Định nghĩa cơ bản của Machine Learning, nơi máy học các quy tắc (pattern) từ dữ liệu để đưa ra dự đoán

Học có giám sát: Máy học từ dữ liệu được cung cấp có nhãn.

Học không giám sát: Máy học từ dữ liệu được cung cấp không có nhãn.

Học tăng cường: Một tác nhân tương tác với môi trường bằng cách thực hiện các hành động và học hỏi từ các lỗi và phần thưởng.

Mục tiêu của học có giám sát là so sánh kết quả đầu ra của mô hình với giá trị thực tế, từ đó điều chỉnh các tham số

Để đánh giá hiệu quả của mô hình, chúng ta chia toàn bộ tập dữ liệu thành ba phần

Tập dữ liệu huấn luyện: sử dụng để huấn luyện mô hình

Tập dữ liệu kiểm chứng: sử dụng để kiểm tra hiệu suất của mô hình sau quá trình huấn luyện

Tập dữ liệu kiểm thử: đánh giá hiệu suất cuối cùng

**K-Nearest Neighbors (KNN):** phân loại một điểm dữ liệu dựa trên các láng giềng gần nhất của nó

**Linear Regression**: một thuật toán học có giám sát được sử dụng để dự đoán các giá trị liên tục

**Logistic Regression**: được dùng cho các bài toán phân loại nhị phân, dùng hàm Sigmoid để dự đoán xác suất, Hàm chi phí cross-entropy

**Naïve Bayes**, đây là một thuật toán phân loại dựa trên Định lý Bayes, nổi bật với giả định rằng tất cả các đặc trưng đầu vào đều độc lập với nhau.

**Support Vector Machine (SVM)**, một thuật toán mạnh mẽ được sử dụng cho cả phân loại và hồi quy. Trong phân loại, mục tiêu của SVM là tìm ra một ranh giới quyết định tối ưu, gọi là **hyperplane**, có khoảng cách lớn nhất đến các điểm dữ liệu gần nhất của mỗi lớp (được gọi là "support vectors").

Neural Network

**cấu trúc cơ bản của NN**, bao gồm các lớp đầu vào, lớp ẩn, lớp đầu ra, cùng với vai trò của các node, trọng số (W) và bias (b)

**quá trình hoạt động và huấn luyện của NN**

quá trình **Feedforward**, nơi dữ liệu được truyền qua các lớp để tạo ra dự đoán.

để tối ưu hóa mô hình, chúng ta sử dụng **Gradient Descent** – một thuật toán tìm cực tiểu của hàm chi phí

**Backpropagation (lan truyền ngược)**, thuật toán cốt lõi giúp tính toán gradient hiệu quả để cập nhật trọng số và bias trong mạng.

**hiện tượng Overfitting** và các phương pháp hiệu quả để khắc phục nó

Overfitting xảy ra khi mô hình học quá chi tiết từ dữ liệu huấn luyện, dẫn đến việc mất khả năng tổng quát hóa tốt trên dữ liệu mới.

Dấu hiệu dễ nhận biết là khi độ chính xác huấn luyện rất cao nhưng độ chính xác trên dữ liệu kiểm tra lại thấp.

Để giải quyết tình trạng này, một trong những cách tiếp cận đầu tiên là **tăng kích thước dữ liệu huấn luyện**

L2 Regularization (hay Weight Decay) và L1 Regularization

L2 Regularization giúp ngăn chặn các trọng số trở nên quá lớn bằng cách thêm tổng bình phương trọng số vào hàm chi phí, từ đó giảm độ phức tạp và tăng khả năng tổng quát hóa của mô hình.

Trong khi đó, L1 Regularization cũng thêm một đại lượng vào hàm chi phí, nhưng là tổng giá trị tuyệt đối của các trọng số, đặc biệt hiệu quả trong việc đẩy các trọng số nhỏ về 0.

**Hiện tượng vanishing gradient** là khi gradient (đạo hàm) của hàm lỗi trở nên quá nhỏ trong quá trình lan truyền ngược qua các lớp của mạng nơ-ron sâu, khiến các trọng số ở các lớp đầu mạng cập nhật rất chậm hoặc ngừng học hoàn toàn.

Các Hàm kích hoạt phổ biến

Sigmoid:

Đầu ra (0, 1), cho phân loại nhị phân.

Nhược điểm: Vanishing Gradient.

Tanh:

Đầu ra (-1, 1), zero-centered, gradient mạnh hơn Sigmoid.

Vẫn gặp Vanishing Gradient.

ReLU (Rectified Linear Unit):

f(x)=max(0,x).

Ưu điểm: Khắc phục Vanishing Gradient, hiệu quả tính toán.

Nhược điểm: Dying ReLU.

Leaky ReLU:

Khắc phục Dying ReLU bằng gradient nhỏ khi x<0.

Parametric ReLU (PReLU):

Hệ số cho phần âm được học tự động.

ELU (Exponential Linear Unit):

Sử dụng hàm mũ cho giá trị âm, khắc phục Dying ReLU.

Swish:

f(x)=x⋅σ(x), hiệu suất tốt trên mô hình sâu.

Liên tục, khả vi, không đơn điệu.

Softmax:

Chuyển vector đầu vào thành phân phối xác suất cho phân loại đa lớp.

Đầu ra (0, 1) và tổng bằng 1.

 **Giới thiệu Tiền xử lý dữ liệu**

* Quan trọng để cải thiện chất lượng và hiệu quả phân tích dữ liệu.
* Xử lý dữ liệu thiếu, nhiễu, không nhất quán, dư thừa.

 **Các yếu tố ảnh hưởng đến chất lượng dữ liệu**

* Độ chính xác (Accuracy)
* Độ đầy đủ (Completeness)
* Tính nhất quán (Consistency)
* Tính kịp thời (Timeliness)
* Độ tin cậy (Believability)
* Khả năng diễn giải (Interpretability)

 **Các kỹ thuật tiền xử lý dữ liệu chính**

* **Làm sạch dữ liệu (Data Cleaning)**: Điền giá trị thiếu, làm mịn dữ liệu nhiễu, loại bỏ ngoại lệ, giải quyết không nhất quán.
* **Tích hợp dữ liệu (Data Integration)**: Kết hợp dữ liệu từ nhiều nguồn, xử lý khác biệt tên/giá trị thuộc tính, tránh dư thừa.
* **Giảm kích thước dữ liệu (Data Reduction)**: Giảm dung lượng mà không mất thông tin quan trọng.
  + Giảm chiều dữ liệu (Dimensionality Reduction).
  + Giảm số lượng dữ liệu (Numerosity Reduction).
* **Biến đổi dữ liệu (Data Transformation)**: Thay đổi/tái cấu trúc dữ liệu cho phù hợp thuật toán.
  + Chuẩn hóa dữ liệu (Normalization).
  + Phân biệt hóa dữ liệu (Discretization) và Tạo cấp độ khái niệm (Concept Hierarchy Generation).

Transformer và cơ chế Attention

**cơ chế Attention**, một phương pháp chọn lọc giúp mô hình tập trung vào những yếu tố quan trọng trong dữ liệu, giống như cách bộ não con người chọn lọc thông tin.

Tiếp theo, tôi sẽ trình bày về **Transformer**, một kiến trúc được thiết kế dựa trên mô hình Encoder-Decoder, kết hợp những ưu điểm của cả Mạng nơ-ron tích chập (CNN) và Mạng nơ-ron hồi tiếp (RNN) để xử lý hiệu quả thông tin chuỗi.

Cấu trúc Encoder:

Cấu trúc Decode:

ưới đây là tóm tắt chức năng của từng thành phần chính trong kiến trúc Transformer:

* **Word Embedding**: Chuyển đổi mỗi từ trong câu thành một vector có kích thước cố định, mang thông tin ngữ nghĩa của từ đó.
* **Positional Encoding**: Cung cấp thông tin về vị trí tương đối và tuyệt đối của từ trong câu, vì Transformer không có tính tuần tự như RNN. Thường sử dụng các hàm sin và cos để mã hóa vị trí.
* **Self-Attention**: Cho phép mỗi từ trong chuỗi đầu vào "chú ý" đến các từ khác trong cùng một chuỗi để hiểu ngữ cảnh. Nó tính toán mức độ liên quan giữa các từ thông qua Query, Key và Value vectors.
* **Multi-Head Attention**: Thay vì chỉ có một Self-Attention, Transformer sử dụng nhiều "head" chạy song song. Mỗi head học các biểu diễn khác nhau của thông tin, giúp mô hình nắm bắt được nhiều khía cạnh phụ thuộc trong chuỗi và tăng khả năng mô hình hóa các phụ thuộc dài hạn.
* **Masked Multi-Head Attention**: Một biến thể của Multi-Head Attention được sử dụng trong Decoder. Nó đảm bảo rằng khi dự đoán một từ, mô hình chỉ "nhìn thấy" các từ đã được tạo ra trước đó trong chuỗi đầu ra, ngăn chặn việc "nhìn trước" các từ tương lai.
* **Encoder-Decoder Multi-Head Attention**: Nằm trong Decoder, thành phần này cho phép Decoder chú ý đến thông tin từ đầu ra của Encoder. Query đến từ Decoder, trong khi Key và Value đến từ Encoder, giúp Decoder tham chiếu trực tiếp đến các thông tin quan trọng từ chuỗi đầu vào.
* **Residuals (Skip Connections) và Layer Normalization**:
  + **Residuals**: Giúp truyền thông tin trực tiếp qua các lớp, ngăn chặn vấn đề vanishing gradient và giúp huấn luyện các mạng sâu dễ dàng hơn.
  + **Layer Normalization**: Chuẩn hóa đầu ra của mỗi lớp để có trung bình bằng 0 và phương sai bằng 1, giúp ổn định quá trình huấn luyện và giảm hiện tượng lệch hiệp phương sai (covariance shift).
* **Positionwise Feed-forward Networks (FFN)**: Một mạng nơ-ron truyền thẳng (fully connected) được áp dụng độc lập tại mỗi vị trí trong chuỗi. Nó giúp mô hình xử lý và biến đổi thông tin đã được tổng hợp từ lớp attention.
* **Encoder**: Khối này nhận chuỗi đầu vào và biến đổi nó thành một chuỗi các biểu diễn ngữ cảnh. Mỗi lớp Encoder bao gồm Multi-Head Attention, Residuals, Layer Normalization và FFN. Nó học mối quan hệ nội tại giữa các từ trong chuỗi đầu vào.
* **Decoder**: Khối này nhận biểu diễn ngữ cảnh từ Encoder và tạo ra chuỗi đầu ra. Mỗi lớp Decoder bao gồm Masked Multi-Head Attention, Encoder-Decoder Attention, Residuals, Layer Normalization và FFN. Nó học cách tạo ra chuỗi đầu ra từng bước, đồng thời tham chiếu đến thông tin từ chuỗi đầu vào.

**Federated Learning (FL)**, một phương pháp học máy tiên tiến ra đời để khắc phục những hạn chế của cách học truyền thống, đặc biệt là vấn đề bảo mật dữ liệu.

Thay vì yêu cầu các máy khách chia sẻ dữ liệu thô, FL cho phép họ giữ lại dữ liệu của mình và chỉ gửi các mô hình đã huấn luyện cục bộ lên máy chủ trung tâm.

Máy chủ sẽ tổng hợp các mô hình này thành một mô hình chung, sau đó phân phối lại cho các máy khách để tiếp tục cải thiện.

Chúng ta sẽ đi sâu vào ba loại chính của FL dựa trên cách lựa chọn mẫu dữ liệu:

**Horizontal Federated Learning**, nơi các máy khách có dữ liệu khác nhau nhưng cùng đặc trưng;

**Vertical Federated Learning**, nơi các máy khách có cùng tập mẫu nhưng đặc trưng khác nhau; và

**Federated Transfer Learning**, áp dụng khi cả mẫu và đặc trưng đều khác nhau.

ba kỹ thuật tổng hợp mô hình quan trọng:

**All Model Averaging (AMA)**, nơi trung bình hóa tất cả các mô hình;

**One Model Selection (OMS)**, lựa chọn mô hình tốt nhất duy nhất; và

**Best Model Average (BMA)**, tính trung bình của một tập hợp các mô hình tốt nhất.