ra của tổ hợp tuyến tính thành xác suất thuộc về từng lớp.

b) K-Nearest Neighbor(KNN)

Kernel Support Vector Machine (Kernel SVM) là một biến thể của Support Vector Machine (SVM) được sử dụng để giải quyết các bài toán phân loại không thể phân chia tuyến tính. Trong SVM, mục tiêu là tìm ra một ranh giới phẳng (hyperplane) phân chia các lớp sao cho khoảng cách từ các điểm dữ liệu gần nhất đến hyperplane là lớn nhất có thể. Tuy nhiên, đôi khi các lớp dữ liệu không thể được phân chia hoàn toàn bằng một hyperplane tuyến tính.

Kernel SVM giải quyết vấn đề này bằng cách sử dụng một kỹ thuật gọi là ”kernel trick”. Thay vì phân loại trực tiếp trong không gian đặc trưng ban đầu, kernel SVM ánh xạ dữ liệu vào một không gian đặc trưng cao hơn (thường là không gian nhiều chiều hơn) thông qua một hàm kernel. Trong không gian đặc trưng mới này, dữ liệu có thể được phân chia tuyến tính bằng một hyperplane.

c) Support Vector Machine (SVM)

SVM là một thuật toán học máy có giám sát được sử dụng cho cả phân loại và hồi quy. Mặt phẳng quyết định (siêu phẳng) là mặt phẳng phân tách giữa một tập hợp các đối tượng có các thành viên lớp khác nhau. Mục tiêu chính của thuật toán SVM là tìm siêu phẳng tối ưu trong không gian N chiều có thể phân tách các điểm dữ liệu trong các lớp khác nhau trong không gian đặc trưng. Siêu phẳng cố gắng sao cho khoảng cách giữa các điểm gần nhất của các lớp khác nhau phải lớn nhất có thể. Một lựa chọn hợp lý được coi là siêu phẳng tốt nhất là siêu phẳng thể hiện khoảng cách hoặc lề lớn nhất giữa hai lớp.

Ngoài ra còn các thuật toán khác như:

- Instance-based Algorithms
- Regularization Algorithms
- Bayesian Algorithms
- Clustering Algorithms
- Artificial Neural Network Algorithms
- Dimensionality Reduction Algorithms
- Ensemble Algorithms

1.2. Deep learning

1.2.1. Khái niệm

Deep learning là một phần của lĩnh vực học máy, nhưng nó khác biệt ở một số điểm quan trọng. Deep learning cho phép máy tính giải quyết những vấn đề phức tạp bằng cách ”học” từ một lượng lớn dữ liệu. Điều này đã thúc đẩy nhiều ứng dụng và dịch vụ trí tuệ nhân tạo nhằm nâng cao tự động hóa và cải thiện các tác vụ phân tích và vật lý mà không cần sự can thiệp của con người. Công nghệ học sâu thường là cốt lõi của nhiều sản phẩm và dịch vụ hàng ngày như trợ lý kỹ thuật số, điều khiển TV bằng giọng nói từ xa, phát hiện gian lận thẻ tín dụng, ô tô tự lái và nhiều ứng dụng khác. Điều này làm tăng tính tự động hóa và cải thiện trải nghiệm người dùng trong nhiều lĩnh vực khác nhau.

1.2.2. Phân biệt học sâu với học máy

Các thuật toán học máy thường tận dụng dữ liệu có cấu trúc và được gắn nhãn để đưa ra dự đoán. Các tính năng cụ thể được xác định từ dữ liệu và tổ chức thành các bảng. Thông thường, dữ liệu phải trải qua một số xử lý trước để đảm bảo định dạng cấu trúc.

Trong khi đó, học sâu thường loại bỏ một số bước xử lý trước dữ liệu. Các thuật toán học sâu có khả năng nhập và xử lý dữ liệu phi cấu trúc như văn bản và hình ảnh. Hơn nữa, chúng tự động hóa việc trích xuất tính năng, giảm bớt sự tham gia của con người trong quá trình này.

Ví dụ, khi phân loại các loài động vật từ bộ ảnh, các thuật toán học sâu có thể tự động xác định các đặc điểm quan trọng như tai mà không cần sự hỗ trợ của con người. Thông qua các quá trình giảm độ dốc và lan truyền ngược, các thuật toán học sâu tự điều chỉnh mô hình của mình để phù hợp với dữ liệu, đồng thời đưa ra dự đoán với độ chính xác cao.

Cả học máy và học sâu đều có khả năng thực hiện các loại học khác nhau, bao gồm học có giám sát, học không giám sát và học bán giám sát. Điều này giúp chúng phù hợp với nhiều loại vấn đề và môi trường khác nhau.

1.2.3. Linear Regression

Linear regression là một mô hình thống kê đơn giản được sử dụng để dự đoán giá trị của một biến phụ thuộc dựa trên một hoặc nhiều biến độc lập. Nó giả định mối quan hệ tuyến tính giữa các biến độc lập và biến phụ thuộc. Trong mô hình linear regression, chúng ta cố gắng tìm ra một đường thẳng tốt nhất để phù hợp với dữ liệu.

Mô hình linear regression có dạng toán học như sau:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n + \varepsilon$$

Trong đó:

- y: là biến phụ thuộc (biến dự đoán).
- $x_1, x_2, \ldots, x_n$: là các biến độc lập.
- β_0: là hệ số chặn.
- $\beta_1, \beta_2, \ldots, \beta_n$: là các hệ số của các biến độc lập.
- ε: là sai số ngẫu nhiên.

Trong *linear regression*, *loss function* (hàm mất mát) thường được chọn là *Mean Squared Error* (MSE), hay còn được gọi là *Quadratic Loss*. MSE là phổ biến và được sử dụng rộng rãi trong các bài toán hồi quy (*regression*), bao gồm cả *linear regression*.

Công thức của hàm mất mát cho *linear regression* được biểu diễn như sau:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2.$$

Trong đó:

- N: là số lượng mẫu trong tập dữ liệu.
- y_i: là giá trị thực tế của mẫu thứ i.
- $\hat{y}_i$: là giá trị dự đoán của mẫu thứ i.
- $\left(y_i - \hat{y}_i\right)^2$: là bình phương của sai số giữa giá trị thực tế và giá trị dự đoán của mỗi mẫu.

Mục tiêu của *linear regression* là tìm ra các tham số $\boldsymbol{\beta}$ sao cho MSE nhỏ nhất, tức là tìm ra đường thẳng (hoặc siêu phẳng trong không gian nhiều chiều) phù hợp nhất với dữ liệu. Điều này thường được thực hiện bằng các phương pháp tối ưu hoá như *Gradient Descent* để tối thiểu hoá MSE.

Mô hình linear regression thường được sử dụng để dự đoán giá trị số, như dự đoán giá nhà dựa trên diện tích, dự đoán doanh số bán hàng dựa trên quảng cáo, hoặc bất kỳ tình huống nào có mối quan hệ tuyến tính giữa biến đầu vào và biến đầu ra.

1.2.4. Gradient descent

Gradient Descent là một thuật toán tối ưu hóa được sử dụng để cập nhật các tham số của một mô hình máy học như mạng neural, linear regression, logistic regression, và nhiều mô hình khác, dựa trên đạo hàm của một hàm mất mát (loss function) liên quan đến các tham số đó.

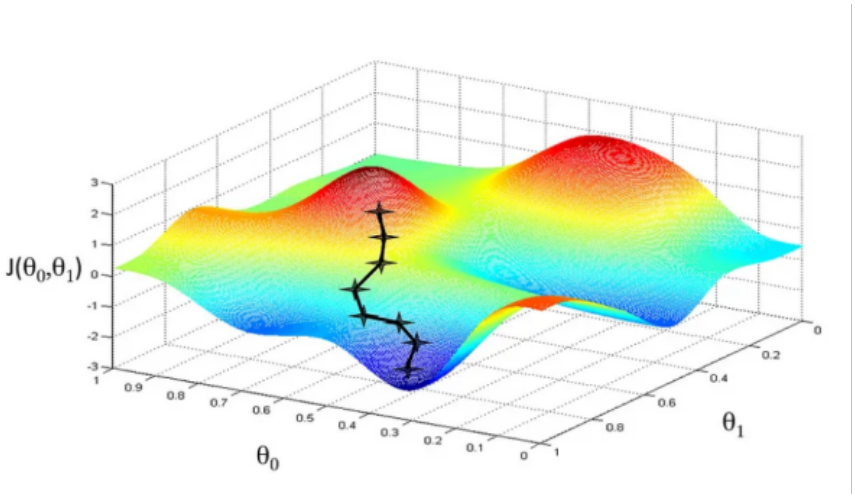
Ý tưởng chính của Gradient Descent là điều chỉnh các tham số của mô hình theo hướng ngược lại của đạo hàm của hàm mất mát, với một tỷ lệ học (learning rate) nhất định. Mục tiêu là di chuyển từ vị trí hiện tại trên bề mặt hàm mất mát đến điểm cực tiểu của nó, nơi hàm mất mát đạt được giá trị nhỏ nhất.

Công thức tổng quát của *gradient descent*:

$$\theta_j := \theta_j - \alpha \frac{\partial J(\boldsymbol{\theta})}{\partial \theta_j}$$

Trong đó:

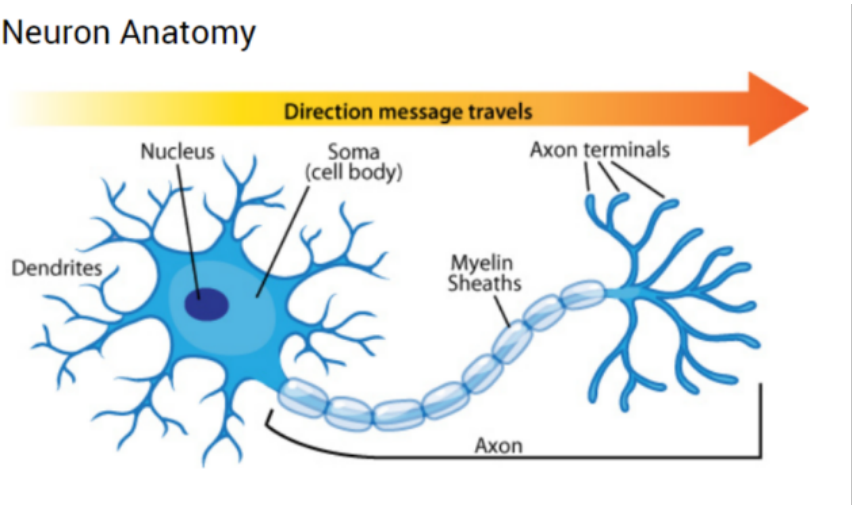
- θ_j: tham số thứ j của mô hình.
- α: tỷ lệ học (*learning rate*).
- $J(\boldsymbol{\theta})$: hàm mất mát (*loss function*).
- $\frac{\partial J(\boldsymbol{\theta})}{\partial \theta_j}$: đạo hàm của $J(\boldsymbol{\theta})$ theo θ_j.



Hình 1.2. Gradient descent

1.2.5. Neural network

Mạng neural là một mô hình tính toán lấy cảm hứng từ cách mà não của con người hoạt động. Nó được sử dụng để giải quyết các vấn đề như phân loại, dự đoán và nhận dạng dữ liệu trong lĩnh vực trí tuệ nhân tạo và học máy. Phương thức này tạo ra một hệ thống thích ứng được máy tính sử dụng để học hỏi từ sai lầm của chúng và liên tục cải thiện



Hình 1.3. Cấu tạo của Neuron

Mạng nơ - ron nhân tạo hoạt động tương tự như nơ - ron trong não bộ con người, với mỗi nơ - ron là một hàm toán học có khả năng thu thập và phân loại dữ liệu. Cấu trúc này giống như các phương pháp thống kê dựa trên đồ thị đường cong hoặc phân tích hồi quy với các nút mạng được kết nối với nhau

1.3. Xử lý ngôn ngữ tự nhiên

Xử lý ngôn ngữ tự nhiên – NLP (Nature Language Processing) là một nhánh của Trí tuệ nhân tạo, tập trung vào việc nghiên cứu sự tương tác giữa máy tính và ngôn ngữ tự nhiên của con người.

Mục tiêu của NLP là giúp máy tính hiểu và thực hiện hiệu quả những nhiệm vụ liên quan đến ngôn ngữ của con người như: tương tác giữa người và máy, cải thiện hiệu quả giao tiếp giữa con người với con người; nâng cao hiệu quả xử lý văn bản và lời nói.

NLP ngày nay được ứng dụng ngày càng nhiều và rộng rãi trên nhiều lĩnh vực. Càng ngày càng có nhiều mô hình NLP mạnh mẽ ra đời, làm cải tiến và thay đổi đối với nhiều mặt trong cuộc sống. Một vài mô hình NLP nổi tiếng có thể kể đến như: ChatGPT, Poe, Gemini, Siri, Google Assistant…

1.3.1.Một số bài toán trong NLP

Xử lý ngôn ngữ tự nhiên được ứng dụng trong nhiều bài toán thực tế, cụ thể như sau:

- Mô hình hóa ngôn ngữ (Language modelling)

Mô hình hóa ngôn ngữ (LM) gán một xác suất cho bất kỳ chuỗi từ nào. Về cơ bản, trong bài toán này, ta cần dự đoán từ tiếp theo xuất hiện theo trình tự, dựa trên lịch sử của các từ đã xuất hiện trước đó. LM rất quan trọng trong các ứng dụng khác nhau của NLP, và là lý do tại sao máy móc có thể hiểu được thông tin định tính. Một số ứng dụng của Mô hình hóa ngôn ngữ bao gồm: nhận dạng giọng nói, nhận dạng ký tự quang học, nhận dạng chữ viết tay, dịch máy và sửa lỗi chính tả.

- Phân loại văn bản (Text classification)

Phân loại văn bản gán các danh mục được xác định trước cho văn bản dựa trên nội dung của nó. Cho đến nay, phân loại văn bản là ứng dụng phổ biến nhất của NLP, được sử dụng để xây dựng các công cụ khác nhau như trình phát hiện thư rác và chương trình phân tích cảm xúc.

- Trích xuất thông tin (Information extraction)

Trích xuất thông tin (IE) tự động trích xuất thông tin có liên quan từ các tài liệu văn bản không có cấu trúc và / hoặc bán cấu trúc. Ví dụ về các loại tài liệu này bao gồm lịch sự kiện từ email hoặc tên của những người được đề cập trong một bài đăng trên mạng xã hội.

- Truy xuất thông tin (Information retrieval)

Trích xuất thông tin là bài toán làm nhiệm vụ tìm kiếm các tài liệu có liên quan từ một bộ dữ liệu lớn các tài liệu liên quan đến truy vấn do người dùng thực hiện. Google là một loại hệ thống Truy xuất Thông tin (IR) phổ biến nhất mà chúng ta thường sử dụng.

- Tác tử phần mềm hội thoại (Conversational agent)

Tác tử phần mềm hội thoại thuộc AI hội thoại, liên quan đến việc xây dựng các hệ thống đối thoại mô phỏng các tương tác của con người. Các ví dụ phổ biến về AI hội thoại bao gồm Alexa, Siri, Google Home, Cortana, hay trợ lý ảo ViVi. Các công nghệ như chatbot cũng được hỗ trợ bởi tác tử phần mềm hội thoại và ngày càng phổ biến trong các doanh nghiệp.

- Tóm tắt văn bản (Text summarization)

Tóm tắt văn bản là quá trình rút ngắn một tập hợp dữ liệu để tạo một tập hợp con đại diện cho thông tin quan trọng nhất hoặc có liên quan trong nội dung gốc.

- Hỏi đáp (Question answering)

Hỏi đáp là bài toán xây dựng các hệ thống có thể tự động trả lời cho các câu hỏi do con người đặt ra bằng ngôn ngữ tự nhiên. Đây là bài toán có thể coi là tổng quan nhất trong NLP, nó có thể thực hiện được nhiều nhiệm vụ khác nhau trong NLP: tóm tắt văn bản, hỏi đáp, truy xuất nội dung, phân loại văn bản...

- Dịch máy (Machine translation)

Dịch máy (MT) là một nhánh con của ngôn ngữ học tính toán liên quan đến việc chuyển đổi một đoạn văn bản từ ngôn ngữ này sang ngôn ngữ khác. Một ứng dụng phổ biến của loại này là Google Dịch.

- Mô hình hóa chủ đề (Topic modelling)

Mô hình hóa chủ đề là một kỹ thuật Học máy không giám sát giúp khám phá cấu trúc chủ đề của một bộ tài liệu lớn. Ứng dụng NLP này là một công cụ khá phổ biến, được sử dụng trên nhiều lĩnh vực khác nhau – như Văn học, và Tin sinh học.

1.3.2. Một số mô hình trong NLP

Trong xử lý ngôn ngữ tự nhiên (NLP), có nhiều mô hình và kiến trúc khác nhau được sử dụng để giải quyết các tác vụ như dịch máy, phân loại văn bản, nhận diện thực thể, tóm tắt văn bản,...

1.3.2.1. Mô hình túi từ - Bag of word(BoW)

Mô hình túi từ biểu diễn văn bản theo cách đơn giản bằng cách đếm tần suất xuất hiện của các từ mà không quan tâm đến ngữ nghĩa hay vị trí của từ trong văn bản. BoW thường được sử dụng với các phương pháp truyền thống như Naive Bayes hoặc SVM. Mô hình này thường được ứng dụng trong những bài toán không yêu cầu phân tích ngữ pháp phức tạp như phân tích cảm xúc (Sentiment Analysis), phân loại văn bản (Text Classification),…

Ưu và nhược điểm của BoW

- Ưu điểm:

+ Đơn giản, dễ cài đặt.

+ Hiệu quả với những bài toán cơ bản (phân loại văn bản, lọc spam, phân tích cảm xúc...).

+ Là nền tảng để hiểu các mô hình phức tạp hơn (TF-IDF, word embeddings, BERT...).

- Nhược điểm:

+ Mất ngữ cảnh: không quan tâm thứ tự từ.

+ Vector rất thưa (sparse) khi từ vựng lớn.

+ Không phản ánh ngữ nghĩa: “dog” và “puppy” coi như 2 từ hoàn toàn khác nhau.

+ Dễ gây overfitting nếu dữ liệu huấn luyện nhỏ nhưng từ vựng lớn.

1.3.2.2. Mô hình TF - IDF

Mô hình TF-IDF (Term Frequency-Inverse Document Frequency) là một cải tiến của mô hình Bag of Words (BoW), giúp giải quyết một số hạn chế của BoW bằng cách không chỉ tính tần suất từ xuất hiện mà còn xem xét tầm quan trọng của từ trong một tập tài liệu lớn hơn. TF-IDF có nhiều ứng dụng trong các bài toán NLP, đặc biệt là khi bạn muốn tìm ra các từ có ý nghĩa quan trọng hơn trong một tài liệu so với các từ xuất hiện phổ biến nhưng ít mang giá trị (như ”và”, ”là”, ”ở”).

Term Frequecy (TF):

Tần suất của từ t trong tài liệu d:

$$TF(t,d) = \frac{f_{t,d}}{\sum_k f_{k,d}}$$

Trong đó $f_{t,d}$ là số lần từ t xuất hiện trong tài liệu d.

. Inverse Document Frequency (IDF)

Mức độ hiếm của từ t trong tập dữ liệu:

$$IDF(t) = \log \frac{N}{1+n_t}$$

Trong đó:

- N: tổng số tài liệu.
- n_t: số tài liệu có chứa từ t.

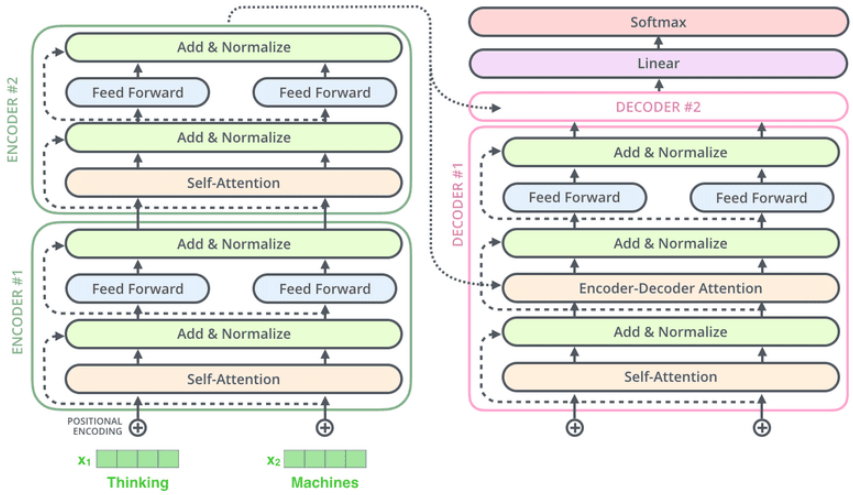
TF–IDF:

Trọng số TF–IDF của từ t trong tài liệu d:

$$TF\text{-}IDF(t,d) = TF(t,d) \times IDF(t)$$

1.3.2.3. Mô hình Transformer

Transformer là mô hình mạng nơ-ron đã tạo ra cuộc cách mạng trong NLP, đặc biệt với các thành phần như self-attention cho phép mô hình học ngữ cảnh từ toàn bộ chuỗi văn bản. Mô hình này đã thay thế RNN/LSTM trong nhiều tác vụ nhờ khả năng xử lý song song và học hiệu quả trên dữ liệu lớn.



Hình 1.4. Cơ chế hoạt động của Transformer

Đặc điểm chính của Transformer:

Transformer sử dụng cơ chế Attention (Tập trung), thay vì tuần tự xử lý từng bước như RNN, để học mối quan hệ giữa các phần tử trong chuỗi dữ liệu (ví dụ: từ trong câu) một cách đồng thời. Điều này giúp nó có khả năng xử lý các chuỗi dữ liệu dài mà không bị hạn chế bởi các vấn đề về ghi nhớ dài hạn như LSTM.

Kiến trúc chính của Transformer bao gồm hai phần:

+ Encoder (Bộ mã hóa): Nhận chuỗi đầu vào và chuyển đổi thành một biểu diễn trung gian.

+ Decoder (Bộ giải mã): Dựa vào biểu diễn trung gian từ encoder và sinh ra chuỗi đầu ra.

Mỗi phần của mô hình gồm nhiều lớp, và mỗi lớp có hai thành phần chính:

+ Multi-head Self-Attention (Chú ý đa đầu tự động): Cho phép mô hình chú ý đến các phần khác nhau trong chuỗi đầu vào cùng một lúc.

+ Feedforward Network (Mạng nơ-ron lan truyền tiến): Mạng nơ-ron truyền thống để xử lý đầu ra từ self-attention.

Cơ chế Attention

Self-Attention là thành phần quan trọng nhất của Transformer. Nó cho phép mô hình xem xét mối quan hệ giữa mọi phần tử trong chuỗi với nhau, không chỉ giữa các từ liền kề mà giữa bất kỳ từ nào trong câu.

Ví dụ, trong một câu như ”The cat sat on the mat,” Self-Attention giúp mô hình hiểu rằng từ ”cat” có liên quan đến từ ”sat” hơn là từ ”mat.”

Self-Attention hoạt động bằng cách tính trọng số giữa các từ trong câu, cho biết từ nào quan trọng hơn với từ hiện tại. Quy trình này bao gồm việc tính toán ba vector chính từ mỗi từ trong chuỗi:

- Query (Truy vấn)

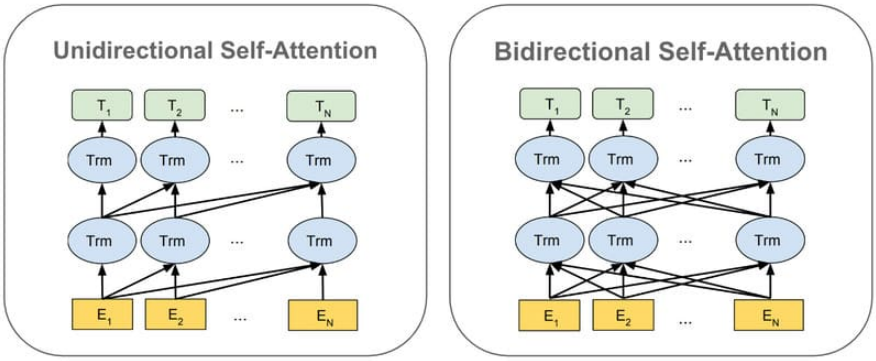
- Key (Khóa)

- Value (Giá trị)

Tổng trọng số của các từ trong chuỗi giúp mô hình quyết định phần nào của câu cần chú ý hơn.

1.3.2.4. Mô hình BERT

BERT là một biến thể của Transformer, tập trung chủ yếu vào phần Encoder của kiến trúc gốc. Khác với các mô hình dựa trên Transformer thông thường chỉ đọc văn bản theo một chiều (từ trái sang phải như GPT hoặc từ phải sang trái), BERT sử dụng cơ chế bidirectional (hai chiều), nghĩa là nó xem xét ngữ cảnh của từ cả từ phía trước và phía sau trong cùng một lúc.



Hình 1.5. Cơ chế self -attention trong kiến trúc của mô hình BERT

Đặc điểm nổi bật của BERT:

Training hai chiều (Bidirectional Training): BERT khác biệt so với các mô hình trước đó vì nó không chỉ đọc văn bản theo một chiều. Nhờ việc nhìn cả trước và sau một từ, BERT hiểu ngữ cảnh tổng thể của câu và từ, giúp mô hình biểu diễn ý nghĩa từ chính xác hơn.

Masked Language Model (MLM):

+ Trong quá trình huấn luyện, BERT sử dụng một phương pháp gọi là Masked Language Model. Cụ thể, một số từ trong câu sẽ bị ”mask” (che giấu), và nhiệm vụ của mô hình là dự đoán những từ bị che giấu đó.

+ Điều này giúp mô hình học được ngữ cảnh của từ dựa trên các từ xung quanh, chứ không chỉ dựa vào ngữ cảnh của từ trước nó.

Next Sentence Prediction (NSP):

BERT cũng được huấn luyện với một nhiệm vụ khác là Next Sentence Prediction. Mô hình sẽ được cho hai câu và phải dự đoán xem liệu câu thứ hai có phải là câu tiếp theo của câu thứ nhất hay không. Điều này giúp mô hình hiểu được mối quan hệ giữa các câu trong văn bản.

Ngày nay ngày càng nhiều các mô hình NLP hiện đại, được phát triển và cải tiến từ các mô hình cũ được ra đời với hiệu suất ngày càng cao như GPT, T5 ra đời, giúp nâng cao sức mạnh của NLP trong thời đại hiện nay

CHƯƠNG 2: TỔNG QUAN MÔ HÌNH

2.1. Support vector machine

SVM (Support vector machine) là một thuật toán học có giám sát trong học máy, được sử dụng để giải quyết các bài toán phân loại và hồi quy. SVM tìm ra đường phân chia tốt nhất giữa các điểm dữ liệu và sử dụng nó để dự đoán nhãn của các dữ liệu mới.

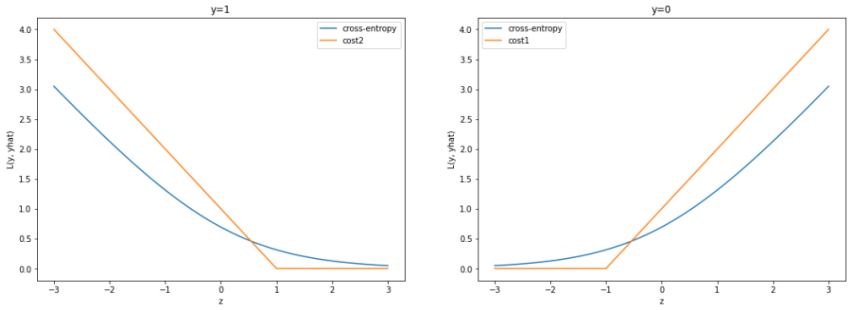
2.1.1. Hàm mất mát trong SVM

Trong SVM chúng ta có một thay đổi đột phá đó là tìm cách xấp xỉ hàm mất mát dạng cross-entropy của Logistic bằng một hàm mà chỉ phạt những điểm ở gần đường biên thay vì phạt những điểm ở xa đường biên bằng cách đưa mức phạt về 0.

Cụ thể đó là hai hàm phạt $\text{cost}_1()$ và $\text{cost}_2()$ tương ứng với $y = 0$ và $y = 1$ như bên dưới:

$$\begin{cases} \text{cost}_1(z) = \max(1 + z,\, 0) & \text{nếu } y = 0, \\ \text{cost}_2(z) = \max(0,\, 1 - z) & \text{nếu } y = 1. \end{cases}$$

Hai hàm này thể hiện chi phí phải bỏ ra nếu phân loại sai các nhãn lần lượt thuộc 0 hoặc 1. Dạng tổng quát của chúng là $\max(0, t)$, còn được gọi là *hinge loss*. Đây là một trong những hàm mất mát thường gặp trong *machine learning*.



Hình 2.1. Hình minh họa hàm mất mát cost1, cost 2

Ta nhận thấy hình dạng của các hàm mất mát cost_1 và cost_2 cũng gần tương tự như *cross-entropy*. Điểm khác biệt chính đó là giá trị của mất mát bằng 0 nếu $z \geq 1$ (đối với nhãn $y = 1$) hoặc $z \leq -1$ (đối với nhãn $y = 0$). Theo các hàm mất mát mới này, chúng ta bỏ qua việc phạt phân loại sai những điểm nằm xa đường biên. Đối với những điểm nằm gần đường biên nhất thì mới ảnh hưởng tới hàm mất mát. Tập hợp những điểm nằm gần đường biên sẽ giúp xác định đường biên và được gọi là *tập hỗ trợ* (support vector).

Như vậy sau khi thay đổi hàm phạt ta thu được hàm mất mát mới dạng:

$$\mathcal{L}(\mathbf{w}) = \sum_{i=1}^{n} -\Big[y_i \ \text{cost}_1(\hat{y}_i) + (1 - y_i) \ \text{cost}_2\big(1 - \hat{y}_i\big) \Big].$$

SVM cho phép ta giảm thiểu *quá khớp* thông qua một thành phần điều chuẩn cũng tương tự như hồi quy Logistic:

$$\mathcal{L}(\mathbf{w}) = C\left(\sum_{i=1}^{n} -\Big[\, y_i \ \text{cost}_1(\hat{y}_i) + (1 - y_i) \ \text{cost}_2\big(1 - \hat{y}_i\big)\Big]\right) + \underbrace{\frac{\lambda}{2}\,\|\mathbf{w}\|_2^2}_{\text{regularization term}} \quad .$$

Trong công thức trên thì hằng số $C > 0$ thể hiện ảnh hưởng của sai số phân loại lên hàm mất mát. Trong khi $\lambda > 0$ là hằng số của thành phần điều chuẩn (*regularization term*) thể hiện tác động của độ lớn trọng số hồi quy **w** lên hàm mất mát.

Khi tăng tỷ lệ λ/C có thể giúp các trọng số của mô hình được kiểm soát về độ lớn, thông qua đó làm cho độ phức tạp của đường biên phân chia giảm và kiểm soát hiện tượng quá khớp.

2.1.2. Điều kiện KKT

Bài toán tối ưu có hàm mục tiêu và hệ điều kiện ràng buộc còn được gọi là *bài toán gốc* (primal problem). Để giải trực tiếp bài toán gốc là tương đối khó nên chúng ta sẽ chuyển sang giải bài toán tối ưu trên *hàm đối ngẫu Lagrange* (Lagrange Dual Function).

$$g(\boldsymbol{\lambda}, \boldsymbol{\nu}) \ = \ \min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\nu}) \ = \ \tfrac{1}{2}\,\mathbf{x}^\top A\mathbf{x} + \mathbf{b}^\top \mathbf{x} + r + \boldsymbol{\lambda}^\top H\mathbf{x} + \boldsymbol{\nu}^\top G\mathbf{x}.$$

Trong đó $\boldsymbol{\lambda}, \boldsymbol{\nu}$ là những véc tơ hệ số có kích thước lần lượt bằng với số lượng các điều kiện ràng buộc phương trình và bất phương trình và có giá trị lớn hơn hoặc bằng 0. Trong trường hợp bài toán gốc không tồn tại hệ điều kiện bất phương trình thì hàm đối ngẫu Lagrange có dạng:

$$g(\boldsymbol{\lambda}) \ = \ \min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) \ = \ \tfrac{1}{2}\,\mathbf{x}^\top A\mathbf{x} + \mathbf{b}^\top \mathbf{x} + r + \boldsymbol{\lambda}^\top H\mathbf{x}.$$

Ta dễ dàng nhận thấy ưu điểm của hàm đối ngẫu $g(\boldsymbol{\lambda}, \boldsymbol{\nu})$ so với hàm mục tiêu gốc đó là:

- Là một hàm lồi bất kể hàm mục tiêu gốc có là hàm lồi hay không. Nếu tinh tế ta có thể nhận ra hàm đối ngẫu chính là cận dưới (infimum) của hàm mục tiêu gốc.

- Trong điều kiện tồn tại $\mathbf{x}^*$ sao cho $H\mathbf{x}^* = \mathbf{d}$ và $G\mathbf{x}^* \prec \mathbf{e}$ thì chúng ta nói tiêu chuẩn Slater được thỏa mãn. Bài toán đối ngẫu khi thỏa mãn tiêu chuẩn Slater sẽ là một bài toán đối ngẫu mạnh (strong duality). Khi đó hệ điều kiện KKT là điều kiện cần và cũng là điều kiện đủ và giá trị cực tiểu $f^* = g^*$.

Bài toán đối ngẫu có thể được giải thông qua hệ điều kiện KKT. Đối với bài toán tối ưu QP không tồn tại hệ điều kiện ràng buộc bất phương trình thì có hệ điều kiện KKT như bên dưới:

$$\begin{bmatrix} A & H^\top \\ H & 0 \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} -\mathbf{b} \\ \mathbf{d} \end{bmatrix}.$$

2.1.3. Bài toán đối ngẫu SVM

Hàm đối ngẫu trong bài toán tối ưu.

$$g(\boldsymbol{\lambda}) \;=\; \min_{\mathbf{w},\, b} \mathcal{L}(\mathbf{w}, b, \boldsymbol{\lambda}) = \tfrac{1}{2}\|\mathbf{w}\|_2^2 + \sum_{i=1}^{N} \lambda_i \big(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\big)\,.$$

Nghiệm tối ưu của hàm Lagrange có thể được tìm thông qua đạo hàm bậc nhất:

$$\begin{aligned}
\frac{\partial \mathcal{L}(\mathbf{w}, b, \boldsymbol{\lambda})}{\partial \mathbf{w}} &= \mathbf{w} - \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i = \mathbf{0} \;\Rightarrow\; \mathbf{w} = \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i, \\
\frac{\partial \mathcal{L}(\mathbf{w}, b, \boldsymbol{\lambda})}{\partial b} &= -\sum_{i=1}^{N} \lambda_i y_i = 0, \\
\frac{\partial \mathcal{L}(\mathbf{w}, b, \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} &= \big[\, 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b) \,\big]_{i=1}^{N} = \mathbf{0}.
\end{aligned}$$

Bằng một số phép biến đổi đơn giản trên hàm đối ngẫu ta thu được một biểu thức ngắn gọn:

$$\begin{aligned}
g(\boldsymbol{\lambda}) &= \tfrac{1}{2}\|\mathbf{w}\|_2^2 + \sum_{i=1}^{N} \lambda_i - \mathbf{w}^\top \sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i - b \sum_{i=1}^{N} \lambda_i y_i \\
&= \sum_{i=1}^{N} \lambda_i - \tfrac{1}{2}\|\mathbf{w}\|_2^2 \\
&= \sum_{i=1}^{N} \lambda_i - \tfrac{1}{2} \Big(\sum_{i=1}^{N} \lambda_i y_i \mathbf{x}_i \Big)^\top \Big(\sum_{j=1}^{N} \lambda_j y_j \mathbf{x}_j \Big) \\
&= \sum_{i=1}^{N} \lambda_i - \tfrac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \lambda_i \lambda_j y_i y_j \, \mathbf{x}_i^\top \mathbf{x}_j.
\end{aligned}$$

Theo hệ điều kiện *KKT* thì giá trị cực tiểu của hàm $g(\boldsymbol{\lambda})$ đạt được khi

$$\sum_{i=1}^{N} \lambda_i \big(1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b)\big) = 0.$$

Đẳng thức trên đạt được khi $\lambda_i = 0$ hoặc $1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b) = 0,\ \forall i = 1, \ldots, N$. Trên thực tế thì vectơ $\boldsymbol{\lambda}$ là một vectơ thưa có hầu hết các chiều đều bằng 0. Đối với những điểm dữ liệu tương ứng với $\lambda_i > 0$ thì phương trình $1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b) = 0$ sẽ được thỏa mãn và tập hợp những điểm này khi đó sẽ nằm trên *mép của lề*. Tập hợp những điểm này còn gọi là *tập hỗ trợ* (support vector) và được kí hiệu là S.

2.1.4. Dự báo nhãn cho bài toán tối ưu

Nhãn của một quan sát trong mô hình *SVM* sẽ phụ thuộc vào dấu của đường biên:

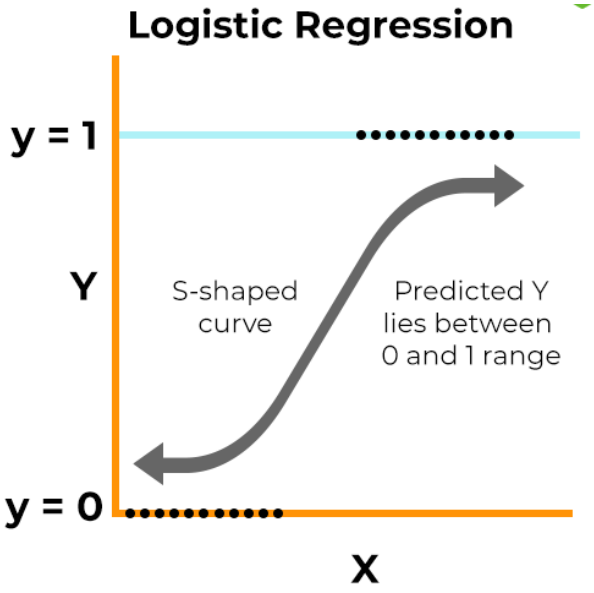
$$h_{\mathbf{w},b}(\mathbf{x}_i) = b + \mathbf{w}^\top \mathbf{x}_i = b + \left(\sum_{(\mathbf{x}_j, y_j) \in S} \lambda_j y_j \mathbf{x}_j^\top \right) \mathbf{x}_i = b + \sum_{(\mathbf{x}_j, y_j) \in S} \lambda_j y_j \, \mathbf{x}_j^\top \mathbf{x}_i.$$

Trong trường hợp $h_{\mathbf{w},b}(\mathbf{x}_i) > 0$ thì điểm được dự báo nhãn 1, và ngược lại là nhãn -1.

Từ dòng thứ nhất sang dòng thứ hai là vì **w** được tính trực tiếp từ *tập hỗ trợ*. Công thức trên cho thấy thay vì phải xác định nhãn dựa trên các hệ số của phương trình đường biên **w**, ta có thể suy ra thông qua các điểm thuộc *tập hỗ trợ* S.

2.2. Logistic Regression

Hồi quy Logistic là một mô hình học máy tuyến tính được sử dụng chủ yếu cho các bài toán phân loại. Khác với hồi quy tuyến tính dự đoán giá trị liên tục, hồi quy Logistic ước lượng xác suất một đối tượng thuộc về một lớp nhất định. Mô hình này sử dụng hàm logistic (hay còn gọi là hàm sigmoid) để ánh xạ giá trị đầu ra về khoảng (0,1), từ đó có thể diễn giải như một xác suất. Bằng cách đặt ngưỡng (thường là 0.5), mô hình có thể quyết định nhãn phân loại cho dữ liệu đầu vào. Nhờ sự đơn giản, dễ cài đặt và hiệu quả trong các trường hợp dữ liệu tuyến tính phân tách, hồi quy Logistic thường được xem là mô hình cơ bản (baseline) trong nhiều bài toán học máy, đặc biệt là trong xử lý ngôn ngữ tự nhiên như phân loại văn bản hay phân tích cảm xúc.



Hình 2.2. Các giả định chính để triển khai Logistic Regression

2.2.1. Hàm mất mát trong hồi quy tuyến tính

Mục tiêu của tất cả các mô hình học có giám sát (supervised learning) trong machine learning là tìm ra một hàm số dự báo mà giá trị của chúng sai khác so với ground truth là nhỏ nhất. Ground truth ở đây chính là giá trị của biến mục tiêu y. Giá trị này được đo lường thông qua các hàm mất mát (loss function). Huấn luyện mô hình machine learning thực chất là quy về tìm cực trị của hàm mất mát. Tuỳ thuộc vào bài toán mà chúng ta có những dạng hàm mất mát khác nhau.

Trong bài toán dự báo chúng ta sẽ sử dụng hàm MSE (Mean Square Error) làm hàm mất mát. Hàm số này có giá trị bằng trung bình của tổng bình phương sai số giữa giá trị dự báo và ground truth. Giả sử chúng ta xét phương trình hồi quy đơn biến gồm n quan sát có biến phụ thuộc là $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}$ và biến đầu vào $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$.

Véc tơ tham số $\mathbf{w} = (w_0, w_1)$ trong đó w_0, w_1 lần lượt là hệ số gốc và hệ số ước lượng. Khi đó phương trình hồi quy tuyến tính đơn biến có dạng:

$$\hat{y}_i = f(x_i) = w_0 + w_1 \cdot x_i$$

Trong đó (x_i, y_i) là điểm dữ liệu thứ i.

Mục tiêu của chúng ta là đi tìm véc tơ **w** sao cho sai số giữa giá trị dự báo và thực tế là nhỏ nhất. Tức là tối thiểu hoá hàm mất mát MSE, được định nghĩa như sau:

$$\mathcal{L}(\mathbf{w}) = \frac{1}{2n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = \frac{1}{2n}\sum_{i=1}^{n}\left(y_i - w_0 - w_1 \cdot x_i\right)^2$$

Ký hiệu $\mathcal{L}(\mathbf{w})$ thể hiện rằng hàm mất mát là một hàm theo tham số **w** trong điều kiện ta đã biết đầu vào là véc tơ **x** và véc tơ biến phụ thuộc **y**. Ta có thể tìm cực trị của phương trình trên dựa vào đạo hàm theo w_0 và w_1 như sau:

- Đạo hàm theo w_0:

$$\frac{\delta\mathcal{L}(\mathbf{w})}{\delta w_0} = -\frac{1}{n}\sum_{i=1}^{n}(y_i - w_0 - w_1 x_i)$$

$$= -\frac{1}{n}\sum_{i=1}^{n} y_i + w_0 + w_1\frac{1}{n}\sum_{i=1}^{n} x_i$$

$$= -\bar{y} + w_0 + w_1\bar{x}$$

$$= 0 \tag{1}$$

- Đạo hàm theo w_1:

$$\frac{\delta \mathcal{L}(\mathbf{w})}{\delta w_1} = -\frac{1}{n}\sum_{i=1}^{n} x_i \left(y_i - w_0 - w_1 x_i\right)$$

$$= -\frac{1}{n}\sum_{i=1}^{n} x_i y_i + w_0 \frac{1}{n}\sum_{i=1}^{n} x_i + w_1 \frac{1}{n}\sum_{i=1}^{n} x_i^2$$

$$= -\overline{xy} + w_0 \bar{x} + w_1 \overline{x^2}$$

$$= 0 \tag{2}$$

Từ phương trình (1) ta suy ra:

$$w_0 = \bar{y} - w_1 \bar{x}.$$

Thế vào phương trình (2) ta tính được:

$$-\overline{xy} + w_0 \bar{x} + w_1 \overline{x^2} = -\overline{xy} + (\bar{y} - w_1 \bar{x})\bar{x} + w_1 \overline{x^2}$$

$$= -\overline{xy} + \bar{y}\bar{x} - w_1 \bar{x}^2 + w_1 \overline{x^2}$$

$$= 0$$

Từ đó suy ra:

$$w_1 = \frac{\overline{xy} - \bar{x}\bar{y}}{\overline{x^2} - \bar{x}^2}$$

Sau khi tính được w_1 thế vào ta tính được:

$$w_0 = \bar{y} - w_1 \bar{x}$$

Đạo hàm bậc nhất bằng 0 mới chỉ là điều kiện cần để **w** là cực trị của hàm mất mát. Để khẳng định cực trị đó là cực tiểu thì chúng ta cần chứng minh thêm đạo hàm bậc hai lớn hơn hoặc bằng 0 hay hàm số đó là hàm lồi.

2.3. Naive Bayes

2.3.1. Bộ phân loại naive Bayes

Naive Bayes Classification (NBC) là một thuật toán dựa trên định lý Bayes về lý thuyết xác suất để đưa ra các phán đoán cũng như phân loại dữ liệu dựa trên các dữ liệu được quan sát và thống kê. Naive Bayes Classification là một trong những thuật toán được ứng dụng rất nhiều trong các lĩnh vực Machine learning dùng để đưa các dự đoán chính xác nhất dự trên một tập dữ liệu đã được thu thập, vì nó khá dễ hiểu và độ chính xác cao. Nó thuộc vào nhóm Supervised Machine Learning Algorithms (thuật toán học có hướng dẫn), tức là máy học từ các ví dụ từ các mẫu dữ liệu đã có. Nhìn chung, khó có cách tính trực tiếp p(c|x). Thay vào đó, quy tắc Bayes thường được sử dụng:

$$p(x_i|c) = p(x_i|\mu_{ci}, \sigma_{ci}^2) = \frac{1}{\sqrt{2\pi\sigma_{ci}^2}} \exp\left(-\frac{(x_i - \mu_{ci})^2}{2\sigma_{ci}^2}\right)$$

Hình 1: Công thức tính c

Dấu bằng thứ hai xảy ra theo quy tắc Bayes, dấu bằng thứ ba xảy ra vì p(x) ở mẫu số không phụ thuộc vào 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.

2.3.2. Các phân phối thường dùng trong NBC

2.3.2.1 Gaussian naive Bayes

Mô hình này được sử dụng chủ yếu trong loại dữ liệu mà các thành phần là các biến liên tục. Với mỗi chiều dữ liệu i và một nhãn c, xi tuân theo một phân phối chuẩn có kỳ vọng μci và phương sai σc2i:

$$p(x_i|c) = p(x_i|\mu_{ci}, \sigma_{ci}^2) = \frac{1}{\sqrt{2\pi\sigma_{ci}^2}} \exp\left(-\frac{(x_i - \mu_{ci})^2}{2\sigma_{ci}^2}\right)$$

Hình 2: Gaussian naive Bayes

Trong đó, bộ tham số θ = μci, σc2i được xác định bằng MLE dựa trên các điểm trong tập huấn luyện thuộc nhãn c.

2.3.2.2 Multinomial naive 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, p(xi|c) 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 : λci = p(xi|c) = Nci . (5.7) Nc Trong đó: -Nci 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. -N 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 Nc =∑di=1 Nci, từ đó ∑di=1 λci = 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 sẽ bằng không, dẫn đến vế phải 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: λ^ci = Nci + α (5.8) Nc + dα với α là một số dương, thường bằng 1, để tránh trườPng hợp tử số bằng không. Mẫu số được cộng với dα để đảm bảo tổng xác suất di=1 λ^ci = 1. Như vậy, mỗi nhãn c được mô tả bởi một bộ các số dương có tổng bằng 1: λ^c = λ^c1,...,λ^cd.

2.3.2.3 Bernoulli Naive Bayes

Mô hình này được áp dụng cho các loại dữ liệu mà mỗi thành phần là một giá trị nhị phân – bằng 0 hoặc 1. Ví dụ, cũng với loại văn bản nhưng thay vì đếm tổng số lần xuất hiện của một từ trong văn bản, ta chỉ cần quan tâm từ đó có xuất hiện hay không. Khi đó, p(xi|c) được tính bởi: p(xi|c) = p(i|c)xi (1 − p(i|c))1−xi với p(i|c) được hiểu là xác suất từ thứ i xuất hiện trong các văn bản của class c, xi bằng 1 hoặc 0 tuỳ vào việc từ thứ i có xuất hiện hay không.