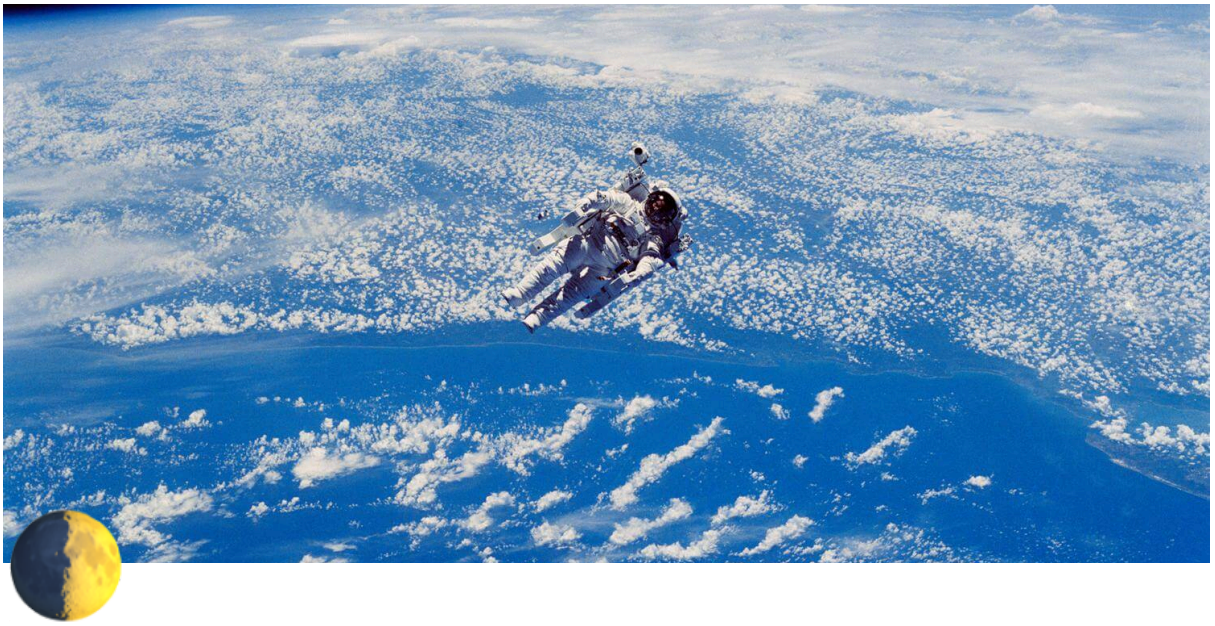

인공지능 개론

인공지능에서 지능에 해당하는 기능은 무엇인가?

인공지능에서 **지능**은 주어진 정보를 바탕으로 학습하고, 추론하며, 문제를 해결하는 능력을 의미한다. **대표적인 기능으로는 인식(시각, 음성), 자연어 처리, 학습, 추론, 계획 수립** 등이 있다. 이 기능들을 통해 인공지능은 인간처럼 판단하고 행동할 수 있게 된다.

인공지능의 종류 3가지에 대해서 설명하시오 (지도학습, 반지도학습, 강화학습)

지도학습은 입력과 정답(label)이 모두 주어진 데이터를 바탕으로 모델을 학습시키는 방식으로, 반복 학습을 통해 오류를 줄여 가면서 점차 정답에 가까워지는 방법이다. 분류나 회귀 문제에 사용된다. **반지도학습**은 일부 데이터에만 정답이 있는 경우로, 소량의 정답 데이터와 대향의 비정답 데이터를 함께 사용하여 학습 성능을 높인다. 적은 레이블로도 전체 데이터의 구조를 활용해 학습 성능을 높일 수 있다. **강화학습**은 에이전트(Agent)가 환경과 상호작용하며 보상을 최대화하는 방향으로 스스로 행동 전략을 학습하는 방식으로, 정답을 직접 주지 않고, 행동에 대한 보상을 통해 최적의 전략을 학습한다. 환경에 대한 사전 지식이 없는 상태로 학습이 진행된다.

전통적인 프로그래밍 방법과 인공지능 프로그램의 차이점은 무엇인가?

전통적인 프로그래밍은 개발자가 명확한 규칙과 알고리즘을 직접 코드로 작성하는 방식이다. 반면 **인공지능 프로그래밍**은 AI가 입력된 데이터로부터 규칙을 스스로 학습하여 패턴을 찾아내고 예측을 수행한다. 전통적 방식은 **'입력 → 규칙(알고리즘) → 출력'** 구조이고, 인공지능은 **'입력 + 정답 → 학습 → 모델 생성 → 예측'**의 흐름을 따른다.

딥러닝과 머신러닝의 차이점은 무엇인가?

머신러닝은 데이터를 기반으로 스스로 규칙을 학습하는 인공지능 기술 전반을 의미하며, 사람이 특징을 설계해야 한다. **딥러닝**은 머신러닝의 한 종류로, 인공신경망을 깊게 쌓아 복잡한 데이터에서 특징을 자동으로 추출하고 학습한다. 즉, 머신러닝은 사람이 특징을 정의하고, 딥러닝은 모델이 특징까지 자동으로 학습한다는 점이 가장 큰 차이다. 응용분야는 **머신러닝은 간단한 데이터를 주로 분석하고 딥러닝은 이미지 인식, 음성인식, 자연어 처리**등을 분석한다.

Classification과 Regression의 주된 차이점은?

Classification은 입력 데이터를 정해진 범주(클래스) 중 하나로 분류하는 문제이며, 출력값이 **이산적인** 형태이다. Regression은 입력에 따라 **연속적인** 수치를 예측하는 문제로, 출력값이 **연속적인** 형태이다. 즉, Classification은 범주형 결과, Regression은 수치형 결과를 예측한다는 점이 핵심적인 차이이다.

머신러닝에서 차원의 저주(curse of dimensionality)란?

차원의 저주는 머신러닝에서 피처(변수) 수가 너무 많아질 때 발생하는 문제로, 데이터의 차원이 증가할 수록 데이터가 희소해지고 계산량이 증가하며 모델의 성능이 저하되는 문제가 발생하는 것을 말한다.

차원 축소(Dimensionality Reduction)는 왜 필요한가?

Dimensionality Reduction은 고차원 데이터에서 불필요한 피처를 줄여 학습 성능과 속도를 개선하기 위해 필요하다. 차원이 높을수록 계산량이 늘고, 과적합이나 차원의 저주가 발생할 위험이 커진다. 차원 축소를 하면 중요한 정보만 유지하면서 복잡도를 낮추면 모델이 더 일반화되기 쉽고, 시각화도 가능해진다.

Ridge와 Lasso의 공통점과 차이점? (Regularization, 규제, Scaling)

Ridge와 Lasso는 모두 모델의 과적합을 방지하기 위해 사용하는 **규제(Regularization)** 기법이며, 손실 함수에 패널티 항을 추가하는 방식이다. 두 방법 모두 학습 전에 스케일링을 해주는 것이 중요하다. 차이점은 패널티 방식인데, **Ridge**는 가중치의 제곱합(L2)을 사용해 모든 피처를 조금씩 줄이고, **Lasso**는 가중치의 절댓값 합(L1)을 사용해 일부 피처의 계수를 0으로 만들어 변수 선택 효과도 있다.

Overfitting vs. Underfitting

Overfitting은 모델이 훈련 데이터에 너무 과하게 맞춰져서 테스트 데이터에서는 성능이 떨어지는 현상이다. 반대로 **Underfitting**은 모델이 너무 단순해서 훈련 데이터조차 제대로 학습하지 못하는 상태다. Overfitting은 일반화 능력이 부족하고, Underfitting은 학습 자체가 부족하다는 점에서 차이가 있다. 적절한 모델 복잡도와 규제 등을 통해 두 문제를 피하는 것이 중요하다.

Feature Engineering과 Feature Selection의 차이점은?

Feature Engineering은 기존 데이터를 바탕으로 새로운 피처를 생성하거나 변환해 모델 성능을 높이는 과정이다. 반면 **Feature Selection**은 이미 존재하는 피처 중에서 중요한 것만 골라내어 불필요한 변수를 제거하는 과정이다. 전자는 피처를 **만드는 작업**, 후자는 피처를 **고르는 작업**이라는 점에서 차이가 있다. 두 작업 모두 모델의 성능 향상과 과적합 방지에 중요한 역할을 한다.

전처리(Preprocessing)의 목적과 방법? (노이즈, 이상치, 결측치)

전처리의 목적은 모델이 데이터를 효과적으로 학습할 수 있도록 품질을 개선하는 데 있다. **주요 방법에는 노이즈 제거, 이상치 처리,결측치 보완** 등이 있으며, 이들은 데이터 오류로 인한 학습 방해를 줄여준다. 예를 들어 결측치는 평균 대체나 삭제로 처리하고, 이상치는 IQR이나 Z-score로 탐지 후 수정하거나 제거한다. 전처리는 모델 성능과 일반화 능력을 높이는 데 필수적인 단계다.

EDA(Exploratory Data Analysis)란? 데이터의 특성 파악(분포, 상관관계)

EDA는 탐색적 데이터 분석으로, 모델 학습 전에 데이터의 분포, 변수 간 상관관계 등을 시각화와 통계를 통해 파악한다. 이를 통해 데이터의 구조와 특성을 이해하고, 전처리나 모델링 방향을 결정할 수 있다. 예를 들어 데이터 분포 분석을 통해 컬럼이 정규분포를 따르는 지 아니면 치우친 분포를 보이는 지를 확인할 수 있고, 변수 간의 상관관계를 분석하여 두 변수가 서로 영향을 얼마나 받고 있는 지 등을 알 수 있다.

회귀에서 절편과 기울기가 의미하는 바는? 딥러닝과 어떻게 연관되는가?

회귀에서 기울기는 변수 간의 관계의 강도를 나타내고, 절편은 입력이 0일 때의 예측값을 의미한다. 이 둘은 데이터의 경향성을 설명해준다. **딥러닝에서도 각 층의 뉴런은 입력에 가중치(기울기)를 곱하고 편향(절편)을 더한 후 활성화 함수를 적용하므로, 기울기와 절편의 개념이 딥러닝에서 뉴런이 학습하는 가중치와 편향과 동일한 역할을 한다.**

Activation function 함수를 사용하는 이유? Softmax, Sigmoid 함수의 차이는?

Activation 함수는 뉴런이 출력값을 비선형적으로 표현할 수 있게 해줘, 딥러닝 모델이 복잡한 패턴을 학습할 수 있도록 만든다. **Sigmoid 함수**는 출력을 0~1 사이로 변환해 이진 분류에 주로 사용되고, **Softmax**는 여러 클래스에 대한 확률 분포를 반환해 다중 분류에 적합하다. Sigmoid는 각 클래스에 독립적으로 작용하지만, Softmax는 전체 클래스의 상대적인 확률을 동시에 고려한다.

Forward propagation, Backward propagation이란?

Forward propagation은 입력 데이터를 순차적으로 각 층을 통과시키며 예측값을 계산하는 과정이다. 반면 **Backward propagation**은 예측값과 실제값의 오차를 기반으로 손실을 계산하고, 그 오차를 뒤로 전파하여 가중치를 업데이트하는 과정이다. 이 두 과정이 반복되면서 딥러닝 모델은 점점 더 정확한 예측을 하도록 학습된다.

손실함수란 무엇인가? 가장 많이 사용하는 손실함수 4가지 종류는?

손실함수는 모델의 예측값과 실제값의 차이를 수치로 나타내는 함수로, 이 값을 최소화하는 방향으로 모델이 학습된다. 자주 사용하는 손실함수에는 회귀에서 사용되는 **평균제곱오차(MSE), 평균절대오차(MAE), 분류에서 사용하는 교차엔트로피(Cross-Entropy), 힌지 손실(Hinge Loss)** 등이 있다. 문제 유형에 따라 적절한 손실함수를 선택하는 것이 중요하다.

옵티마이저(optimizer)란 무엇일까? 옵티마이저와 손실함수의 차이점은?

옵티마이저는 **손실함수를 최소화하기 위해 가중치와 편향 같은 파라미터를 어떻게 조정할지를 결정하는 알고리즘**이다. **손실함수가 예측과 정답 간의 오차를 수치화하는 역할이라면, 옵티마이저는 그 오차를 줄이기 위한 학습 경로를 찾는 역할을 한다**. 즉, 손실함수는 "얼마나 틀렸는가"를 측정하고, 옵티마이저는 "어떻게 고칠 것인가"를 결정한다. 대표적인 옵티마이저로는 SGD, Adam, RMSprop 등이 있다.

경사하강법 의미는? (확률적 경사하강법, 배치 경사하강법, 미니 배치 경사하강법)

경사하강법은 손실함수를 최소화하기 위해 손실의 기울기(경사)를 따라 가중치를 반복적으로 조정하는 최적화 알고리즘이다. **확률적 경사하강법(SGD)**은 한 개의 샘플만으로 가중치를 자주 업데이트해 계산은 빠르지만 불안정하다. **배치 경사하강법(BGD)**은 전체 데이터를 사용해 안정적으로 학습하지만 느리고 자원이 많이 든다. **미니 배치 경사하강법**은 소규모 데이터 묶음으로 학습해 속도와 안정성의 균형을 잡은 방식이다.

교차검증, K-fold 교차검증의 의미와 차이

교차검증은 모델의 일반화 성능을 평가하기 위해 데이터를 여러 번 나눠 훈련과 검증을 반복하는 기법이다. 그중 **K-fold 교차검증**은 데이터를 K개의 동일한 크기로 나누고, 그중 하나를 검증용으로, 나머지를 훈련용으로 사용해 K번 반복한다. 일반적인 교차검증은 단순히 데이터를 훈련/검증으로 한 번 나누는 경우도 포함되지만, K-fold는 이 과정을 여러 번 반복해 더 신뢰도 높은 평균 성능을 얻는다.

하이퍼파라미터 튜닝이란 무엇인가?

하이퍼파라미터 튜닝은 학습 전에 사람이 설정해야 하는 하이퍼파라미터(예: 학습률, 배치 크기, 은닉층 수 등)의 최적 조합을 찾는 과정이다. 모델의 성능은 이 값들에 크게 영향을 받기 때문에, 실험적으로 여러 조합을 시도해 가장 좋은 결과를 내는 설정을 찾아야 한다. 대표적인 방법으로는 Grid Search, Random Search, 베이지안 최적화 등이 있다.

CNN의 합성곱의 역할은?

CNN에서 합성곱(Convolution)은 입력 이미지의 특징을 추출하는 역할을 한다. 작은 필터(커널)를 이용해 이미지 전체를 슬라이딩하면서 엣지, 윤곽, 패턴 등의 시각적 특징을 감지한다. 이 과정을 통해 원본 데이터를 압축하면서도 의미 있는 정보만 남기는 것이 가능해진다. 결과적으로 합성곱은 이미지에서 중요한 공간적 정보를 효과적으로 학습하는 핵심 단계이다.

CNN의 풀링층의 역할은?

CNN에서 풀링층(Pooling layer)은 특징맵의 크기를 줄여 계산량을 줄이고, 중요한 정보만 남기는 역할을 한다. 주로 Max Pooling이 사용되며, 일정 영역에서 가장 큰 값을 선택해 공간 정보를 요약한다. 이를 통해 모델은 위치 변화에 덜 민감해지고 과적합도 줄어든다. 즉, 풀링층은 데이터 압축과 불변성 확보를 동시에 수행한다.

CNN의 Dense Layer의 역할은?

CNN의 Dense Layer(완전 연결층)는 합성곱과 풀링을 통해 추출된 특징들을 기반으로 최종 예측을 수행하는 역할을 한다. 이 층에서는 모든 뉴런이 서로 연결되어 있으며, 특징들을 종합해 클래스 분류나 수치 예측 등의 출력 결과를 만든다. 이미지 분류 문제에서는 보통 마지막에 Softmax를 적용해 각 클래스에 대한 확률을 출력한다. Dense Layer는 전체 정보를 통합해 최종 판단을 내리는 단계이다.

CNN의 stride, filter의 역할? 필터의 가중치는 어떻게 결정되는가?

CNN에서 stride는 필터가 입력 위를 이동하는 간격을 의미하며, stride가 크면 출력 크기는 작아지고 계산 속도는 빨라진다. **filter(또는 kernel)**는 입력의 특정 영역을 스캔하면서 특징을 추출하는 역할을 한다. 이때 각 필터는 다른 시각적 특징(예: 엣지, 곡선 등)을 감지하도록 학습된다. 필터의 가중치는 초기에는 랜덤으로 설정되며, 학습 과정 중 **오차 역전파와 경사하강법**을 통해 자동으로 조정된다.

RNN을 사용하는 이유와 한계점은?

RNN은 시계열 데이터나 문장처럼 **순서가 중요한 데이터**를 처리하기 위해 사용된다. 이전 시점의 출력을 다음 시점으로 전달하며 **시간의 흐름에 따른 정보**를 기억할 수 있어 자연어 처리, 음성 인식 등에 적합하다. 그러나 RNN은 시간이 길어질수록 **기울기 소실(gradient vanishing)** 문제가 발생해 **장기 의존성 학습이 어렵다**는 한계가 있다. 이를 보완하기 위해 LSTM, GRU 같은 구조가 개발되었다.

LSTM을 사용하는 이유와 한계점은?

LSTM은 RNN의 장기 의존성 문제를 해결하기 위해 고안된 구조로, 중요한 정보는 오래 기억하고 불필요한 정보는 잊는 **게이트 구조**를 사용한다. 이를 통해 긴 문장이나 긴 시계열 데이터에서도 안정적으로 학습이 가능하다. 그러나 구조가 복잡하고 계산 비용이 커서 훈련 시간이 오래 걸리며, 많은 데이터를 필요로 하는 한계가 있다. GPT 같은 트랜스포머 모델이 최근 LSTM을 대체하는 추세다.

GRU을 사용하는 이유와 차별성은?

GRU는 LSTM보다 간단한 구조로, 비슷한 성능을 유지하면서 계산량을 줄이기 위해 고안된 순환 신경망이다. 업데이트 게이트와 리셋 게이트만 사용해 정보를 조절하며, LSTM보다 파

라미터 수가 적어 훈련 속도가 빠르고 구현이 간단하다. 특히 비교적 적은 데이터나 자원에서도 안정적인 성능을 낼 수 있다는 장점이 있다. LSTM보다 효율적이지만, 아주 복잡한 장기 의존 문제에서는 성능이 조금 떨어질 수 있다.

결정트리에서 불순도(Impurity) – 지니 계수(Gini Index)란 무엇인가?

결정트리에서 **불순도**는 하나의 노드에 여러 클래스가 섞여 있을 때 혼란스러운 정도를 의미한다. **지니 계수**는 이 불순도를 수치로 나타낸 것으로, 값이 0이면 순수(한 클래스만 존재)하고, 0.5에 가까울수록 클래스가 섞여 있다. 계산식은 $1 - \sum(p_i^2)$ 로, p_i는 각 클래스의 비율이다. 결정트리는 지니 계수가 낮아지는 방향으로 데이터를 분할한다.

앙상블이란 무엇인가?

앙상블은 여러 개의 모델을 결합해 하나의 더 강력한 예측 모델을 만드는 기법이다. 각 모델은 개별적으로 약한 성능을 가질 수 있지만, 서로 다른 오류를 보완해 전체 성능을 높일 수 있다. 보팅, 배깅, 부스팅 등의 방법이 있으며, 대표적인 앙상블 모델로 랜덤 포레스트와 그래디언트 부스팅이 있다. 일반적으로 단일 모델보다 예측 정확도와 안정성이 높다.

부트스트랩핑(bootstrapping)이란 무엇인가?

부트스트랩핑은 전체 데이터에서 **중복을 허용하면서 랜덤하게 샘플을 뽑는 통계적 기법**이다. 이렇게 생성된 여러 샘플은 서로 다른 분포 특성을 가질 수 있어, 모델을 다양하게 학습시키는 데 활용된다. 주로 앙상블 학습에서 **모델의 다양성과 안정성을 확보**하기 위해 사용되며, 예측의 신뢰도 평가에도 활용된다.

배깅(Bagging)이란 무엇인가?

배깅(Bagging)은 **부트스트랩핑으로 생성한 여러 데이터셋에 같은 알고리즘을 적용해 여러 개의 모델을 학습시키고**, 그 예측 결과를 평균(회귀)이나 다수결(분류)로 통합하는 앙상블 기법이다. 이를 통해 **분산을 줄이고 과적합을 완화**할 수 있다. 대표적인 예로는 여러 결정트리를 결합한 랜덤 포레스트가 있다.

주성분 분석(PCA) 이란 무엇인가?

주성분 분석(PCA)은 고차원 데이터를 **저차원 공간으로 축소**하면서도 데이터의 **분산(정보량)을 최대한 보존**하는 차원 축소 기법이다. 여러 변수들 간의 상관관계를 분석해, 가장 많은 정보를 담고 있는 새로운 축(주성분)을 만든다. 이를 통해 노이즈를 줄이고 시각화나 학습 효율을 향상시킬 수 있다.

Dense Layer란 무엇인가?

Dense Layer는 인공신경망에서 **모든 뉴런이 이전 층의 모든 출력과 연결된 완전 연결층**을 말한다. 입력된 특징들을 종합하고 가중합 연산 후 활성화 함수를 적용해 다음 출력으로 전달한다. 주로 마지막에 배치되어 **분류나 회귀와 같은 최종 예측을 수행**하는 데 사용된다. 복잡한 관계를 학습하는 데 중요한 역할을 한다.

규제와 스케일링 차이

규제는 모델이 훈련 데이터에 과적합되는 것을 방지하기 위해 손실 함수에 패널티를 추가하는 방식이다. 주로 선형 회귀, 로지스틱 회귀, 신경망 등에서 사용되며, 가중치의 크기를 제한해 모델 복잡도를 낮춘다. 반면 **스케일링**은 데이터 전처리 단계에서 특성값의 범위를 조정하는 작업으로, 거리 기반 모델이나 경사 하강법을 사용하는 모델에서 학습 속도와 정확도를 높이기 위해 사용된다. 규제는 모델 내부에서 작동하고, 스케일링은 학습 이전에 데이터에 적용된다.

훈련 데이터와 테스트 데이터를 나누는 이유

훈련 데이터와 테스트 데이터를 나누는 이유는 **모델이 학습한 내용을 실제 상황에 잘 일반화할 수 있는지 확인**하기 위해서다. 모델은 훈련 데이터를 통해 패턴을 학습하지만, 같은 데이터로 평가하면 진짜 성능을 알 수 없다. 따라서 학습에 사용하지 않은 테스트 데이터를 따로

두고, 여기에 대해 예측을 수행해 성능을 평가한다. 이렇게 해야 과적합 여부를 확인할 수 있고, 새로운 데이터에 대한 예측 능력을 객관적으로 판단할 수 있다.

딥러닝에서 파라미터

딥러닝에서 **파라미터**는 가중치와 편향처럼 학습을 통해 조정되는 값이다. 이들은 입력을 변형해 출력을 만드는 데 사용되며, 손실을 최소화하는 방향으로 반복적으로 업데이트된다. 층이 많아질수록 파라미터 수도 증가하고, 복잡한 패턴을 학습할 수 있지만 과적합 위험도 커진다. 결국 파라미터는 모델이 데이터를 통해 의미 있는 규칙을 학습하도록 만드는 핵심 요소다.

피처(feature)

피처(feature)는 머신러닝이나 딥러닝 모델이 입력으로 사용하는 데이터의 속성이다. 예를 들어 '키', '몸무게', '성별', '연령' 등이 모두 피처가 될 수 있으며, 모델은 이 피처들을 바탕으로 결과값을 예측한다. 피처의 선택과 전처리 방식은 모델 성능에 직접적인 영향을 주기 때문에 중요하다. 불필요한 피처는 노이즈를 증가시키고, 중요한 피처는 예측력을 높인다. 따라서 피처 엔지니어링은 데이터 분석의 핵심 과정 중 하나이다.

불순도

불순도(impurity)는 하나의 노드 안에 여러 클래스가 섞여 있을 때 혼란스러운 정도를 수치로 나타낸 것이다. 결정트리 알고리즘은 이 불순도를 최소화하는 방향으로 노드를 분할한다. 즉, 한 노드에 하나의 클래스만 포함되도록 하는 것이 목표다. 불순도가 낮을수록 분류가 명확하며, 트리의 성능도 좋아진다.

지니 계수

지니 계수(Gini index)는 불순도를 계산하는 방법 중 하나로, 값이 0이면 완전히 순수한 상태(한 클래스만 있음)를 의미한다. 계산식은 $1 - \sum(p_i^2)$로, p_i는 각 클래스의 비율이다. 예를 들어 한 노드에 A: 50%, B: 50%가 있다면 지니 계수는 0.5이고, A: 100%이면 0이다. 결정트리는 이 값을 기준으로 분할을 결정하며, 지니 계수가 낮을수록 더 나은 분할이다.

확률적 경사하강법

확률적 경사하강법(SGD)은 훈련 데이터를 **한 번에 하나씩 선택해** 가중치를 갱신하는 방식이다. 계산 속도가 빠르고 메모리 사용량이 적다는 장점이 있지만, 매번 다른 데이터로 업데이트하므로 경로가 불안정하고 진동이 심할 수 있다. 하지만 잡음을 통해 지역 최솟값을 벗어날 수 있는 장점도 있다. 대규모 데이터셋을 빠르게 학습하는 데 자주 사용된다.

배치 경사하강법

배치 경사하강법(BGD)은 **전체 훈련 데이터를 모두 사용해** 손실 함수의 평균 기울기를 계산한 뒤 가중치를 한 번에 갱신한다. 학습 경로가 안정적이며, 최적값으로 천천히 수렴하는 특성이 있다. 하지만 데이터가 많을 경우 매 반복마다 전체 데이터를 다 확인해야 하므로 계산 비용이 매우 크다. 따라서 소규모 데이터나 계산 자원이 충분할 때 적합하다.

미니배치 경사 하강법

미니 배치 경사하강법(Mini-Batch Gradient Descent)은 전체 데이터를 **일정 크기의 작은 묶음(미니 배치)으로 나눠** 학습하는 방식이다. 전체 데이터(Batch Gradient)보다 빠르고, 한 샘플씩 처리하는 방식(SGD)보다 안정적이다. 속도와 성능의 균형이 좋아서 딥러닝에서 가장 많이 사용된다. 일반적으로 GPU 병렬 처리를 효율적으로 활용할 수 있다는 장점도 있다.

옵티마이저

옵티마이저(Optimizer)는 모델의 손실 함수 값을 최소화하기 위해 가중치와 편향 같은 파라미터를 어떻게 조정할지 결정하는 알고리즘이다. 경사하강법(Gradient Descent)을 기반으로 하며, 대표적으로 **SGD, Adam, RMSprop** 등이 있다. 옵티마이저는 학습 속도, 수렴 안정성, 일반화 성능에 큰 영향을 준다. 적절한 옵티마이저 선택은 딥러닝 성능 향상에 핵심적인 역할을 한다.

경사하강법은 "무엇을 할지(개념)

옵티마이저는 그것을 "어떻게 할지(구현)"를 나타낸다

교차검증

교차검증(cross-validation)은 데이터를 여러 개의 조각으로 나누고, 그 중 일부는 학습에, 나머지는 검증에 사용하는 과정을 반복해 평균 성능을 평가하는 방법이다. 대표적인 방식은 K-Fold이며, 데이터셋을 K등분해서 K번 학습과 평가를 반복한다. 이렇게 하면 데이터가 적을 때도 모델의 일반화 성능을 안정적으로 평가할 수 있다. 과적합 여부를 확인하는 데 유용하다.

하이퍼파라미터

하이퍼파라미터(hyperparameter)는 모델 학습 전에 사용자가 직접 설정하는 값으로, 학습률, 반복 횟수, 은닉층 수 등이 이에 해당한다. 이 값들은 학습 과정에서 자동으로 조정되지 않기 때문에 경험적 실험이나 교차검증을 통해 최적값을 찾아야 한다. 잘못 설정할 경우 과적합이나 학습 실패가 발생할 수 있다. 따라서 모델의 성능에 미치는 영향이 매우 크다.

앙상블

앙상블(ensemble)은 여러 개의 약한 모델을 결합해 하나의 강한 모델을 만드는 기법이다. 개별 모델이 약하더라도 서로 다른 관점에서 학습된 예측 결과를 종합하면 성능이 높아진다. 주로 분류에서는 다수결, 회귀에서는 평균을 통해 최종 결과를 도출한다. 일반적으로 단일 모델보다 높은 정확도와 안정성을 보인다.

랜덤포레스트

랜덤포레스트(Random Forest)는 대표적인 앙상블 모델로, 여러 개의 결정트리를 만들고 이들의 예측을 종합하여 최종 결과를 도출한다. 각각의 트리는 서로 다른 데이터 샘플과 서로 다른 피처를 사용하여 학습된다. 이처럼 다양성을 가진 모델들을 결합하면, 하나의 트리보다 과적합이 줄고 예측의 신뢰도가 높아진다. 따라서 랜덤포레스트는 앙상블 기법 중에서도 성능과 해석력의 균형이 잘 잡힌 모델로 자주 활용된다.

주도학습 비지도학습

지도 학습(Supervised Learning)은 입력 데이터에 대해 사람이 미리 지정한 정답(레이블)을 함께 제공하는 방식의 학습이다. 예를 들어, 고양이와 개 이미지에 각각 '고양이', '개'라는 정답이 붙어 있을 때, 모델은 이 정답에 최대한 가까운 출력을 내도록 학습한다. 이 방식은 **정답이 명확히 주어진 데이터셋이 충분히 확보되어 있는 환경에서 사용**된다. 학습 이후에는 새로운 입력에 대해 정답을 예측하는 데 활용된다.

반면에 **비지도 학습(Unsupervised Learning)**은 정답(레이블)이 제공되지 않은 데이터를 기반으로 학습하는 방식이다. 예를 들어, 고객 행동 로그 데이터나 산업 현장의 센서 로그처럼 정답 라벨이 존재하지 않거나 수집이 어려운 경우에 사용된다. 특히, **정상 데이터만 수집이 가능하고, 비정상 데이터가 거의 존재하지 않는 환경**에서는 정상 데이터만으로 학습을 수행한 뒤, **테스트 단계에서 정상과 비정상 데이터를 함께 투입하여 이상 탐지를 수행**할 수 있다.

이는 라벨이 없는 상태에서도 모델이 데이터를 분류하거나 구조를 파악할 수 있도록 하며, **불균형한 데이터 구성 문제를 회피하는 데에도 적용 가능**하다.

따라서 두 방식의 차이는 데이터에 레이블이 있는지 여부뿐만 아니라, **사용 환경과 데이터 수집 조건의 현실성**에서도 드러난다. 지도 학습은 정답이 있는 환경에서, 비지도 학습은 레이블이 없거나 정상 데이터만 존재하는 환경에서 활용된다.

혼동 행렬

혼동행렬(Confusion Matrix)은 분류 모델의 성능을 평가할 때 사용하는 표로, 실제 정답과 모델 예측 간의 관계를 4가지 값으로 정리한다. 이 값은 TP(참 양성), FP(거짓 양성), TN(참 음성), FN(거짓 음성)으로 구성된다. 이를 통해 정확도, 정밀도, 재현율, F1 점수 같은 다양한 성능 지표를 계산할 수 있다. 특히 이진 분류 문제에서 오분류 유형을 분석하는 데 유용하다.

혼동 행렬(confusion matrix)이란?

	실제 Positive (1)	**실제 Negative (0)**
예측 Positive (1)	True Positive (TP)	False Positive (FP)
예측 Negative (0)	False Negative (FN)	True Negative (TN)

<의미>

- TP (True Positive): 실제 1인데 모델도 1이라고 맞춤
- TN (True Negative): 실제 0인데 모델도 0이라고 맞춤
- FP (False Positive): 실제는 0인데 모델이 1이라고 틀림 (False Alarm)
- FN (False Negative): 실제는 1인데 모델이 0이라고 틀림 (놓침)

분류 성능 지표 비교: AUC, F1, 정밀도, 재현율, 특이도

1. AUC (Area Under the ROC Curve)

- **의미**: 모델이 양성(Positive)과 음성(Negative)을 얼마나 잘 구별하는지 측정
- **해석**:
  - AUC ≒ 0.5: 무작위 예측 수준
  - AUC ≒ 1.0: 완벽한 분류 성능
- **장점**:
  - 분류 **임계값(threshold)**에 영향을 받지 않음
  - 클래스 불균형 문제에 강함

---

2. F1 Score

- **의미**: 정밀도(Precision)와 재현율(Recall)의 조화 평균 `F1 = 2 × (Precision × Recall) / (Precision + Recall)`
- **사용 시기**:
  - 정밀도와 재현율을 **균형 있게 고려**해야 할 때
  - FP와 FN 모두 중요한 경우 (예: 의료 진단, 사기 탐지)

---

3. Recall (재현율, Sensitivity)

- **의미**: 실제 양성 중에서 모델이 맞게 예측한 비율 `Recall = TP / (TP + FN)`
- **중요한 경우**:
  - 양성을 **놓치면 안 되는 상황** (예: 암 진단)

---

4. Precision (정밀도)

- **의미**: 양성으로 예측한 것 중 실제 양성인 비율 `Precision = TP / (TP + FP)`
- **중요한 경우**:
  - 거짓 양성(False Positive)의 **비용이 클 때**

- 예: 테러 용의자 탐지, 스팸 필터링

---

5. Specificity (특이도)

- **의미**: 실제 음성 중에서 모델이 정확히 음성이라 예측한 비율 `Specificity = TN / (TN + FP)`
- **중요한 경우**:
  - **불필요한 검사나 진단**을 줄여야 할 때 (예: 과잉진단 방지)

성능 지표 요약 비교표

지표	초점	임계값 의존성	불균형 대응	대표 사용 사례
AUC	전체 구분 능력	❌ 아니오	✅ 강함	모델 전체 성능 평가
F1 Score	정밀도와 재현율의 균형	✅ 예	⚠️ 보통	FP와 FN 모두 중요한 상황
Recall	놓치지 않는 예측	✅ 예	⚠️ 보통	민감한 탐지 (암 진단, 사기 탐지 등)
Precision	정확하게 맞추기	✅ 예	⚠️ 보통	FP 비용이 큰 상황 (스팸 필터 등)
Specificity	음성 정확도	✅ 예	⚠️ 보통	과잉 진단 방지 (건강검진 등)

> 💡 참고: TP = True Positive, FP = False Positive, TN = True Negative, FN = False Negative

AUC vs F1 Score in Medical Applications

📌 AUC가 더 중요한 상황 (의료 분야 초기 평가에서 자주 사용)

- 모델의 **전체 분류 성능**을 평가할 때 사용
- **임계값을 정하기 전**, 모델 개발 초기에 유용
- **클래스 불균형(예: 1% 양성, 99% 음성)** 데이터에 효과적
- **모든 임계값 구간에서 성능 분석 가능**
- **예시**: 암 진단 모델에서 ROC 곡선을 사용해 **민감도(Sensitivity)**와 **특이도(Specificity)**의 트레이드오프를 분석할 때

📌 F1 Score가 더 중요한 상황

- **임계값이 고정**된 경우
- **양성 클래스의 성능**에 집중
- *거짓 음성(False Negative)**과 **거짓 양성(False Positive)** 모두가 중요한 경우
- **예시**: 응급 경보 시스템에서 거짓 경보는 피로를 유발하고, 진짜 경고를 놓치는 건 치명적일 수 있음

📊 요약 비교표

항목	AUC	F1 Score	의료 분야 해석
임계값 의존성	❌ 없음	✅ 있음	AUC는 **임계값 이전 평가**에 적합
클래스 불균형 대응	✅ 우수	⚠️ 보통	AUC는 **불균형 데이터에서도 안정적**
중점	전체 분류 능력	양성 클래스의 성능	F1은 **진짜 양성 검출에 집중**
사용 시점	모델 개발 초기	배포/운영 단계	F1은 **임계값 설정 후 실사용 평가**에 적합

✅ 결론

- **AUC**는 모델을 설계하거나 비교할 때, 특히 **불균형 데이터셋**에서 **전체적인 성능을 평가**할 때 적합
- **F1 Score**는 **모델이 실제로 사용되는 상황**, 즉 **임계값이 고정되고** 정밀도/재현율 간 트레이드오프가 중요한 **배포 단계에서 더 중요**
- **의료 인공지능에서는** 일반적으로 **AUC로 후보 모델을 고른 후**, 실제 운영 단계에서는 **F1, 재현율, 정밀도 등**을 활용하여 성능을 미세 조정함

손실 함수 3가지

- **MSE (Mean Squared Error, 평균 제곱 오차)**

  회귀 문제에서 사용되며, 예측값과 실제값의 차이를 제곱해 평균낸 값이다.

  오차가 클수록 손실이 더 커지므로 큰 오차에 민감하다.

  공식: `MSE = (1/n) ∑(y - ŷ)²`

- **Binary Cross Entropy (이진 교차 엔트로피)**

  이진 분류에서 사용되며, 예측 확률과 실제 라벨(0 또는 1)의 차이를 확률적으로 계산한다.

  예측이 정답과 멀수록 손실이 커진다.

  공식: `- [y log(p) + (1 - y) log(1 - p)]`

- **Categorical Cross Entropy (다중 클래스 교차 엔트로피)**

  다중 클래스 분류에서 사용되며, 정답 클래스의 확률 값을 기준으로 손실을 계산한다.

  Softmax 출력과 함께 쓰이며, 예측이 정답에서 멀어질수록 손실이 커진다.

  공식: `- ∑(yᵢ log(pᵢ))`

경사하강법(Gradient Descent)에서 가중치 업데이트

경사하강법은 손실 함수의 기울기를 따라 가중치를 반복적으로 조정하여 손실을 최소화하는 최적화 알고리즘이다.

💡 용어 설명

- $w_{current}$: 현재 가중치
- η (learning rate): 학습률, 얼마나 크게 업데이트할지를 조절하는 상수
- $\frac{\partial L}{\partial w}$: 손실 함수 L을 가중치 w에 대해 편미분한 값 (기울기)

⚙️ 핵심 개념

- 기울기가 **양수**이면 손실을 줄이기 위해 **가중치를 감소**시킨다.
- 기울기가 **음수**이면 손실을 줄이기 위해 **가중치를 증가**시킨다.
- 학습률이 **너무 크면 발산, 너무 작으면 수렴이 느림**

✏️ 예시

- 손실 함수:

$$L = (y - \hat{y})^2$$

- 가중치 w에 대한 기울기:

$$\frac{\partial L}{\partial w} = -2x(y - \hat{y})$$

- 경사하강법 적용:

$$w_{\text{new}} = w_{\text{current}} - \eta \cdot (-2x(y - \hat{y}))$$

- 최종 정리:

$$w_{\text{new}} = w_{\text{current}} + 2\eta x(y - \hat{y})$$문!

러닝 레이트

러닝 레이트(Learning Rate)는 경사하강법에서 **가중치를 얼마나 크게 업데이트할지를 결정하는 하이퍼파라미터**야.

수식에서 학습률은 보통 η로 나타나고, 다음과 같이 사용돼:

$$w_{\text{new}} = w_{\text{current}} - \eta \cdot \frac{\partial L}{\partial w}$$

이때 η가 너무 크면 **최적값을 지나쳐 발산할 수 있고**, 너무 작으면 **수렴은 되지만 매우 느리게 학습돼**.

그래서 적절한 러닝 레이트 설정은 **학습 속도와 안정성에 매우 중요한 역할**을 해.

그래디언트

그래디언트(Gradient)는 손실 함수를 가중치 등 파라미터에 대해 미분한 값으로, 손실이 가장 빨리 줄어드는 방향과 속도를 의미한다. 경사하강법에서는 이 그래디언트를 따라 가중치

를 업데이트해 손실을 최소화한다. 즉, 그래디언트는 모델이 더 나은 예측을 할 수 있도록 파라미터를 조정하는 기준이 된다.

활성화 함수

활성화 함수는 인공신경망에서 각 뉴런이 출력값을 결정할 때 사용하는 함수로, 입력의 가중합을 비선형적으로 변환해준다. 이 함수 덕분에 모델은 단순한 선형 계산을 넘어서 복잡한 패턴을 학습할 수 있다. 대표적으로 시그모이드, ReLU, tanh 같은 함수가 있으며, 이들은 뉴런이 다음 층으로 신호를 얼마나 보낼지 조절하는 역할을 한다.

시그모이드 함수

시그모이드 함수(Sigmoid function)는 입력값을 0과 1 사이의 확률처럼 변환하는 **활성화 함수**다. 수식은 $\sigma(x) = \frac{1}{1+e^{-x}}$로, 입력이 클수록 1에 가까워지고, 작을수록 0에 가까워진다. 주로 이진 분류 문제에서 출력층에 사용되며, 결과를 '참일 확률'로 해석할 수 있게 만든다. 다만, 기울기가 작아지는 **기울기 소실 문제**로 인해 은닉층에서는 잘 쓰이지 않는다.