1. **MACHINE LEARNING**
2. Lịch sử hình thành của Machine Learning

Lịch sử hình thành của máy học (machine learning) là một hành trình kéo dài từ những năm đầu của thế kỷ 20 đến hiện tại, với nhiều bước tiến và tiến bộ quan trọng. Dưới đây là một cái nhìn tổng quan về lịch sử của máy học:

* Thập niên 1950 - 1960:

Trong thập niên 1950 đến 1960, các ý tưởng đầu tiên về máy học bắt đầu hình thành, đặc biệt là trong lĩnh vực trí tuệ nhân tạo (AI). Alan Turing, một nhà toán học và nhà máy tính nổi tiếng, đã đề xuất "máy Turing", một khái niệm cho máy tính có khả năng học. Ông đặt ra câu hỏi liệu một máy tính có thể giả lập được trí tuệ của con người hay không, thông qua trò chơi Imitation Game, sau này được biết đến như là test Turing.

Trong cùng thời kỳ, Marvin Minsky và các nhà khoa học khác đã đề xuất các mô hình mới gọi là "mạng nơ-ron nhân tạo". Các mạng nơ-ron này được lấy cảm hứng từ cách mà não người hoạt động, trong đó các "nơ-ron" nhân tạo được kết nối với nhau để xử lý thông tin. Mặc dù các mô hình ban đầu rất đơn giản và hạn chế, nhưng chúng đã tạo ra nền tảng cho sự phát triển sau này của machine learning.

Những ý tưởng và nghiên cứu trong thập niên này đã đặt nền móng cho sự tiến bộ vượt bậc trong lĩnh vực máy học trong các thập kỷ tiếp theo. Đồng thời, cũng đã mở ra cánh cửa cho những khám phá và ứng dụng mới trong trí tuệ nhân tạo và các lĩnh vực liên quan.

* Thập niên 1970 - 1980:

Sau những bước đầu tiên trong thập kỷ 1950 và 1960, sự phát triển của học máy và các thuật toán liên quan tiếp tục diễn ra mạnh mẽ. Các nhà nghiên cứu và nhà khoa học đã tiếp tục khám phá và phát triển các phương pháp mới như học tăng cường (reinforcement learning) và cây quyết định (decision trees).

Học tăng cường là một phương pháp trong machine learning mà một hệ thống tự động học qua thời gian từ những tương tác với môi trường bằng cách thử và sai. Trong khi đó, cây quyết định là một phương pháp học có giám sát, trong đó một loạt các quyết định dựa trên các đặc trưng của dữ liệu được sử dụng để dự đoán hoặc phân loại. Cả hai phương pháp này đều đã được áp dụng rộng rãi trong các lĩnh vực như xử lý ngôn ngữ tự nhiên và thị giác máy tính.

Trong xử lý ngôn ngữ tự nhiên, học máy được sử dụng để xây dựng các hệ thống nhận diện và tự động xử lý văn bản, từ việc phân loại email vào hòm thư rác đến dịch tự động và phân tích cảm xúc trong văn bản. Các thuật toán như học tăng cường có thể được sử dụng để cải thiện hiệu suất của các hệ thống dịch máy thông qua việc tối ưu hóa các chiến lược dịch.

Những tiến bộ trong học máy và các thuật toán liên quan không chỉ mở ra những cơ hội mới trong nhiều lĩnh vực ứng dụng, mà còn đóng góp vào sự tiến bộ chung của trí tuệ nhân tạo và công nghệ máy tính.

* Thập niên 1980 - 1990:

Sự phát triển của các thuật toán học máy trở nên phổ biến hơn, đặc biệt là trong lĩnh vực học tăng cường và học sâu (deep learning). Một số công nghệ và thuật toán như mạng nơ-ron nhân tạo lan truyền ngược (backpropagation) và máy vector hỗ trợ (support vector machines) được phát triển.

* Thập niên 1990 - 2000:

Các nhà nghiên cứu tập trung vào việc tối ưu hóa và cải tiến các thuật toán học máy hiện có, cũng như sự phát triển của các phương pháp mới như học tập tập trung (ensemble learning).

* Thập niên 2000 - 2010:

Sự phát triển nhanh chóng của internet và dữ liệu kỹ thuật số đã tạo ra điều kiện thuận lợi cho sự bùng nổ của học máy. Với sự gia tăng đáng kể về lượng dữ liệu được tạo ra và thu thập từ các nguồn như mạng xã hội, thiết bị di động, cảm biến và các hệ thống thông minh, khối lượng lớn dữ liệu được gọi là Big Data đã trở thành một nguồn tài nguyên quý giá cho học máy.

Big Data không chỉ đem lại lượng dữ liệu lớn, mà còn đặc biệt về độ phức tạp và đa dạng. Điều này thúc đẩy sự phát triển của các phương pháp xử lý dữ liệu lớn như Hadoop, Spark và các công nghệ lưu trữ dữ liệu phân tán khác. Những công nghệ này cung cấp khả năng xử lý và phân tích dữ liệu với tốc độ và quy mô mà trước đây là không thể.

Sự kết hợp giữa Big Data và các phương pháp học máy đã tạo ra nền tảng cho việc áp dụng máy học vào nhiều lĩnh vực khác nhau. Trong tài chính, học máy được sử dụng để phân tích dữ liệu tài chính và dự đoán xu hướng thị trường. Trong y tế, nó có thể được sử dụng để dự đoán và chuẩn đoán bệnh, cải thiện chăm sóc sức khỏe và dự đoán kết quả điều trị. Trong marketing, học máy giúp tối ưu hóa chiến lược quảng cáo, phân tích hành vi người tiêu dùng và tăng cường trải nghiệm khách hàng.

Với sức mạnh của Big Data và học máy, các tổ chức và doanh nghiệp có thể nắm bắt và phân tích thông tin từ dữ liệu để đưa ra các quyết định chiến lược thông minh và tạo ra giá trị cho doanh nghiệp. Sự kết hợp này không chỉ mang lại lợi ích kinh tế mà còn góp phần vào sự phát triển toàn diện của xã hội và kinh tế số.

* Thập niên 2010 - Hiện tại:

Sự phát triển của deep learning đánh dấu một bước tiến lớn trong lĩnh vực máy học. Công nghệ này đã mang lại những tiến bộ đáng kể trong việc xử lý dữ liệu phức tạp và tạo ra các mô hình dự đoán chính xác hơn. Deep learning, dựa trên các mạng nơ-ron sâu, đã trở thành một trong những công nghệ quan trọng nhất trong lĩnh vực máy học và trí tuệ nhân tạo.

Công nghệ deep learning cho phép mạng nơ-ron có khả năng học từ dữ liệu không có cấu trúc hoặc dữ liệu phức tạp mà không cần định nghĩa rõ ràng các đặc trưng. Điều này làm cho các mô hình học máy dựa trên deep learning trở nên mạnh mẽ hơn và có khả năng xử lý các nhiệm vụ phức tạp như nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên, và dự đoán.

Các công ty công nghệ hàng đầu như Google, Facebook, và Microsoft đã nhận ra tiềm năng của deep learning và tiếp tục đầu tư mạnh mẽ vào nghiên cứu và phát triển các ứng dụng của học máy và trí tuệ nhân tạo. Các ứng dụng của deep learning không chỉ giúp cải thiện trải nghiệm người dùng trên các nền tảng trực tuyến mà còn có ứng dụng rộng rãi trong nhiều lĩnh vực như y tế, tài chính, ô tô tự hành, và nhiều lĩnh vực khác.

Với sự phát triển nhanh chóng của deep learning, dự đoán và ứng dụng của học máy và trí tuệ nhân tạo sẽ tiếp tục mở ra những cơ hội mới và tạo ra những tiến bộ đáng kể trong tương lai. Đây là một phần quan trọng của cuộc cách mạng số và sẽ tiếp tục đóng góp vào sự phát triển toàn diện của xã hội và kinh tế.

1. Định nghĩa về Machine Learning

Machine Learning (Học máy) là thuật ngữ chỉ các hành động dạy máy tính để cải thiện một nhiệm vụ mà nó đang thực hiện. Cụ thể khả năng của Machine Learning là sử dụng các thuật toán để phân tích những thông tin sẵn có, sau đó học hỏi và đưa ra những quyết định, dự đoán về những thứ có liên quan. Thay vì phải lập trình ra một phần mềm với các hành động, hướng dẫn chi tiết để thực hiện một nhiệm vụ thì máy tính sẽ được học cách thực hiện các nhiệm vụ đó thông qua một lượng dữ liệu và các thuật toán.

Machine learning đang trở thành một yếu tố quan trọng trong lĩnh vực khoa học dữ liệu đang phát triển. Bằng cách sử dụng các phương pháp thống kê và đào tạo các thuật toán, machine learning có khả năng cung cấp các phân loại hoặc dự đoán và khám phá thông tin chi tiết từ những dự án khai thác dữ liệu.

Sử dụng thông tin chuyên sâu để thúc đẩy các quyết định về ứng dụng và kinh doanh cũng như tác động đến các chỉ số tăng trưởng. Khi dữ liệu lớn mở rộng và phát triển, nhu cầu về các nhà khoa học dữ liệu cũng tăng theo. Họ được yêu cầu giúp xác định các câu hỏi kinh doanh chính và thông tin cần thiết để trả lời chúng.

Thường thì các bài toán trong machine learning được phân thành hai loại chính: dự đoán (prediction) và phân loại (classification). Các bài toán dự đoán thường liên quan đến việc ước lượng giá trị, như giá nhà, giá xe, trong khi các bài toán phân loại thường liên quan đến việc xác định danh tính hoặc loại của một đối tượng, như nhận diện chữ viết tay, đồ vật, và các nhiệm vụ tương tự.

1. Các phương pháp Machine Learning

* **Supervised machine learning (Học Có Giám Sát)**

Học máy có giám sát, hay còn gọi là máy học có giám sát, là phương pháp sử dụng các tập dữ liệu được gắn nhãn để huấn luyện các thuật toán phân loại dữ liệu hoặc dự đoán kết quả một cách chính xác. Mô hình được điều chỉnh trọng lượng của nó khi dữ liệu đầu vào được đưa vào, trong quá trình xác nhận chéo để tránh tình trạng quá mức hoặc thiếu thông tin.

Học máy có giám sát hỗ trợ tổ chức giải quyết nhiều vấn đề thực tế quy mô lớn, như phân loại thư rác trong hộp thư đến cá nhân.

Các phương pháp thường được sử dụng trong Học máy có giám sát bao gồm: logistic regression, neural networks, linear regression, naive bayes, random forest và support vector machine (SVM).

* **Unsupervised machine learning (Học Không Giám Sát)**

Học máy không giám sát là phương pháp sử dụng các thuật toán máy học để phân tích và phân cụm các tập dữ liệu không được gắn nhãn. Không yêu cầu sự can thiệp của con người, các thuật toán này có khả năng phát hiện mẫu hoặc nhóm dữ liệu ẩn, làm cho chúng lý tưởng cho việc phân tích dữ liệu khám phá và nhận diện hình ảnh và mẫu.

Học máy không giám sát cũng được sử dụng để giảm số lượng tính năng trong một mô hình thông qua quá trình giảm kích thước, với phương pháp như phân tích thành phần chính và phân tích giá trị đơn lẻ.

Các thuật toán thường được sử dụng trong Học máy không giám sát bao gồm: k-means clustering, neural networks và các phương pháp phân cụm xác suất.

* **Semi-supervised learning (Học Tăng Cường)**

Học máy bán giám sát cung cấp một phương pháp hiệu quả giữa học máy có giám sát và không giám sát. Trong quá trình đào tạo, nó sử dụng một tập dữ liệu có nhãn nhỏ hơn để hướng dẫn phân loại và trích xuất tính năng từ một tập dữ liệu lớn hơn, không được gắn nhãn. Phương pháp này có thể giải quyết vấn đề thiếu dữ liệu được gắn nhãn cho thuật toán học có giám sát và làm giảm chi phí đánh nhãn đối với dữ liệu.

* Deep Learning (Học Sâu)

Học sâu là một phần của học máy, đặc biệt là mạng nơ-ron sâu có ba lớp trở lên. Những mạng nơ-ron này cố gắng mô phỏng hành vi của não người, giúp deep learning "học" từ lượng lớn dữ liệu. Công nghệ này đã thúc đẩy nhiều ứng dụng và dịch vụ trí tuệ nhân tạo như trợ lý kỹ thuật số, điều khiển từ xa hỗ trợ giọng nói và phát hiện gian lận thẻ tín dụng.

Công nghệ deep learning được áp dụng rộng rãi trong các sản phẩm và dịch vụ hàng ngày như trợ lý kỹ thuật số, điều khiển từ xa hỗ trợ giọng nói và phát hiện gian lận thẻ tín dụng, cũng như trong các lĩnh vực mới nổi như ô tô tự lái.

1. Quy trình hoạt động của máy học

Quy trình làm việc với machine learning gồm 5 bước cụ thể từng bước trong machine learning như sau:

Data collection – thu thập dữ liệu: để máy tính có thể học được bạn cần có một bộ dữ liệu (dataset), bạn có thể tự thu thập chúng hoặc lấy các bộ dữ liệu đã được công bố trước đó. Lưu ý là bạn phải thu thập từ nguồn chính thống, có như vậy dữ liệu mới chính xác và máy có thể học một cách đúng đắng và đạt hiệu quả cao hơn.

Preparing data **-** chuẩn bị dữ liệu*:* bước này dùng để chuẩn hóa dữ liệu, loại bỏ các thuộc tính không cần thiết, gán nhãn dữ liệu, mã hóa một số đặc trưng, trích xuất đặc trưng, rút gọn dữ liệu nhưng vẫn đảm bảo kết quả… Bước này tốn thời gian nhất tỉ lệ thuận với số lượng dữ liệu bạn có. Bước 1 và 2 thường chiếm hơn 70% tổng thời gian thực hiện.

Training model – huấn luyện mô hình: bước này là bước huấn luyện cho mô hình hay chính là cho nó học trên dữ liệu đã thu thập và xử lý ở hai bước đầu.

Evaluating model – đánh giá mô hình: sau khi đã huấn luyện mô hình xong, cần dùng các độ đo để đánh giá mô hình, tùy vào từng độ đo khác nhau mà mô hình cũng được đánh giá tốt hay không khác nhau. Độ chính xác của mô hình đạt trên 80% được cho là tốt.

Improving the performance **–** cải thiện*:* sau khi đã đánh giá mô hình, các mô hình đạt độ chính xác không tốt thì cần được train lại, mô hình sẽ lặp lại từ bước 3, cho đến khi đạt độ chính xác như kỳ vọng. Tổng thời gian của 3 bước cuối rơi vào khoảng 30% tổng thời gian thực hiện.

1. Các thuật toán máy học cơ bản

* Neural networks

Mạng thần kinh, còn được gọi là mạng thần kinh nhân tạo (ANN) hoặc mạng thần kinh mô phỏng (SNN), là một tập hợp con của học máy.

Neural networks là một chuỗi các thuật toán cố gắng nhận ra các mối quan hệ cơ bản trong một tập hợp dữ liệu thông qua một quá trình bắt chước cách thức hoạt động của bộ não con người. Theo nghĩa này, mạng lưới thần kinh đề cập đến các hệ thống tế bào thần kinh, có bản chất hữu cơ hoặc nhân tạo.

Neural networks là thuật toán được dùng trong việc nhận dạng các mẫu và đóng một vai trò quan trọng trong các ứng dụng bao gồm dịch ngôn ngữ tự nhiên, nhận dạng hình ảnh, nhận dạng giọng nói và tạo hình ảnh.

* Linear regression

Hồi quy tuyến tính là một loại phân tích thống kê được sử dụng để dự đoán mối quan hệ giữa hai biến. Nó giả định mối quan hệ tuyến tính giữa biến độc lập và biến phụ thuộc và nhằm mục đích tìm ra đường phù hợp nhất mô tả mối quan hệ. Đường này được xác định bằng cách giảm thiểu tổng bình phương chênh lệch giữa giá trị dự đoán và giá trị thực tế.

Hồi quy tuyến tính thường được sử dụng trong nhiều lĩnh vực, bao gồm kinh tế, tài chính và khoa học xã hội, để phân tích và dự đoán xu hướng trong dữ liệu. Nó cũng có thể được mở rộng thành hồi quy tuyến tính bội, trong đó có nhiều biến độc lập và hồi quy logistic, được sử dụng cho các vấn đề phân loại nhị phân.

* Logistic regression

Hồi quy logistic là một kỹ thuật phân tích dữ liệu sử dụng toán học để tìm ra mối quan hệ giữa hai yếu tố dữ liệu. Sau đó, kỹ thuật này sử dụng mối quan hệ đã tìm được để dự đoán giá trị của những yếu tố đó dựa trên yếu tố còn lại. Dự đoán thường cho ra một số kết quả hữu hạn, như có hoặc không.

Ví dụ: giả sử bạn muốn đoán xem khách truy cập trang web của bạn sẽ nhấp vào nút thanh toán trong giỏ hàng của họ hay không. Phân tích hồi quy logistic xem xét hành vi của khách truy cập trước đây, chẳng hạn như thời gian dành cho trang web và số lượng các mặt hàng trong giỏ hàng. Quá trình phân tích này xác định rằng, trước đây, nếu khách truy cập dành hơn năm phút trên trang web và thêm hơn ba mặt hàng vào giỏ hàng, họ sẽ nhấp vào nút thanh toán. Nhờ vào thông tin này, sau đó, hàm hồi quy logistic có thể dự đoán hành vi của một khách mới truy cập trang web.

* Clustering

Là nhiệm vụ chia dân số hoặc điểm dữ liệu thành một số nhóm sao cho các điểm dữ liệu trong cùng một nhóm giống với các điểm dữ liệu khác trong cùng một nhóm và không giống với các điểm dữ liệu trong các nhóm khác. Về cơ bản, nó là một tập hợp các đối tượng trên cơ sở sự giống nhau và không giống nhau giữa chúng.

* Decision trees

Là thuật toán được sử dụng để dự đoán giá trị số (hồi quy) và phân loại dữ liệu. Decision trees sử dụng một chuỗi phân nhánh của các quyết định được liên kết có thể được biểu diễn bằng sơ đồ cây. Một trong những ưu điểm của decision trees là chúng dễ xác thực và kiểm tra, không giống thuật toán Neural networks.

* Random forests

Trong một khu rừng ngẫu nhiên, thuật toán máy học dự đoán một giá trị hoặc danh mục bằng cách kết hợp các kết quả từ một số cây quyết định.

* Support Vector Machines (SVM)
* Top of Form

Support Vector Machines (SVM) là một thuật toán máy học phổ biến được áp dụng cho cả bài toán phân loại và hồi quy. Mục tiêu của SVM là tìm ra ranh giới phân chia tối ưu giữa các nhóm dữ liệu.

Trong bài toán phân loại, SVM cố gắng tạo ra một ranh giới phân chia (gọi là siêu phẳng) sao cho khoảng cách từ các điểm dữ liệu gần nhất tới ranh giới này (còn gọi là margin) là lớn nhất. Các điểm dữ liệu gần nhất này được gọi là "vector hỗ trợ" (support vectors), từ đó xuất phát tên gọi của thuật toán.

SVM hoạt động hiệu quả trong các không gian chiều cao (high-dimensional space), có khả năng xử lý các tập dữ liệu có số lượng biến lớn. Nó cũng thích hợp cho các tập dữ liệu lớn, nhưng cần cân nhắc kỹ lưỡng khi lựa chọn các tham số quan trọng như loại kernel (ví dụ: linear, polynomial, radial basis function) để phù hợp với dữ liệu cụ thể và tránh overfitting.

* Naive Bayes Classifier

Naive Bayes Classifier, hay còn được gọi là Phân loại Bayes Đơn giản, là một mô hình phân loại trong máy học, được xây dựng dựa trên định lý Bayes với giả định "naive" (ngây thơ). Điều này ngụ ý rằng các biến độc lập với nhau, mặc dù trong thực tế thường không phải như vậy. Tuy nhiên, mặc dù có giả định này, mô hình Naive Bayes vẫn có hiệu suất tốt trong nhiều tình huống thực tế.

Cách hoạt động của Naive Bayes Classifier dựa trên tính toán xác suất để đánh giá xác suất của một điểm dữ liệu thuộc vào từng lớp phân loại. Dựa trên các xác suất này, điểm dữ liệu được phân loại vào lớp có xác suất cao nhất.

Mô hình Naive Bayes thường được áp dụng trong các bài toán như phân loại văn bản, phân loại email spam, phát hiện cảm xúc trong văn bản, và nhiều ứng dụng khác trong xử lý ngôn ngữ tự nhiên và phân loại dữ liệu. Mặc dù đơn giản, nhưng Naive Bayes có thể hoạt động hiệu quả và đem lại kết quả tốt cho nhiều loại dữ liệu.

* K-Nearest Neighbors (K-NN)

K-Nearest Neighbors (K-NN) là một thuật toán máy học không giám sát được sử dụng cho các bài toán phân loại và hồi quy. Trong K-NN, việc phân loại hoặc dự đoán giá trị mới được thực hiện dựa trên việc xem xét các điểm dữ liệu gần nhất trong tập huấn luyện.

Cụ thể, khi phải phân loại một điểm dữ liệu mới, K-NN tính toán khoảng cách giữa điểm dữ liệu này với tất cả các điểm trong tập huấn luyện. Sau đó, nó chọn ra K điểm gần nhất và dùng đa số phiếu (ví dụ: phân loại dựa trên lớp phổ biến nhất trong K điểm) để quyết định lớp của điểm dữ liệu mới.

Thuật toán K-NN không yêu cầu việc huấn luyện mô hình trước, mà chỉ cần lưu trữ tất cả các điểm dữ liệu trong tập huấn luyện để sau đó sử dụng cho việc phân loại hoặc dự đoán. Tuy nhiên, việc tính toán khoảng cách với mỗi điểm dữ liệu trong tập huấn luyện có thể làm tăng độ phức tạp tính toán của thuật toán khi số lượng điểm dữ liệu lớn.

1. **Ứng dụng của Machine learning**

Một số ứng dụng của **learning machine** như sau:

Speech recognition (Nhận diện giọng nói)*:* Nhận dạng giọng nói cho phép máy tính , ứng dụng và phần mềm hiểu và dịch dữ liệu giọng nói của con người thành văn bản cho các giải pháp kinh doanh . Mô hình nhận dạng giọng nói hoạt động bằng cách sử dụng trí tuệ nhân tạo (AI) để phân tích giọng nói và ngôn ngữ của bạn , xác định bằng cách học các từ bạn đang nói, sau đó xuất những từ đó với độ chính xác phiên âm dưới dạng nội dung mô hình hoặc dữ liệu văn bản trên màn hình.

Customer service (Chăm sóc khách hàng)*:* Chatbots trực tuyến đang thay thế các tác nhân con người trong hành trình của khách hàng, thay đổi cách nghĩ về sự tương tác của khách hàng trên website và nền tảng xã hội.

Computer vision (Thị giác máy tính): Công nghệ AI này cho phép máy tính lấy thông tin có ý nghĩa từ video, hình ảnh kỹ thuật số và các đầu vào trực quan khác, sau đó thực thi hành động thích hợp.

Recommendation engines (Công cụ gợi ý): Sử dụng dữ liệu hành vi tiêu dùng trong quá khứ, các thuật toán**AI learning** có thể giúp khám phá các xu hướng dữ liệu có thể được sử dụng để phát triển các chiến lược cross-sell hiệu quả hơn.

Fraud detection (Phát hiện gian lận)*:* Các ngân hàng và các tổ chức tài chính có thể sử dụng máy học để phát hiện các giao dịch đáng ngờ.

Machine learning được ứng dụng cực kỳ nhiều trong đời sống hiện nay trong mọi lĩnh vực:

* Tài chính – ngân hàng
* Sinh học
* Nông nghiệp
* Tìm kiếm, trích xuất thông tin
* Tự động hóa
* Robotics
* Hóa học
* Mạng máy tính
* Khoa học vũ trụ
* Quảng cáo
* Xử lý ngôn ngữ tự nhiên
* Thị giác máy tính

Và còn rất rất nhiều lĩnh vực mà machine learning có thể được áp dụng, machine learning tỏ ra cực kỳ hiệu quả, hơn hẳn con người trong cụ thể các lĩnh vực mà chúng được áp dụng.

Một ví dụ đơn giản về sự ứng dụng của Machine Learning là trong lĩnh vực dự báo thời tiết. Trong quá trình dự báo thời tiết, các chuyên gia sẽ sử dụng các phép tính và quan sát từ các thông tin về thời tiết trong quá khứ để dự đoán các điều kiện thời tiết trong tương lai. Tuy nhiên, khi có một lượng lớn quan sát đến hàng triệu hoặc hàng tỉ, con người sẽ gặp khó khăn trong việc xử lý và tính toán trên dữ liệu lớn đó. Hơn nữa, việc xử lý dữ liệu lớn như vậy có thể dẫn đến các sai sót trong dự đoán.

Ứng dụng Machine Learning trong việc dự báo thời tiết giúp máy tính học từ các quan sát được thu thập trong quá khứ. Nhờ vào khả năng xử lý lượng lớn dữ liệu một cách nhanh chóng và hiệu quả, máy tính có thể tạo ra dự đoán về thời tiết trong tương lai với độ chính xác cao hơn rất nhiều so với con người.

Điều này làm nổi bật sự quan trọng và hiệu quả của Machine Learning trong thời đại công nghệ 4.0. Việc hiểu và áp dụng Machine Learning không chỉ mang lại lợi thế lớn cho cá nhân mà còn đóng góp tích cực vào sự phát triển toàn cầu trong lĩnh vực công nghệ.

1. **XỬ LÝ NGÔN NGỮ TỰ NHIÊN**
2. Định nghĩa về xử lý ngôn ngữ tự nhiên

Xử lý ngôn ngữ tự nhiên là việc sử dụng các kỹ thuật xử lý ngôn ngữ tự nhiên (NLP) để phân tích và thao tác ngôn ngữ của con người trong bối cảnh tính toán. Điều này có thể bao gồm các nhiệm vụ như dịch ngôn ngữ, nhận dạng giọng nói, tạo văn bản, phân tích tình cảm và hơn thế nữa. NLP là một lĩnh vực đang phát triển nhanh chóng, với một loạt các ứng dụng trong các lĩnh vực như trí tuệ nhân tạo, truy xuất thông tin và tương tác giữa con người và máy tính.

Các tổ chức ngày nay có khối lượng lớn dữ liệu thoại và văn bản từ nhiều kênh liên lạc khác nhau như email, tin nhắn văn bản, bảng tin trên mạng xã hội, tệp video, tệp âm thanh và nhiều hơn nữa. Họ sử dụng phần mềm NLP để tự động xử lý dữ liệu này, phân tích ý định hoặc cảm xúc trong tin nhắn và phản hồi bằng người thật theo thời gian thực.

Xử lý ngôn ngữ tự nhiên (NLP) đóng một vai trò quan trọng trong việc hiệu quả hóa phân tích dữ liệu văn bản và giọng nói. Công nghệ này có khả năng xử lý sự đa dạng trong phương ngữ, tiếng lóng và những biến thường về ngôn ngữ thường gặp trong các cuộc trò chuyện hàng ngày. Các công ty tích hợp NLP vào các quá trình tự động, bao gồm:

* Xử lý, phân tích và lưu trữ lượng lớn tài liệu văn bản.
* Phân tích phản hồi từ khách hàng hoặc ghi âm cuộc gọi tới tổng đài.
* Triển khai chatbot cho dịch vụ khách hàng tự động.
* Trả lời các câu hỏi về người, sự vật, thời gian và địa điểm.
* Phân loại và trích xuất thông tin từ văn bản.

Ngoài ra, còn có thể tích hợp NLP trực tiếp vào các ứng dụng tương tác với khách hàng để tăng cường khả năng giao tiếp hiệu quả. Ví dụ, một Chatbot có thể phân tích và phân loại các truy vấn từ khách hàng, tự động trả lời các câu hỏi thường gặp và chuyển hướng các truy vấn phức tạp đến bộ phận hỗ trợ khách hàng. Sự tự động hóa này không chỉ giảm chi phí mà còn giúp nhân viên tránh mất thời gian vào các truy vấn lặp lại, từ đó cải thiện mức độ hài lòng của khách hàng.

1. Các thành phần trong xử lý ngôn tự nhiên

Về cơ bản, có hai thành phần chính bên trong một hệ thống NLP, gồm:

* **Hiểu ngôn ngữ tự nhiên** (Natural Language Understanding - NLU)

Hiểu ngôn ngữ tự nhiên là một phần quan trọng của NLP, tập trung vào việc phân tích ý nghĩa ẩn sau các câu. NLU cho phép phần mềm tìm kiếm các câu đồng nghĩa hoặc xử lý các từ có nhiều nghĩa khác nhau.

Trong phần này, các chủ đề cơ bản về NLU như ánh xạ từ đầu vào sang biểu diễn có ý nghĩa và phân tích các khía cạnh khác nhau của ngôn ngữ.

* **Tạo ngôn ngữ tự nhiên** (Natural Language Generation - NLG)

Tạo ngôn ngữ tự nhiên tập trung vào việc tạo ra văn bản hội thoại giống như con người dựa trên từ khóa hoặc chủ đề cụ thể. Quá trình này bao gồm:

* Lập kế hoạch văn bản, trong đó tìm kiếm và truy xuất nội dung từ cơ sở tri thức.
* Lập kế hoạch về câu bằng cách chọn câu từ cần thiết để thiết lập giọng điệu.
* Hiện thực hóa văn bản, ánh xạ kế hoạch thành cấu trúc câu có ý nghĩa.

1. Cách hoạt động của xử lý ngôn ngữ tự nhiên

* Tokenization

Tokenization là quá trình việc rút gọn từ và khôi phục nguyên thể từ bằng cách đơn giản hóa các từ về dạng gốc của chúng. Chẳng hạn, quá trình này có thể biến từ "starting" thành "start".

Loại bỏ từ dừng đảm bảo rằng các từ không đóng góp ý nghĩa đáng kể cho câu, như "cho" và "với", sẽ được loại bỏ khỏi văn bản.

* Morphological analysis

Phân tích hình thái là quá trình chia nhỏ một câu hoặc đoạn văn thành các đơn vị ngôn ngữ nhỏ nhất có ý nghĩa và xác định hình thái, loại từ và ý nghĩa của từng đơn vị đó.

* Parsing

Trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP), sử dụng các quy tắc ngữ pháp để xác định xem văn bản đầu vào có hợp lệ về mặt cú pháp hay không, đồng thời giúp hiểu nghĩa của văn bản đó. Quá trình này không chỉ phân tích cú pháp của văn bản mà còn kiểm tra lỗi cú pháp, do đó còn được biết đến là phân tích cú pháp.

* Named Entity Recognition (NER)

Nhận dạng thực thể được đặt tên (NER) có thể được coi là một quá trình làm cho máy có thể nhận dạng các đối tượng theo lớp của chúng và các thông số kỹ thuật khác. Ngoài ra, với thông tin về nhận dạng thực thể được đặt tên này, chúng tôi đã thảo luận về cách chúng tôi có thể triển khai NER bằng cách sử dụng thư viện spaCy và NLTK để nhận dạng con người, địa điểm, tổ chức, quốc gia, thời gian và sự kiện.

* Sentiment analysis

Phân tích cảm xúc (hoặc khai thác ý kiến) là một kỹ thuật xử lý ngôn ngữ tự nhiên (NLP) được sử dụng để xác định xem dữ liệu là tích cực, tiêu cực hay trung tính. Phân tích tình cảm thường được thực hiện trên dữ liệu văn bản để giúp doanh nghiệp theo dõi tình cảm thương hiệu và sản phẩm trong phản hồi của khách hàng , đồng thời hiểu được nhu cầu của khách hàng.

* Machine Translation

Machine translation là quá trình dịch tự động văn bản từ ngôn ngữ tự nhiên này sang ngôn ngữ tự nhiên khác bằng ứng dụng máy tính. Điều này có nghĩa là bạn thêm văn bản vào phần mềm dịch máy bằng ngôn ngữ nguồn và để công cụ tự động chuyển văn bản sang ngôn ngữ đích đã chọn.

* Question Answering

Question Answering (QA) là một lĩnh vực trong xử lý ngôn ngữ tự nhiên (NLP) tập trung vào phát triển hệ thống hoặc mô hình có khả năng hiểu và trả lời câu hỏi từ người dùng một cách tự động. Mục tiêu của QA là giúp máy tính hiểu câu hỏi được đặt và trả lời chúng bằng cách trích xuất thông tin từ nguồn dữ liệu có sẵn.

* Information Retrieval-based QA: Sử dụng kỹ thuật truy xuất thông tin để tìm kiếm và trích xuất câu trả lời từ nguồn dữ liệu lớn.
* Machine Learning-based QA: Sử dụng mô hình học máy để dự đoán câu trả lời dựa trên dữ liệu đào tạo.

1. Ứng dụng của xử lý ngôn ngữ tự nhiên

Nhận dạng chữ viết: Có hai loại nhận dạng, ví dụ như loại đầu tiên là nhận dạng bản in: nhận dạng và chuyển đổi văn bản sách giáo khoa. Khó hơn nữa là nhận dạng chữ viết tay, vốn khó vì chữ viết tay không có mẫu rõ ràng và thay đổi tùy theo từng người. Với chương trình nhận dạng xuất bản, hàng ngàn cuốn sách thư viện có thể được chuyển đổi thành tài liệu điện tử trong thời gian ngắn. Nhận dạng chữ viết tay của con người có ứng dụng trong tội phạm học và bảo mật thông tin (nhận dạng chữ ký điện tử).

Nhận dạng tiếng nói: Nhận biết âm thanh và chuyển đổi chúng thành văn bản tương ứng. Giúp mọi người sử dụng thiết bị nhanh hơn và dễ dàng hơn. Đây cũng là bước đầu tiên cần thực hiện để hiện thực hóa giao tiếp giữa con người và robot. Một ví dụ về ứng dụng nhận dạng giọng nói đang giúp đỡ người khiếm thị.

Tổng hợp tiếng nói: Từ văn bản được tự động tổng hợp thành giọng nói. Thay vì phải tự đọc nội dung của một cuốn sách hoặc trang web, nó sẽ tự động đọc. Giống như nhận dạng giọng nói, tổng hợp giọng nói là trợ thủ đắc lực cho người mù nhưng ngược lại, nó là bước cuối cùng trong quá trình giao tiếp giữa robot và con người.

Dịch tự động (Machine translate)*:* là một chương trình dịch tự động từ ngôn ngữ này sang ngôn ngữ khác. Nghiên cứu và ứng dụng dịch máy hiện nay. Phát triển nhanh chóng, ứng dụng có độ chính xác cao.

Tìm kiếm thông tin (Information retrieval): Hãy hỏi và chương trình sẽ tự động tìm nội dung phù hợp nhất. Thông tin đang bùng nổ, đặc biệt là với sự phát triển của Internet khiến việc lấy thông tin trở nên dễ dàng hơn bao giờ hết. Vấn đề là tìm được thông tin phù hợp và đặc biệt thông tin đó phải đáng tin cậy. Các công cụ tìm kiếm trên Internet như Google hay Yahoo hiện chỉ phân tích nội dung rất đơn giản dựa trên tần suất từ khóa, thứ hạng trang và nhiều tiêu chí đánh giá khác để đưa ra kết luận. Kết quả là nhiều lượt tìm kiếm không nhận được câu trả lời phù hợp, thậm chí dẫn đến các liên kết không liên quan do các website dùng thủ đoạn lừa đảo để giới thiệu sản phẩm (kỹ thuật SEO - Search Engine Optimization). Trên thực tế, cho đến nay chưa có công cụ tìm kiếm nào hiểu được ngôn ngữ tự nhiên của con người, ngoại trừ www.ask.com, được coi là “hiểu biết”; trả lời các câu hỏi có cấu trúc đơn giản nhất.

Tóm tắt văn bản: Từ một văn bản dài được tóm tắt thành một văn bản ngắn hơn nhưng vẫn chứa những nội dung thiết yếu nhất.

Khai phá dữ liệu và phát hiện tri thức*:* là tìm kiếm thông tin mới trong nhiều tài liệu khác nhau. Khai thác dữ liệu mô phỏng việc học tập của con người và tìm thấy thông tin hữu ích. Ở mức độ đơn giản với các công cụ tìm kiếm, nó cho phép bạn đặt câu hỏi theo cách mà công cụ tìm kiếm tìm ra câu trả lời phù hợp nhất.

Ví dụ ứng dụng của NLP:

Social Listening:

Trong thời đại cạnh tranh ngày càng gia tăng, việc lắng nghe khách hàng trên các kênh xã hội để hiểu xu hướng và thực hiện các chiến dịch tiếp thị phù hợp là rất cần thiết. Tuy nhiên, việc phân tích hàng nghìn, hàng nghìn bình luận của khách hàng trên các fan page, website… chưa bao giờ là điều dễ dàng. Chưa kể kết quả phân tích kém chính xác vì có quá nhiều thư rác, dữ liệu rác.

Những hạn chế của hệ thống Social Listening cũ sẽ không còn là nỗi lo với công nghệ xử lý ngôn ngữ tự nhiên thuần Việt của EM&AI:

Lọc dữ liệu rác/spam: tối ưu chất lượng dữ liệu, tăng hiệu suất xử lý và độ chính xác của kết quả phân tích.

Phân tích Intent/Entity/Sentiment: phát hiện và thông báo những biểu hiện bất thường, vấn đề, scandal mà doanh nghiệp có thể gặp phải.

Trích xuất Insight, báo cáo tự động: thông qua Entity Classification, hệ thống nhận diện những chủ đề được đề cập thường xuyên và đang hình thành xu hướng của cộng đồng mạng.

Helpdesk:

Bộ phận trợ giúp là bộ phận hỗ trợ dịch vụ của một công ty có vai trò cung cấp thông tin và hỗ trợ giải quyết các thắc mắc của khách hàng. Helpdesk hiện đang được sử dụng ở nhiều công ty thuộc bộ phận chăm sóc khách hàng. Phần mềm hỗ trợ khách hàng giúp các công ty giải quyết các yêu cầu của khách hàng hiệu quả hơn bằng hệ thống quản lý yêu cầu. Tuy nhiên, nếu số lượng vé quá lớn sẽ gây ra các vấn đề sau:

* Nhân viên không thể xử lý được tất các yêu cầu của khách hàng
* Không nhận diện được yêu cầu được ưu tiên xử lý.
* Nhân viên xử lý không đúng chuyên môn.
* Mất nhiều thời gian xử lý những vấn đề lặp lại.

Công nghệ xử lý ngôn ngữ tự nhiên thuần Việt có thể giúp ích được cho hệ thống Helpdesk của doanh nghiệp với những tính năng như:

* Tự động phân tích yêu cầu với công nghệ xử lý ngôn ngữ tự nhiên tiếng Việt sau đó phân loại, ưu tiên yêu cầu dựa trên phân tích cảm xúc (Sentiment Analytics).
* Tự động định tuyến yêu cầu của khách hàng đến nhân viên có đúng chuyên môn.
* Hỗ trợ phản hồi nhanh yêu cầu của khách hàng.
* Đừng để khách hàng quay lưng với doanh nghiệp bạn để tìm đến những dịch vụ tốt hơn. Hãy chuẩn bị những công cụ để mang lại cho khách hàng một dịch vụ chất lượng nhất.

Document Management System:

Nhiều công ty đau đầu trong việc tìm kiếm thông tin giữa hàng chục nghìn tài liệu được in và lưu trữ. Ngoài ra, việc không quản lý hoạt động in ấn của công ty dẫn đến lãng phí giấy in.

Doanh nghiệp có thể tiết kiệm tới 80% thời gian khôi phục tài liệu/văn bản bằng cách duy trì kiểm soát chặt chẽ hoạt động in ấn của công ty thông qua hệ thống quản lý tài liệu.

Một số tính năng của hệ thống Document Management có thể kể đến như:

* Phân tích nội dung: ứng dụng NLP và AI trong phân tích nội dung văn bản.
* Tóm lượt nội dung: trích xuất những nội dung quan trọng giúp quản lí và hỗ trợ truy xuất nhanh chóng.
* Bổ sung nội dung: cho phép truy cập vào văn bản trực tuyến, bổ sung thông tin, mã code mà không cần file gốc.
* Phân loại nội dung: phân loại văn bản theo nội dung chính, giúp sắp xếp và trích xuất dễ dàng nhanh chóng.
* Mã hóa chủ đề: chuyển đổi chủ đề dưới dạng QR code/barcode phục vụ cho việc quản lí, trích xuất dữ liệu/văn bản in ấn.

Hệ thống Document Management có thể kết nối với máy in vật lí hiện tại của doanh nghiệp để sử dụng những tính năng trên. Ứng dụng cao cho hầu hết doanh nghiệp thường xuyên in ấn và lưu trữ với khối lượng lớn.

Có thể kể tới một số bài toán xử lý ngôn ngữ tự nhiên trong phát triển chatbot bao gồm:

* Bài toán xác định ý định người dùng (intent detection)
* Bài toán trích xuất thông tin (named entity extraction)
* Quản lý hội thoại (dialogue management)
* Mô hình sinh hội thoại cho chatbot.

1. **PHÂN LOẠI VĂN BẢN**
2. Định nghĩa về văn bản

Khái niệm văn bản là một khái niệm cơ bản trong lĩnh vực ngôn ngữ học và trí tuệ nhân tạo, đề cập đến một đoạn văn bản hoặc một tập hợp các từ, cụm từ và câu được sắp xếp theo một trật tự nhất định để truyền đạt ý nghĩa hoặc thông điệp. Dưới dạng dữ liệu, văn bản thường được biểu diễn bằng các chuỗi ký tự hoặc các token, trong đó mỗi token có thể là một từ, một cụm từ, một ký tự hoặc một số ký tự đặc biệt.

Chi tiết hơn, một văn bản có thể bao gồm các thành phần sau:

Từ và Cụm Từ: Là các đơn vị cơ bản của văn bản, bao gồm các từ đơn (như "nhà", "đi", "là") hoặc các cụm từ (như "nhà hàng", "điều kiện thời tiết", "trong khi").

Câu: Là một chuỗi các từ và cụm từ được sắp xếp theo một cách cú pháp nhất định để truyền đạt một ý nghĩa hoặc một thông điệp. Mỗi câu thường kết thúc bằng dấu chấm câu, dấu phẩy, hoặc các dấu câu khác.

Đoạn: Là một nhóm các câu liên quan nhau, thường chứa một ý chính hoặc một phần của thông điệp được truyền đạt trong văn bản.

Tài liệu Văn Bản: Là một tập hợp các đoạn văn bản và các phần khác nhau của văn bản, có thể là một bài báo, một trang web, một email, một cuốn sách, v.v.

Ngôn Ngữ và Cú Pháp: Văn bản được viết bằng một ngôn ngữ cụ thể và tuân thủ các quy tắc cú pháp của ngôn ngữ đó, bao gồm cú pháp ngữ pháp, cú pháp từ vựng, và cú pháp ngữ cảnh.

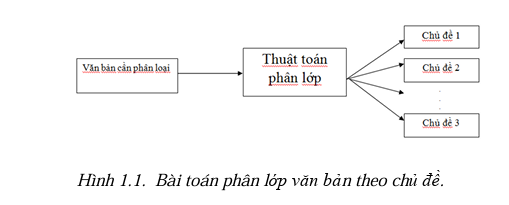
Văn bản không chỉ là một nguồn thông tin quan trọng mà còn là đối tượng nghiên cứu và xử lý quan trọng trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP), máy học, khai phá dữ liệu, và nhiều lĩnh vực khác.

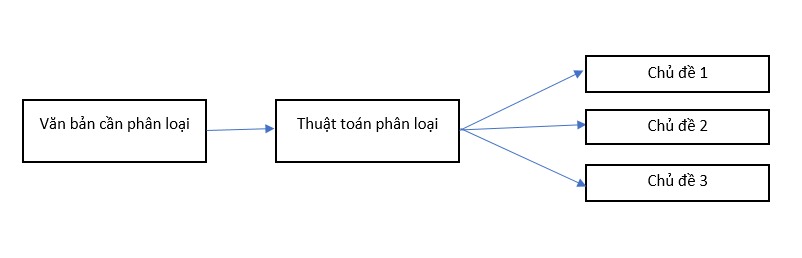
1. Định nghĩa về phân lớp văn bản

Phân loại văn bản là quá trình tự động gán nhãn (tên lớp / nhãn lớp) cho các văn bản ngôn ngữ tự nhiên vào một hoặc nhiều lớp được xác định trước. Đây là một bước quan trọng trong khai phá dữ liệu văn bản, nơi các văn bản được phân loại dựa trên nội dung của chúng vào các chủ đề đã được xác định trước.

Phân loại văn bản có nhiều ứng dụng, bao gồm hỗ trợ trong quá trình tìm kiếm thông tin, chiết lọc thông tin, lọc văn bản hoặc tự động dẫn đường cho các văn bản tới các chủ đề xác định trước. Quá trình này có thể được thực hiện thủ công hoặc tự động bằng cách sử dụng các kỹ thuật học máy có giám sát.

Các hệ thống phân loại văn bản có thể được áp dụng trong nhiều ngữ cảnh, như phân loại tài liệu trong thư viện điện tử, phân loại các bài báo trên các trang tin điện tử, và nhiều ứng dụng khác. Khi được triển khai hiệu quả, các hệ thống này có thể cung cấp kết quả đáng kể, giúp tối ưu hóa công việc và tạo ra lợi ích đáng kể cho người sử dụng.



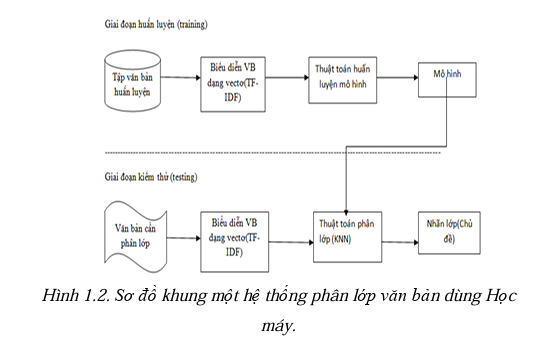


1. Mô hình phân loại văn bản

Phân loại văn bản là một bài toán trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP), trong đó mục tiêu là tự động gán các nhãn hoặc danh mục cho các văn bản dựa trên nội dung của chúng. Đây là một bước quan trọng trong khai phá dữ liệu văn bản, giúp tổ chức và hiểu được các tài liệu văn bản một cách tự động và hiệu quả. Dưới đây là các bước cơ bản của bài toán phân loại văn bản:

* Thu thập dữ liệu: Bước đầu tiên là thu thập các tài liệu văn bản cần phân loại từ các nguồn khác nhau như trang web, cơ sở dữ liệu, tập tin văn bản, v.v.
* Tiền xử lý dữ liệu: Dữ liệu văn bản thường cần được tiền xử lý để loại bỏ các thành phần không cần thiết như dấu câu, stop words (các từ phổ biến như "and", "a", "an"), và thực hiện việc chuẩn hóa văn bản như việc chuyển đổi chữ in thường thành chữ in hoa.
* Biểu diễn văn bản: Các văn bản sau khi đã được tiền xử lý cần được biểu diễn dưới dạng các vector số hóa để có thể được sử dụng trong các mô hình máy học. Các phương pháp phổ biến bao gồm TF-IDF (Term Frequency-Inverse Document Frequency) và Word Embeddings (như Word2Vec, GloVe).
* Xây dựng mô hình: Tiếp theo, một mô hình máy học hoặc học sâu (deep learning) được huấn luyện bằng dữ liệu huấn luyện đã được biểu diễn để dự đoán nhãn cho các văn bản mới. Các mô hình phổ biến bao gồm Naive Bayes, Support Vector Machines (SVM), Decision Trees, Random Forests, và Neural Networks.
* Đánh giá mô hình: Mô hình được đánh giá bằng cách sử dụng các phương pháp đánh giá như cross-validation hoặc holdout validation để đảm bảo hiệu suất của nó trên dữ liệu kiểm tra.
* Tinh chỉnh và triển khai: Nếu cần thiết, mô hình được tinh chỉnh và cải thiện trước khi triển khai vào sản phẩm hoặc ứng dụng thực tế.

Bài toán phân loại văn bản có ứng dụng rộng rãi trong nhiều lĩnh vực như phân loại email (spam vs. non-spam), phân loại tin tức, phát hiện cảm xúc trong văn bản, và tổ chức thông tin trong các hệ thống quản lý tri thức.



1. Ưng dụng của phân loại văn bản trong thực tiễn

Việc phân loại văn bản có rất nhiều ứng dụng trong thực tế, từ các lĩnh vực công nghệ thông tin đến ngành y tế và tài chính. Dưới đây là một số ứng dụng cụ thể của việc phân loại văn bản trong thực tế:

* Phân loại email:

Gmail, Outlook và các dịch vụ email khác sử dụng phân loại văn bản để tự động phân loại thư đến vào các hộp thư như hộp thư đến chính, hộp thư quan trọng hoặc hộp thư rác.

* Phân loại tài liệu và tài nguyên công ty:

Công ty sử dụng phân loại văn bản để tổ chức và quản lý tài liệu nội bộ, từ hợp đồng và báo cáo đến chính sách và quy trình.

* Phân loại sản phẩm và dịch vụ:

Các trang web thương mại điện tử sử dụng phân loại văn bản để tự động phân loại và gắn nhãn cho các sản phẩm, giúp người dùng dễ dàng tìm kiếm và mua hàng.

* Phân loại dữ liệu y tế:

Trong lĩnh vực y tế, phân loại văn bản được sử dụng để tổ chức và phân loại thông tin y tế, bao gồm bệnh án, báo cáo y khoa và thông tin liên quan đến bệnh nhân.

* Phân loại tài chính:

Các tổ chức tài chính sử dụng phân loại văn bản để phân loại và phân tích thông tin tài chính, bao gồm báo cáo tài chính, dự báo xu hướng thị trường và phân tích dữ liệu tài chính.

* Phân loại tin tức và bài viết:

Các trang web tin tức và blog sử dụng phân loại văn bản để tự động gắn nhãn và phân loại các tin tức và bài viết theo chủ đề hoặc danh mục.

* Phân loại ý kiến và phản hồi khách hàng:

Các doanh nghiệp sử dụng phân loại văn bản để phân tích ý kiến và phản hồi của khách hàng, từ đánh giá sản phẩm đến bài đánh giá trên mạng xã hội.

* Phân loại dữ liệu trong ngành sản xuất:

Trong ngành sản xuất, phân loại văn bản có thể được sử dụng để tổ chức và quản lý dữ liệu liên quan đến quy trình sản xuất, bảo trì và kiểm soát chất lượng.

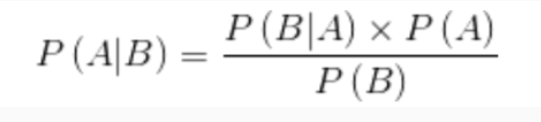
**Naive Bayes**

1. Định nghĩa về Naive Bayes

Naive Bayes Classification (NBC) là một phương pháp dựa trên Định lý Bayes về xác suất, được sử dụng để đưa ra dự đoán và phân loại dữ liệu dựa trên thông tin quan sát và thống kê từ dữ liệu. NBC là một trong những thuật toán phổ biến trong lĩnh vực Machine Learning, được áp dụng rộng rãi để tạo ra các dự đoán chính xác từ tập dữ liệu thu thập được. Thuật toán này được đánh giá cao vì tính dễ hiểu và độ chính xác cao của nó. NBC thuộc nhóm thuật toán Học có giám sát (Supervised Machine Learning Algorithms), nghĩa là nó học từ các ví dụ được cung cấp từ các mẫu dữ liệu đã biết trước.

ĐỊNH LÝ:

Định lý Bayes cho phép tính xác suất xảy ra của một sự kiện ngẫu nhiên A khi biết sự kiện liên quan B đã xảy ra. Xác suất này được ký hiệu là P(A|B), và đọc là “xác suất của A nếu có B”. Đại lượng này được gọi xác suất có điều kiện hay xác suất hậu nghiệm vì nó được rút ra từ giá trị được cho của B hoặc phụ thuộc vào giá trị đó



Theo định lí Bayes, P(A|B) sẽ phụ thuộc vào 3 yếu tố:  
- Xác suất xảy ra A của riêng nó, không quan tâm đến B. Kí hiệu là P(A).  
- Xác suất xảy ra B của riêng nó, không quan tâm đến A. Kí hiệu là P(B).  
- Xác suất xảy ra B khi biết A xảy ra. Kí hiệu là P(B|A). Đại lượng này gọi là khả năng (likelihood) xảy ra B khi biết A đã xảy ra.

* Xác suất xảy ra A của riêng nó, không quan tâm đến B. Kí hiệu là P(A).
* Xác suất xảy ra B của riêng nó, không quan tâm đến A. Kí hiệu là P(B).
* Xác suất xảy ra B khi biết A xảy ra. Kí hiệu là P(B|A). Đại lượng này gọi là khả năng (likelihood) xảy ra B khi biết A đã xảy ra.

1. Cách hoạt động của Naive Bayes

Cách hoạt động của thuật toán Naive Bayes dựa trên định lý Bayes về xác suất. Thuật toán này giả định rằng các đặc trưng đầu vào độc lập với nhau, tức là sự xuất hiện của một đặc trưng không phụ thuộc vào sự xuất hiện của các đặc trưng khác. Dựa trên giả định này, Naive Bayes tính toán xác suất của một lớp dựa trên các đặc trưng của mẫu dữ liệu.

Cách hoạt động như sau:

* + Xác định các đặc trưng: Đầu tiên, thuật toán cần xác định các đặc trưng hoặc thuộc tính của dữ liệu. Ví dụ, trong bài toán phân loại email, các đặc trưng có thể bao gồm từ vựng xuất hiện trong email, số lượng từ trong email, hoặc các đặc điểm của email như độ dài, tỷ lệ các ký tự in hoa, v.v.
  + Tính toán xác suất của mỗi lớp: Naive Bayes tính toán xác suất của mỗi lớp dựa trên các đặc trưng của mẫu dữ liệu. Đối với mỗi lớp, nó tính toán xác suất có điều kiện của mỗi đặc trưng dựa trên dữ liệu huấn luyện.
  + Áp dụng định lý Bayes: Sau khi tính toán xác suất có điều kiện cho mỗi lớp, Naive Bayes áp dụng định lý Bayes để tính toán xác suất của mỗi lớp dựa trên các đặc trưng của mẫu dữ liệu. Điều này giúp dự đoán xác suất của mỗi lớp cho mẫu dữ liệu mới.
  + Chọn lớp có xác suất cao nhất: Naive Bayes chọn lớp có xác suất cao nhất là lớp dự đoán cho mẫu dữ liệu. Nó xác định lớp mà mẫu dữ liệu có xác suất cao nhất thuộc vào và gán lớp đó cho dự đoán cuối cùng.
  + Đưa ra dự đoán: Cuối cùng, thuật toán đưa ra dự đoán bằng cách gán lớp có xác suất cao nhất cho mẫu dữ liệu. Điều này có nghĩa là nó xác định lớp mà mẫu dữ liệu được phân loại vào và đưa ra dự đoán tương ứng.

1. Ưu điểm và nhược điểm của Naive Bayes

Ưu điểm:

* Dễ thực hiện và nhanh chóng: Thuật toán Naive Bayes thực hiện phân loại một cách dễ dàng và nhanh chóng. Vì vậy, nó thích hợp cho các bài toán có thời gian đáp ứng nhanh hoặc cần sự đơn giản.
* Hội tụ nhanh hơn so với các mô hình khác: Naive Bayes thường hội tụ nhanh hơn các mô hình phân biệt khác như hồi quy logistic, giúp giảm thời gian huấn luyện.
* Yêu cầu ít dữ liệu đào tạo: Thuật toán Naive Bayes yêu cầu ít dữ liệu đào tạo hơn so với một số mô hình khác, điều này có thể rất hữu ích khi dữ liệu huấn luyện có hạn.
* Khả năng mở rộng cao: Naive Bayes có khả năng mở rộng tốt và chia tỷ lệ tuyến tính với số lượng đặc trưng dự đoán và điểm dữ liệu.
* Dự đoán xác suất và xử lý dữ liệu liên tục và rời rạc: Thuật toán Naive Bayes có thể đưa ra dự đoán xác suất, giúp người dùng hiểu rõ hơn về độ tin cậy của kết quả phân loại. Ngoài ra, nó cũng có khả năng xử lý cả dữ liệu liên tục và rời rạc.
* Phân loại cả hai bài toán nhị phân và đa lớp: Naive Bayes có thể được sử dụng cho cả hai loại bài toán phân loại: nhị phân (hai lớp) và đa lớp (nhiều lớp). Điều này làm cho nó trở thành một lựa chọn linh hoạt cho nhiều loại bài toán.

Nhược điểm:

* Giả định độc lập: Một trong những giả định chính của Naive Bayes là giả định về độc lập giữa các đặc trưng khi biết lớp. Trong thực tế, các đặc trưng thường không độc lập hoàn toàn, điều này có thể dẫn đến các dự đoán không chính xác nếu có sự tương quan giữa các đặc trưng.
* Tần số không: Nếu một đặc trưng trong dữ liệu test không xuất hiện trong tập dữ liệu huấn luyện, Naive Bayes sẽ gán xác suất của nó bằng 0, dẫn đến việc mô hình không thể tạo ra dự đoán cho những trường hợp này.
* Dự đoán không chính xác khi dữ liệu phức tạp: Naive Bayes thường hoạt động tốt trên dữ liệu đơn giản và có cấu trúc rõ ràng. Tuy nhiên, khi dữ liệu phức tạp và có tương quan cao giữa các đặc trưng, Naive Bayes có thể tạo ra các dự đoán không chính xác.
* Dễ bị ảnh hưởng bởi dữ liệu nhiễu: Naive Bayes có thể bị ảnh hưởng bởi dữ liệu nhiễu hoặc các đặc trưng không quan trọng. Điều này có thể dẫn đến việc mô hình không hoạt động tốt trên dữ liệu thực tế.
* Không phản ứng tốt với biến đầu vào liên tục: Naive Bayes thường không phản ứng tốt với các biến đầu vào liên tục và yêu cầu việc chuyển đổi các biến này thành các biến rời rạc trước khi huấn luyện mô hình.

1. Ứng dụng của Naive Bayes

* Phân loại văn bản: Naive Bayes thường được sử dụng trong phân loại văn bản, chẳng hạn như phân loại email vào thư rác và thư không phải thư rác. Nó cũng có thể được áp dụng trong việc phân loại văn bản theo chủ đề hoặc ngôn ngữ.
* Dự đoán cảm xúc từ văn bản: Thuật toán Naive Bayes cũng được sử dụng để phân tích cảm xúc từ văn bản, như xác định liệu một bình luận trên mạng xã hội là tích cực, tiêu cực hay trung tính.
* Lọc spam: Với khả năng phân loại hiệu quả và tính nhanh nhẹn, Naive Bayes thường được sử dụng trong các hệ thống lọc thư rác để phân biệt giữa email spam và email hợp lệ.
* Hệ thống gợi ý: Naive Bayes có thể được tích hợp vào hệ thống gợi ý để đề xuất sản phẩm hoặc nội dung dựa trên lịch sử hoạt động của người dùng.
* Phân loại dữ liệu y tế: Trong lĩnh vực y học, Naive Bayes có thể được sử dụng để phân loại dữ liệu y tế, như xác định liệu một bệnh nhân có mắc một loại bệnh nhất định hay không dựa trên các dữ liệu y tế.
* Phát hiện gian lận: Naive Bayes cũng có thể được sử dụng trong các hệ thống phát hiện gian lận, chẳng hạn như phát hiện gian lận trong giao dịch tài chính hoặc giao dịch trực tuyến.
* Phân loại tin tức: Thuật toán này có thể được áp dụng trong việc phân loại tin tức theo chủ đề, như phân loại tin tức thể thao, tin tức chính trị, tin tức khoa học, v.v.

1. **KỸ THUẬT TF – IDF**
2. Định nghĩa TF – IDF

**Tần số từ (TF - Term Frequency)**

Term Frequency (TF), hay Tần suất xuất hiện của từ, là số lần một từ xuất hiện trong một văn bản. Do các văn bản có thể có độ dài khác nhau, một số từ có thể xuất hiện nhiều lần trong một văn bản dài hơn so với một văn bản ngắn. Do đó, để chuẩn hóa, term frequency thường được chia cho độ dài của văn bản, tức là tổng số từ trong một văn bản.



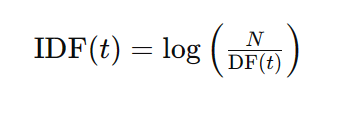
Ví dụ: Nếu từ "machine" xuất hiện 10 lần trong một văn bản có tổng cộng 1000 từ, thì TF của "machine" trong văn bản đó là 10/1000=0,01.

**Tần số văn bản nghịch đảo (IDF - Inverse Document Frequency)**

IDF là viết tắt của "Inverse Document Frequency" (Tần số văn bản nghịch đảo). Đây là một khái niệm quan trọng trong xử lý ngôn ngữ tự nhiên và khai phá dữ liệu văn bản, được sử dụng để đo lường mức độ quan trọng của một từ trong một tập hợp các văn bản.

Cụ thể, IDF đo lường tần suất xuất hiện của một từ trong toàn bộ tập hợp các văn bản và định lượng mức độ quan trọng của từ đó. Mục đích của IDF là tìm ra những từ xuất hiện hiếm trong tập hợp các văn bản, tức là những từ đặc biệt có khả năng đặc trưng và có thể phân biệt các văn bản với nhau.

Công thức tính IDF cho một từ t thường là số lượng văn bản trong tập dữ liệu chia cho Document Frequency (DF) của từ t. Để tránh việc chia cho 0, có thể thêm một hằng số vào mẫu số. Một công thức phổ biến cho IDF là:



Trong đó:

N là tổng số văn bản trong tập dữ liệu.

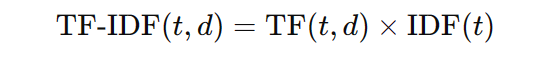
DF(t) là Tần số văn bản của từ t, tức là số lượng văn bản trong tập dữ liệu mà từ t xuất hiện trong đó.

Ví dụ: Nếu trong một tập dữ liệu có tổng cộng 10,000 văn bản, và từ "machine" xuất hiện trong 1,000 văn bản, thì IDF của "machine" là log(10000/1000)= log(10)= 1.

Hàm logarithm được sử dụng để giảm thiểu ảnh hưởng của các từ phổ biến mà không cung cấp nhiều thông tin quan trọng. Càng cao giá trị IDF, tức là càng hiếm khi từ xuất hiện trong các văn bản, càng được coi là quan trọng và có khả năng phân biệt cao giữa các văn bản.

**Tần số từ - Tần số văn bản nghịch đảo (TF-IDF)**

TF-IDF, viết tắt của "Term Frequency – Inverse Document Frequency", là một chỉ số thống kê thể hiện mức độ quan trọng của một từ trong một văn bản so với một tập hợp các văn bản. Giá trị TF-IDF tăng tương ứng với số lần một từ xuất hiện trong văn bản, nhưng thường được điều chỉnh bằng tần số của từ trong toàn bộ tập hợp các văn bản, giúp điều chỉnh thực tế là một số từ xuất hiện thường xuyên hơn nói chung.



Ví dụ: Nếu TF của từ "machine" trong một văn bản là 0.01 và IDF của "machine" trong toàn bộ tập dữ liệu là 1, thì TF-IDF của "machine" trong văn bản đó là 0,01x1= 0,01.

1. Vai trò của TF – IDF

TF-IDF (Term Frequency-Inverse Document Frequency) có vai trò quan trọng trong xử lý ngôn ngữ tự nhiên và khai phá dữ liệu văn bản, và được sử dụng rộng rãi trong nhiều ứng dụng như tìm kiếm thông tin, phân loại văn bản, và trích xuất thông tin. Dưới đây là một số vai trò chính của TF-IDF:

* Xác định từ khóa quan trọng: TF-IDF giúp xác định những từ khóa quan trọng trong một văn bản bằng cách tăng cường trọng số cho những từ xuất hiện ít trong văn bản đó nhưng xuất hiện nhiều trong các văn bản khác.
* Cải thiện hiệu suất tìm kiếm: Trong các hệ thống tìm kiếm, TF-IDF được sử dụng để đánh giá độ phù hợp của một văn bản với một truy vấn tìm kiếm. Các từ có TF-IDF cao hơn được coi là những từ khóa quan trọng và giúp cải thiện chất lượng kết quả tìm kiếm.
* Phân loại văn bản: Trong các bài toán phân loại văn bản, TF-IDF được sử dụng để biểu diễn các văn bản dưới dạng vector có kích thước cố định. Các mô hình học máy sau đó có thể được huấn luyện trên các vector này để phân loại các văn bản vào các lớp khác nhau.
* Trích xuất thông tin: TF-IDF có thể được sử dụng để trích xuất thông tin quan trọng từ các văn bản, như trích xuất từ khóa, phát hiện chủ đề hoặc phân loại văn bản vào các danh mục cụ thể.
* Loại bỏ từ không quan trọng: Các từ có TF-IDF thấp có thể được loại bỏ hoặc được coi là không quan trọng trong các ứng dụng xử lý ngôn ngữ tự nhiên như làm sạch văn bản hoặc giảm kích thước của dữ liệu.

Tóm lại, TF-IDF đóng vai trò quan trọng trong việc biểu diễn và đánh giá mức độ quan trọng của các từ trong văn bản, giúp cải thiện hiệu suất của các hệ thống tìm kiếm và phân loại văn bản, và hỗ trợ trong quá trình trích xuất thông tin từ dữ liệu văn bản.

1. Ưu điểm và nhược điểm của TF-IDF

Ưu điểm:

* Tính tương đối: TF-IDF không chỉ đơn giản là đếm số lần xuất hiện của một từ trong một tài liệu mà còn xem xét tần suất của từ đó so với tần suất của nó trong tập hợp các tài liệu khác. Điều này giúp làm nổi bật những từ quan trọng trong một tài liệu cụ thể so với các từ thông thường.
* Giảm nhiễu: Những từ phổ biến như "và", "hoặc" có thể xuất hiện nhiều lần trong các tài liệu, nhưng chúng thường không mang lại nhiều ý nghĩa. TF-IDF giúp giảm nhiễu bằng cách giảm tầm quan trọng của những từ này trong việc phân tích và tạo ra biểu đồ từ khóa.
* Đa dạng phương pháp: TF-IDF có thể được sử dụng cho nhiều mục đích trong xử lý ngôn ngữ tự nhiên, bao gồm phân loại văn bản, trích xuất từ khóa, và tìm kiếm thông tin.
* Dễ hiểu và triển khai: Phương pháp này khá đơn giản để hiểu và triển khai, không cần nhiều kiến thức chuyên sâu về toán học hay machine learning.

Nhược điểm:

* Không xử lý ngữ cảnh: TF-IDF không xem xét ngữ cảnh của từ trong văn bản. Điều này có nghĩa là nó có thể bỏ qua mối quan hệ giữa các từ hoặc không xử lý tốt trong các trường hợp cụ thể.
* Không xử lý đồng nghĩa và từ đồng nghĩa: TF-IDF không xem xét các từ đồng nghĩa (synonyms) hoặc từ đồng nghĩa (homographs), điều này có thể làm mất đi một số thông tin quan trọng trong văn bản.
* Nhược điểm với văn bản ngắn: Trong trường hợp các văn bản ngắn, TF-IDF có thể không hiệu quả vì sự hiện diện và tần suất của từ không đủ để phản ánh tầm quan trọng của chúng.

Mặc dù có nhược điểm nhất định, TF-IDF vẫn là một phương pháp mạnh mẽ và phổ biến trong xử lý ngôn ngữ tự nhiên, đặc biệt là trong các ứng dụng như tìm kiếm thông tin và phân loại văn bản.

import os

import random

import string

import nltk

nltk.download('punkt')

nltk.download('stopwords')

from nltk import word\_tokenize

from collections import defaultdict

from nltk import FreqDist

from nltk.corpus import stopwords

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.naive\_bayes import MultinomialNB

from sklearn import metrics

import pickle

Đầu tiên cần import các thư viện cần thiết như os, random, string, nltk và sklearn. Thư viện os được sử dụng để tương tác với hệ điều hành, trong khi random được sử dụng để tạo số ngẫu nhiên. Thư viện nltk là một thư viện phổ biến trong xử lý ngôn ngữ tự nhiên, và trong đoạn mã này, chúng ta sử dụng nó để tải xuống các tài nguyên cần thiết như tokenizer và danh sách từ dừng. Thư viện sklearn cung cấp các công cụ để xây dựng và đánh giá các mô hình học máy.

Sau đó, mã tiến hành tiền xử lý dữ liệu văn bản. Các bước tiền xử lý này bao gồm việc tải xuống và tách từ, loại bỏ các từ dừng và các ký tự đặc biệt, và chuẩn hóa văn bản thành dạng thống nhất. Điều này giúp làm sạch và chuẩn hóa dữ liệu để sử dụng cho việc huấn luyện mô hình.

Tiếp theo, mã sử dụng các công cụ từ thư viện sklearn để biến đổi văn bản thành các biểu diễn số hóa. Cụ thể, nó sử dụng TfidfVectorizer và CountVectorizer để tạo ma trận tf-idf và ma trận đếm từ từ các văn bản. Các biểu diễn số hóa này sẽ được sử dụng để huấn luyện mô hình phân loại.

Cuối cùng, mã sử dụng mô hình Multinomial Naive Bayes từ sklearn để huấn luyện và phân loại văn bản vào các chủ đề khác nhau. Sau đó, nó đánh giá hiệu suất của mô hình bằng cách sử dụng các phương pháp đánh giá như accuracy, precision và recall.

stop\_words = set(stopwords.words('english'))

# thêm stopword

stop\_words.add('said')

stop\_words.add('mr')

Hàm words() từ thư viện stopwords của nltk được sử dụng để tải danh sách các từ dừng trong tiếng Anh. Các từ dừng là các từ phổ biến không mang ý nghĩa và thường bị loại bỏ trong quá trình xử lý văn bản.

Hàm set() được sử dụng để chuyển danh sách các từ dừng thành một tập hợp (set) để tối ưu hóa việc truy xuất và kiểm tra các phần tử. Tập hợp này chứa các từ dừng tiếng Anh.

Hai dòng cuối được sử dụng để thêm các từ dừng bổ sung vào tập hợp stop\_words. Trong trường hợp này, các từ "said" và "mr" được thêm vào danh sách các từ dừng. Điều này giúp loại bỏ các từ này khỏi văn bản trong quá trình tiền xử lý để cải thiện chất lượng của dữ liệu đầu vào cho mô hình phân loại văn bản.

# đường dẫn thư mục cơ sở trong đó các tệp tin văn bản được lưu trữ

BASE\_DIR = 'D:\\DA2\\Computer\_Science\_Project2\\DEMO\\bbc'

# danh sách các nhãn đại diện cho các chủ đề hoặc loại văn bản

LABELS = ['business', 'entertainment', 'politics', 'sport', 'tech']

LABELS chứa một danh sách các nhãn đại diện cho các chủ đề hoặc loại văn bản. Danh sách này bao gồm các nhãn: 'business', 'entertainment', 'politics', 'sport', và 'tech'. Các nhãn này đại diện cho các chủ đề hoặc loại văn bản mà mô hình phân loại được huấn luyện để nhận diện.

def create\_data\_set():

    with open('data.txt', 'w', encoding='utf8') as outfile:

        for label in LABELS:

            dir = '%s/%s' % (BASE\_DIR, label)

            for filename in os.listdir(dir):

                fullfilename = '%s/%s' % (dir, filename)

                print(fullfilename)

                with open(fullfilename, 'rb') as file:

                    text = file.read().decode(errors= 'replace').replace('\n', '')

                    outfile.write('%s\t%s\t%s\n' % (label, filename, text))

Hàm create\_data\_set() này sẽ tạo ra một tệp 'data.txt' chứa dữ liệu từ tất cả các tệp tin văn bản trong các thư mục của các nhãn được xác định, với mỗi bản ghi bao gồm nhãn, tên tệp và nội dung của văn bản.

# tạo hàm để đọc dữ liệu từ file data.txt biến đổi thành danh sách dữ liệu gồm nhãn

def setup\_docs():

    docs = [] # (label, text) Khởi tạo một danh sách rỗng để chứa các bộ dữ liệu

    with open('data.txt', 'r', encoding='utf8') as datafile:

        for row in datafile:

            parts = row.split('\t') # Tách mỗi dòng thành các phần bằng cách sử dụng dấu tab ('\t') làm dấu phân cách

            doc = ( parts[0], parts[2].strip() ) # Tạo một bộ dữ liệu mới, trong đó phần tử đầu tiên là nhãn, và phần tử thứ hai là văn bản

            docs.append(doc) # Thêm bộ dữ liệu mới vào danh sách docs

        return docs # Trả về danh sách docs

Đoạn mã trên định nghĩa một hàm `setup\_docs()` để đọc dữ liệu từ tệp 'data.txt' và chuyển đổi nó thành một danh sách các bộ dữ liệu gồm nhãn và văn bản tương ứng. Hàm này hoạt động bằng cách duyệt qua từng dòng trong tệp 'data.txt', tách dòng thành các phần bằng dấu phân cách tab (`'\t'`), và sau đó tạo một bộ dữ liệu mới từ nhãn và văn bản trong mỗi dòng. Cuối cùng, danh sách các bộ dữ liệu này được trả về để sử dụng trong việc huấn luyện và kiểm tra mô hình phân loại văn bản.

Hàm này là một phần quan trọng trong quá trình tiền xử lý dữ liệu trước khi huấn luyện mô hình, vì nó cho phép bạn tải dữ liệu từ tệp văn bản đã được chuẩn bị trước và biến đổi nó thành một định dạng phù hợp để sử dụng cho các bước tiếp theo trong quy trình xây dựng mô hình phân loại văn bản.

# làm sạch văn bản đầu vào

def clean\_text(text):

    # loại bỏ dấu câu bằng cách thay dấu câu bằng khoảng trắng

    text = text.translate(str.maketrans('', '', string.punctuation))

    # chuyển đổi văn bản thành chữ thường

    text = text.lower()

    return text

Đoạn mã trên định nghĩa một hàm `clean\_text()` để thực hiện việc làm sạch văn bản đầu vào trước khi tiến hành tiền xử lý. Hàm này thực hiện hai bước chính để chuẩn bị dữ liệu cho việc xử lý tiếp theo:

Loại bỏ các dấu câu: Dòng mã sử dụng phương thức `translate()` để loại bỏ các dấu câu từ văn bản. Các dấu câu được thay thế bằng khoảng trắng, giúp làm sạch văn bản và loại bỏ yếu tố không cần thiết trong quá trình xử lý.

Chuyển đổi văn bản thành chữ thường: Hàm `lower()` được sử dụng để chuyển đổi tất cả các ký tự trong văn bản thành chữ thường. Điều này giúp đồng nhất hóa văn bản, loại bỏ sự phân biệt giữa chữ hoa và chữ thường trong quá trình xử lý dữ liệu.

Kết quả, hàm `clean\_text()` trả về văn bản sau khi đã được làm sạch và chuẩn hóa, sẵn sàng cho các bước tiền xử lý tiếp theo như tách từ và loại bỏ từ dừng. Điều này giúp cải thiện chất lượng của dữ liệu và làm cho quá trình xây dựng mô hình phân loại văn bản trở nên hiệu quả hơn.

# loại bỏ stopword cho văn bản

def get\_tokens(text):

    # tách từ

    tokens = word\_tokenize(text)

    # lọc từ đã tách và loại bỏ stopword

    tokens = [t for t in tokens if not t in stop\_words]

    return tokens

Đoạn mã trên định nghĩa một hàm `get\_tokens()` để thực hiện hai công việc chính: tách từ và loại bỏ các từ dừng (stopwords) từ văn bản đầu vào. Quá trình tiền xử lý này là một phần quan trọng trong quá trình xây dựng mô hình phân loại văn bản, nhằm làm sạch và chuẩn bị dữ liệu để có thể được sử dụng hiệu quả trong việc huấn luyện mô hình.

Trong chi tiết:

Hàm sử dụng `word\_tokenize()` từ thư viện `nltk` để tách văn bản thành các từ hoặc token riêng biệt, tạo ra một danh sách các token.

Sau đó, hàm sử dụng một biểu thức danh sách (list comprehension) để lọc và loại bỏ các từ dừng từ danh sách các token đã tách ra từ văn bản. Việc này giúp loại bỏ các từ không mang ý nghĩa và thường không cần thiết trong việc phân loại văn bản.

Kết quả, hàm trả về danh sách các token đã được làm sạch và loại bỏ các từ dừng, sẵn sàng cho các bước tiền xử lý tiếp theo như vectơ hóa hoặc huấn luyện mô hình.

Hàm get\_tokens() là một bước quan trọng trong quy trình tiền xử lý dữ liệu văn bản, đảm bảo rằng dữ liệu được chuẩn bị một cách chính xác và hiệu quả cho việc xây dựng mô hình phân loại văn bản.

def print\_frequency\_dist(docs):

    tokens = defaultdict(list)

    # tạo một danh sách khổng lồ gồm tất cả các từ cho mỗi danh mục

    for doc in docs:

        doc\_label = doc[0]

        #doc\_text = doc[1] sau khi đã tìn ra các từ xuất hiện nhiều nhất, đến bước clean text #1

        doc\_text = clean\_text(doc[1]) # clean text, xóa bỏ các dấu câu #2

        #doc\_tokens = word\_tokenize(doc\_text) #3

        doc\_tokens =get\_tokens(doc\_text) #4

        tokens[doc\_label].extend(doc\_tokens)

    for category\_label, category\_tokens in tokens.items():

        print(category\_label)

        fd = FreqDist(category\_tokens)

        print(fd.most\_common(20))

Đoạn mã trên định nghĩa một hàm `print\_frequency\_dist(docs)` để thực hiện việc in ra tần suất xuất hiện của các từ trong từ điển của mỗi danh mục văn bản. Quá trình này giúp chúng ta hiểu được phân phối từ vựng trong từng danh mục văn bản và có cái nhìn tổng quan về các từ phổ biến trong mỗi danh mục. Dưới đây là mô tả chi tiết về cách hàm này hoạt động:

Khởi tạo từ điển và xây dựng danh sách từ cho mỗi danh mục:

Đầu tiên, chúng ta khởi tạo một từ điển mặc định (defaultdict) với các giá trị ban đầu là danh sách rỗng. Trong từ điển này, mỗi danh mục văn bản sẽ có một danh sách các từ xuất hiện trong các văn bản thuộc danh mục đó.

Tiếp theo, với mỗi văn bản trong danh sách `docs`, chúng ta lấy nhãn của văn bản (danh mục văn bản) và làm sạch văn bản bằng cách loại bỏ dấu câu và chuẩn hóa chữ viết. Sau đó, chúng ta tách văn bản đã làm sạch thành các từ và thêm chúng vào danh sách từ tương ứng với nhãn của văn bản trong từ điển.

In ra tần suất xuất hiện của từng từ trong từ điển:

Duyệt qua từng cặp nhãn và danh sách từ trong từ điển.

Sử dụng `FreqDist` từ thư viện `nltk` để tính toán tần suất xuất hiện của mỗi từ trong danh sách từ.

In ra 20 từ phổ biến nhất và tần suất xuất hiện của chúng trong danh mục văn bản tương ứng.

Hàm `print\_frequency\_dist()` là một phần quan trọng trong quá trình phân tích dữ liệu văn bản, giúp chúng ta hiểu được cấu trúc và nội dung của từng danh mục văn bản. Điều này cung cấp thông tin hữu ích cho quá trình tiền xử lý và xây dựng mô hình phân loại văn bản.

# chia tập dữ liệu thành tập huấn luyện và tập kiểm tra

def get\_splits(docs):

    # trộn dữ liệu trong danh sách docs đảm bảo tính ngẫu nhiên

    random.shuffle(docs)

    # Khởi tạo các danh sách rỗng để lưu trữ các bộ dữ liệu huấn luyện và kiểm tra cùng với nhãn tương ứng

    X\_train = []# training documents

    y\_train = []# corresponding training labels

    X\_test = []# test documents

    y\_test = []# corresponding test labels

    pivot = int(.80 \* len(docs)) # tính đoán index của điểm chia tách giữa tập huấn luyện và tập kiểm tra (80% train và 20% test)

    for i in range(0, pivot):

        X\_train.append(docs[i][1]) # Lặp qua các bộ dữ liệu từ đầu đến điểm chia tách (80%), thêm dữ liệu và nhãn tương ứng vào tập huấn luyện

        y\_train.append(docs[i][0])

    for i in range(pivot, len(docs)):

        X\_test.append(docs[i][1]) # Lặp qua các bộ dữ liệu từ điểm chia tách đến cuối danh sách (20%), thêm dữ liệu và nhãn tương ứng vào tập kiểm tra

        y\_test.append(docs[i][0])

    return X\_train, X\_test, y\_train, y\_test

Đoạn mã trên định nghĩa một hàm `get\_splits(docs)` để chia tập dữ liệu thành tập huấn luyện và tập kiểm tra, sao cho 80% dữ liệu được sử dụng cho việc huấn luyện và 20% dữ liệu được sử dụng cho việc kiểm tra. Dưới đây là mô tả chi tiết về cách hàm này hoạt động:

Trộn dữ liệu và khởi tạo danh sách huấn luyện và kiểm tra:

Đầu tiên, chúng ta sử dụng hàm `random.shuffle()` để trộn ngẫu nhiên danh sách dữ liệu `docs`, đảm bảo tính ngẫu nhiên khi chia tập dữ liệu.

Sau đó, chúng ta khởi tạo các danh sách rỗng để lưu trữ dữ liệu và nhãn tương ứng cho tập huấn luyện và tập kiểm tra.

Chia tập dữ liệu:

Sử dụng chỉ mục `pivot` để xác định điểm chia tách giữa tập huấn luyện và tập kiểm tra, với 80% dữ liệu được sử dụng cho việc huấn luyện.

Duyệt qua danh sách dữ liệu `docs` và thêm dữ liệu và nhãn tương ứng vào tập huấn luyện cho các chỉ mục từ 0 đến `pivot - 1`.

Thêm dữ liệu và nhãn tương ứng vào tập kiểm tra cho các chỉ mục từ `pivot` đến cuối danh sách.

Trả về tập huấn luyện và tập kiểm tra:

Trả về các danh sách `X\_train`, `X\_test`, `y\_train`, `y\_test` chứa dữ liệu và nhãn tương ứng của tập huấn luyện và tập kiểm tra.

Hàm `get\_splits()` là một bước quan trọng trong quy trình chuẩn bị dữ liệu cho việc huấn luyện và kiểm tra mô hình phân loại văn bản. Điều này giúp chúng ta có thể đánh giá hiệu suất của mô hình trên tập dữ liệu độc lập để đảm bảo tính khách quan và chính xác của mô hình.

# đánh giá hiệu suất của một mô hình phân loại trên tập dữ liệu kiểm tra

def evaluate\_classifier(title, classifier, vectorizer, X\_test, y\_test):

    X\_test\_tfidf = vectorizer.transform(X\_test) # Biến đổi tập dữ liệu kiểm tra X\_test thành ma trận TF-IDF (đảm bảo đầu vào cho mô hình có ma trận phù hợp)

    y\_pred = classifier.predict(X\_test\_tfidf) # Sử dụng mô hình phân loại classifier để dự đoán nhãn cho dữ liệu kiểm tra

    precision = metrics.precision\_score(y\_test, y\_pred, average='macro') # Tính toán độ chính xác của mô hình dự đoán

    recall = metrics.recall\_score(y\_test, y\_pred, average='macro') # Tính toán chỉ số Recall của mô hình

    f1 = metrics.f1\_score(y\_test, y\_pred, average='macro') # Tính toán chỉ số F1-score của mô hình dự đoán

    print("%s\t%f\t%f\t%f\n" % (title, precision, recall, f1)) # In ra tiêu đề của phân loại, cùng với các chỉ số đánh giá Precision, Recall và F1-score

Đoạn mã trên định nghĩa một hàm `evaluate\_classifier(title, classifier, vectorizer, X\_test, y\_test)` để đánh giá hiệu suất của một mô hình phân loại trên tập dữ liệu kiểm tra. Dưới đây là mô tả chi tiết về cách hàm này hoạt động:

Biến đổi dữ liệu kiểm tra:

Hàm này sử dụng vectorizer đã được huấn luyện để biến đổi tập dữ liệu kiểm tra `X\_test` thành ma trận TF-IDF. Điều này đảm bảo rằng đầu vào cho mô hình có cùng định dạng với dữ liệu mà mô hình đã được huấn luyện trên.

Dự đoán nhãn cho dữ liệu kiểm tra:

Sau khi có ma trận TF-IDF của dữ liệu kiểm tra, hàm sử dụng mô hình phân loại đã được huấn luyện `classifier` để dự đoán nhãn cho từng mẫu dữ liệu trong tập kiểm tra.

Đánh giá hiệu suất của mô hình:

Hàm tính toán các chỉ số đánh giá bao gồm độ chính xác (precision), chỉ số recall và F1-score bằng cách so sánh nhãn dự đoán với nhãn thực tế trên tập dữ liệu kiểm tra.

Các chỉ số này được tính toán thông qua hàm `precision\_score`, `recall\_score`, và `f1\_score` từ thư viện `sklearn.metrics`.

In ra kết quả:

Cuối cùng, hàm in ra tiêu đề của phân loại cùng với các chỉ số đánh giá precision, recall và F1-score. Điều này giúp chúng ta đánh giá hiệu suất của mô hình trên tập dữ liệu kiểm tra.

Hàm `evaluate\_classifier()` là một công cụ quan trọng để đánh giá hiệu suất của mô hình phân loại văn bản trên tập dữ liệu kiểm tra và đưa ra thông tin về độ chính xác, độ phủ và F1-score của mô hình. Điều này giúp chúng ta có cái nhìn tổng quan về khả năng phân loại của mô hình trên dữ liệu mới mà nó chưa từng nhìn thấy.

# huấn luyện một mô hình phân loại trên tập dữ liệu được cung cấp và lưu trữ mô hình và vectorizer để sử dụng cho dữ liệu mới

def train\_classifier(docs):

    X\_train, X\_test, y\_train, y\_test = get\_splits(docs) # chia tập dữ liệu thành tập huấn luyện và tập kiểm tra

    # Khởi tạo một đối tượng CountVectorizer để biến đổi các văn bản thành các vectơ đặc trưng

    vectorizer = CountVectorizer(stop\_words='english', ngram\_range=(1, 3), min\_df=3, analyzer='word')

    # tạo ma trận văn bản đặc trưng từ dữ liệu huấn luyện

    dtm = vectorizer.fit\_transform(X\_train)

    # Khởi tạo và huấn luyện một mô hình phân loại Naive Bayes

    naive\_bayes\_classifier = MultinomialNB().fit(dtm, y\_train)

    # Đánh giá hiệu suất của mô hình trên tập huấn luyện và in ra kết quả

    evaluate\_classifier("Naive Bayes\tTRAIN\t", naive\_bayes\_classifier, vectorizer, X\_train, y\_train)

    # Đánh giá hiệu suất của mô hình trên tập kiểm tra và in ra kết quả

    evaluate\_classifier("Naive Bayes\tTEST\t", naive\_bayes\_classifier, vectorizer, X\_test, y\_test)

    # Đặt tên cho tệp lưu trữ mô hình phân loại và lưu trữ mô hình phân loại đã được huấn luyện vào tệp

    clf\_filename = 'naive\_bayes\_classifier.pkl'

    pickle.dump(naive\_bayes\_classifier, open(clf\_filename, 'wb'))

    # Đặt tên cho tệp lưu trữ vectorizer và lưu

    vec\_filename = 'count\_vectorizer.pkl'

    pickle.dump(vectorizer, open(vec\_filename, 'wb'))

Đoạn mã trên định nghĩa một hàm `train\_classifier(docs)` để huấn luyện một mô hình phân loại trên tập dữ liệu được cung cấp và lưu trữ mô hình và vectorizer để sử dụng cho dữ liệu mới. Dưới đây là mô tả chi tiết về cách hàm này hoạt động:

Chia tập dữ liệu:

Hàm này sử dụng hàm `get\_splits()` để chia tập dữ liệu `docs` thành tập huấn luyện và tập kiểm tra.

Biến đổi dữ liệu và huấn luyện mô hình:

Sử dụng `CountVectorizer` để biến đổi các văn bản thành các vectơ đặc trưng.

Tạo ma trận văn bản đặc trưng từ dữ liệu huấn luyện bằng cách sử dụng phương thức `fit\_transform()` của vectorizer.

Khởi tạo và huấn luyện một mô hình phân loại Naive Bayes trên ma trận văn bản đặc trưng.

Đánh giá hiệu suất mô hình:

Sử dụng hàm `evaluate\_classifier()` để đánh giá hiệu suất của mô hình trên cả tập huấn luyện và tập kiểm tra. Kết quả được in ra để đánh giá độ chính xác, độ phủ và F1-score của mô hình.

Lưu trữ mô hình và vectorizer:

Sử dụng `pickle.dump()` để lưu trữ mô hình phân loại đã được huấn luyện vào một tệp.

Lưu trữ vectorizer cũng được thực hiện tương tự.

Hàm `train\_classifier()` là một bước quan trọng trong quá trình xây dựng và triển khai mô hình phân loại văn bản. Điều này giúp chúng ta huấn luyện mô hình trên dữ liệu có sẵn và lưu trữ mô hình và vectorizer để có thể sử dụng lại trong tương lai cho việc phân loại dữ liệu mới.

# phân loại một đoạn văn bản mới vào một trong các nhãn của mô hình phân loại đã được huấn luyện

def classify(text):

    # load mô hình phân loại

    clf\_filename = 'D:\\DA2\\Computer\_Science\_Project2\\naive\_bayes\_classifier.pkl'

    nb\_clf = pickle.load(open(clf\_filename, 'rb'))

    # định nghĩa biến là đường dẫn tới tệp chứa vectorizer đã được lưu trữ sau đó mở và load vectorizer từ tệp đã lưu trữ gán cho biến vectorizer

    vec\_filename = 'D:\\DA2\\Computer\_Science\_Project2\\count\_vectorizer.pkl'

    vectorizer = pickle.load(open(vec\_filename, 'rb'))

    # Sử dụng vectorizer để chuyển đổi đoạn văn bản mới thành vectơ đặc trưng, sau đó sử dụng mô hình phân loại để dự đoán nhãn của đoạn văn bản này

    pred = nb\_clf.predict(vectorizer.transform([text]))

    print(f"\n\nThe topic classified for the text is: {str(pred[0]).upper()}\n\n")

Đoạn mã trên định nghĩa một hàm `classify(text)` để phân loại một đoạn văn bản mới vào một trong các nhãn của mô hình phân loại đã được huấn luyện. Dưới đây là mô tả chi tiết về cách hàm này hoạt động:

Load mô hình và vectorizer:

Hàm này sử dụng `pickle.load()` để load mô hình phân loại đã được lưu trữ từ tệp và gán cho biến `nb\_clf`.

Tương tự, nó cũng load vectorizer từ tệp và gán cho biến `vectorizer`.

Chuyển đổi đoạn văn bản mới thành vectơ đặc trưng và dự đoán nhãn:

Sử dụng vectorizer để chuyển đổi đoạn văn bản mới thành vectơ đặc trưng, bằng cách sử dụng phương thức `transform()` của vectorizer.

Sau đó, sử dụng mô hình phân loại để dự đoán nhãn của đoạn văn bản này bằng cách sử dụng phương thức `predict()`.

In ra kết quả phân loại:

Kết quả phân loại được in ra màn hình với chuỗi "The topic classified for the text is:" cùng với nhãn được dự đoán cho đoạn văn bản.

Hàm `classify()` giúp chúng ta áp dụng mô hình phân loại đã huấn luyện để dự đoán nhãn của một đoạn văn bản mới và in ra kết quả phân loại. Điều này rất hữu ích cho việc phân loại dữ liệu mới mà mô hình chưa từng gặp phải.

if \_\_name\_\_ == '\_\_main\_\_':

    #create\_data\_set()

    #docs = setup\_docs()

    #print\_frequency\_dist(docs)

    #train\_classifier(docs)

    new\_doc = """This is an unit-linked fund that gives customers the right to choose investment businesses oriented towards sustainable development. The difference of "PRUlink Green Future" comes from the asset and portfolio selection criteria. The fund's three main investment areas include climate, health, and progress. """

    classify(new\_doc)

    print("Done")

Bạn đã sử dụng câu lệnh `if \_\_name\_\_ == '\_\_main\_\_':` để chỉ chạy các hàm khi tập lệnh được chạy trực tiếp, và không được thực thi khi tập lệnh được import vào một module khác. Trong trường hợp này, bạn đã bỏ comment từng hàm một để thực thi các bước trong quy trình, từ việc tạo tập dữ liệu, phân tích tần suất, huấn luyện mô hình và cuối cùng là phân loại một văn bản mới và in ra kết quả phân loại.

Nếu bạn chạy đoạn mã này, nó sẽ thực hiện phân loại cho đoạn văn bản mới `new\_doc` mà bạn đã cung cấp và in ra kết quả. Cuối cùng, nó sẽ in ra thông báo "Done" để chỉ ra rằng quá trình đã hoàn thành.