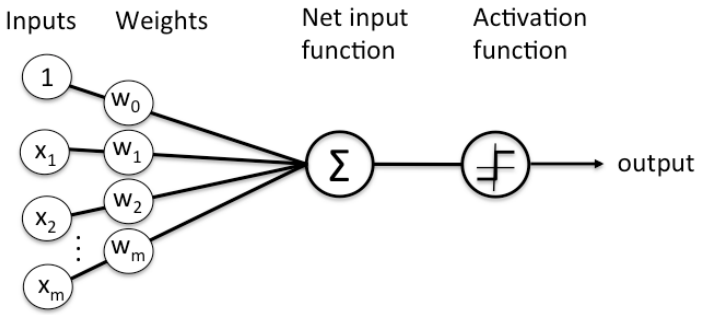
인공 신경망과 DNN 이론

1) 인간의 두뇌
   (1) 1,000억 개의 뉴런들이 100조 개의 시냅스로 열결
   (2) 뇌 학습 - 뉴런은 계산을 담당하며, 시냅스는 정보를 전달하고 저장하는 역할

2) 심층 신경망 : DEEP NEURAL NETWORK(DNN)
   - 입력층과 출력층 사이, 다중의 은닉층 포함
   - 다양하고 복잡한 비선형적 관계 학습 가능
   - 복잡한 비선형 관계도 정확한 모델이 가능
   - DNN 구조 안에 사용되는 뉴련과 시냅스의 숫자도 매우 많음
     - 이러한 DNN을 처리하기 위해서는 큰 규목의 데이터 서버 사용

3) 뉴런의 출력 계산 과정
   (1) 변수



4) 활성화 함수
   (1) AI 반도체
   - MAC (Multiply-and-ADD) 계산기들은 많은 숫자를 처리하고, 이들을 모두 병렬계산에 투입
   - MAC의 결과를 바로 출력값으로 사용하지 않고, 활성화 함수를 거침

   (2) 활성화 함수는 종류가 다양, 장단점과 쓰임새가 다름

   (3) ReLU, Linear, Sigmoid 등

5) 역전파 알고리즘
   (1) 가장 보편적인 인공 신경망 학습 알고리즘
   - 지도 학습 (Supervised Learning)에 사용 : 답이 존재
   - 다층 순행 공급 (Feedforward) 신경망에 사용

   (2) 순방향
   - 입력값 제시 후 출력값 계산을 왼쪽에서 오른쪽 (Forward)
   - 각각의 뉴런에 계산값 저장

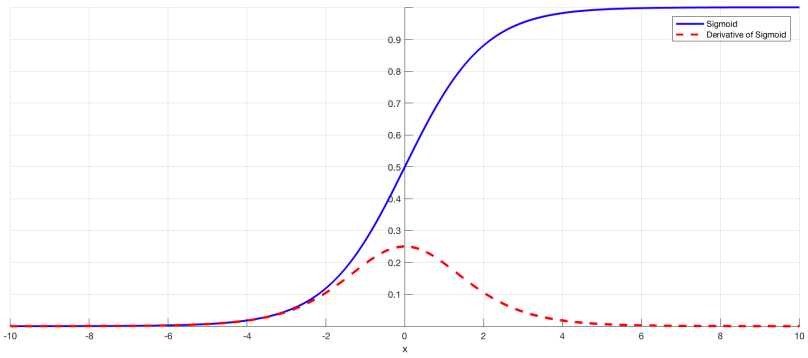
   (3) 역방향
   - 출력값 제시 후 신경망의 가중치 보정은 오른쪽에서 왼쪽 (Backward)
   - 순방향 계산 결과 활용

기울기 유실 문제 (Vanishing Gradient)

1) 기울기 유실

(1) 증상

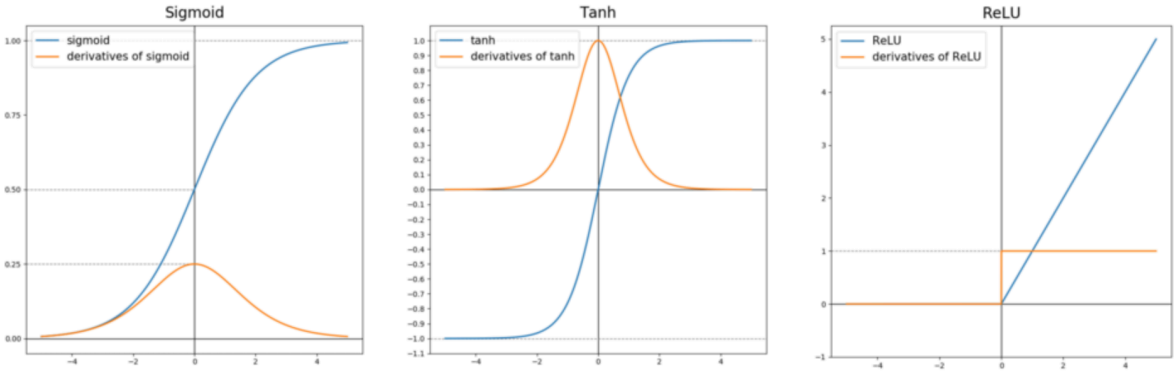
- 아무리 새로운 데이터를 주어도 인공 신경망이 학습을 하지 않음
- 신경망 가중치가 더 이상 변하지 않음
- 이유 : 활성화 함수 sigmoid(x) = $1/(1+e^{-x})$



(2) 기울기의 의미

- 가중치를 보정할 때 사용한 편미분이 기울기
- 기울기 값 0 -> 보정치 0
- X 값(뉴런의 출력값)이 커질수록 기울기는 0에 수렴
- 학습이 후반부에 이르면, 대부분의 뉴런의 출력값은 커지고, 기울기의 값이 0에 가까워짐
- 한 뉴런의 기울기(편미분)는 입력값에 대한 “감도”, sensitivity

2) 해결 방법 : 활성화 함수 교체



심장병 데이터 처리

1) Pandas 데이터 프레임
   (1) 지도학습에서 필수
      • 학습용/평가용, 입력값/출력값으로 분할
      • 나누는 순서는 상관X

학습용 입력값 (X_train)	학습용 출력값 (Y_train)
평가용 입력값 (X_test)	평가용 출력값 (Y_test)

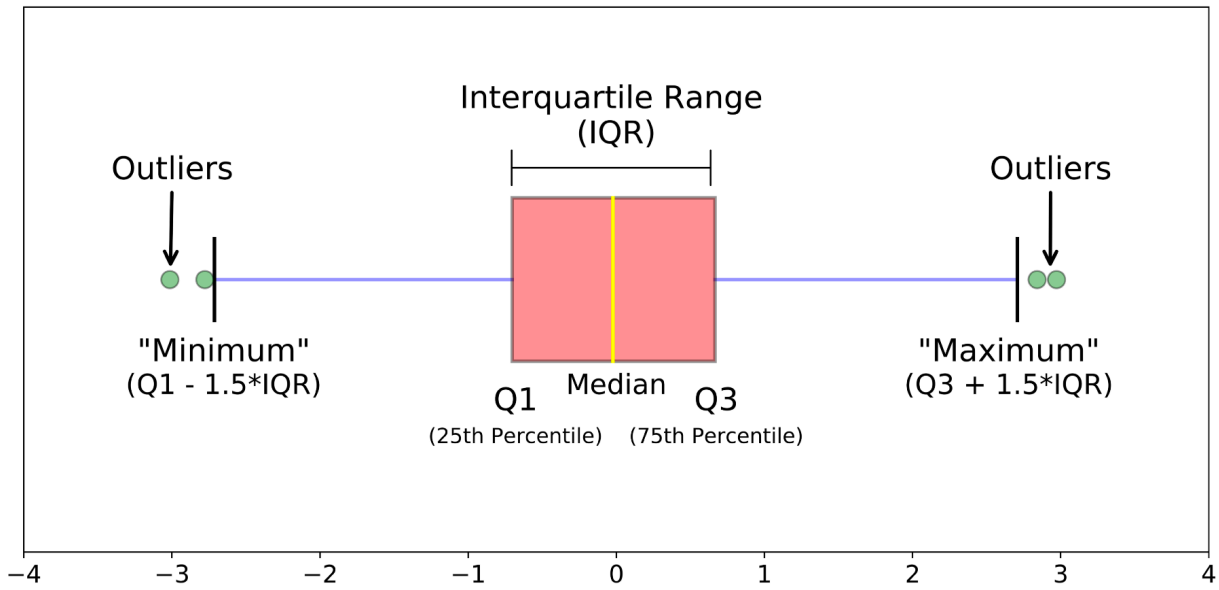
데이터 정규화와 시각화

1) Z-점수 정규화
   (1) 머신러닝 알고리즘
      • 데이터의 특정 패턴을 찾음
      • 새로운 값 예측

   (2) 심장병 데이터의 예시
      • Trestbps : 94 ~ 200
      • Oldpeak : 0 ~ 6.2
      => 기계학습을 할 때 범위가 더 넓은 요소가 큰 영향을 미침

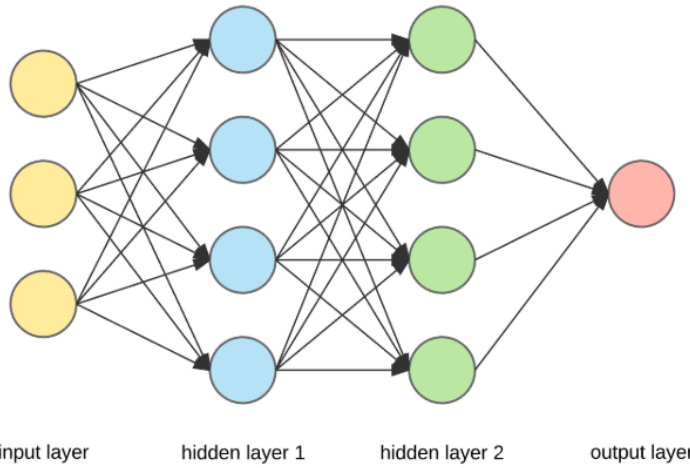
   (3) Z-점수 = (X - 평균) / 표준편차
      • 평균은 0, 표준편차는 1

2) 상자그림 (Box Plot)
   (1) 데이터의 특징을 5가지 값으로 시각적 표현
      • 사분위수 (Q1, 중앙값, Q3)와 최솟값, 최댓값

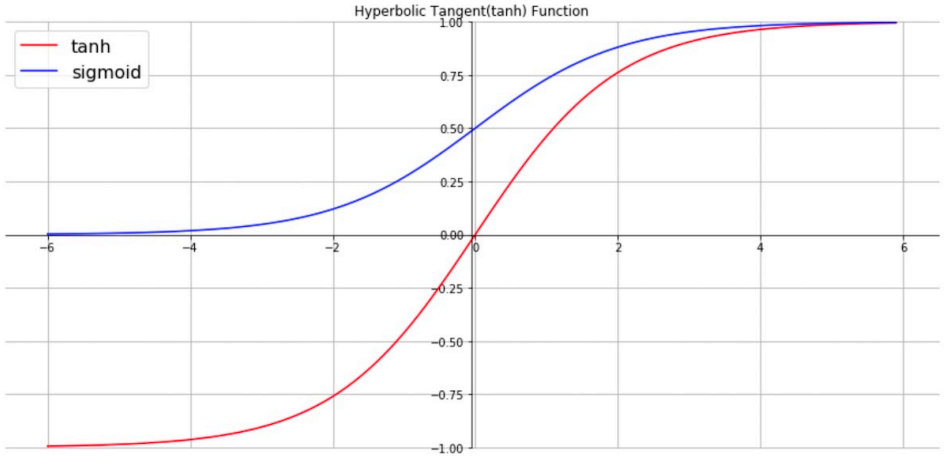


DNN 구현

1) DNN 설계도



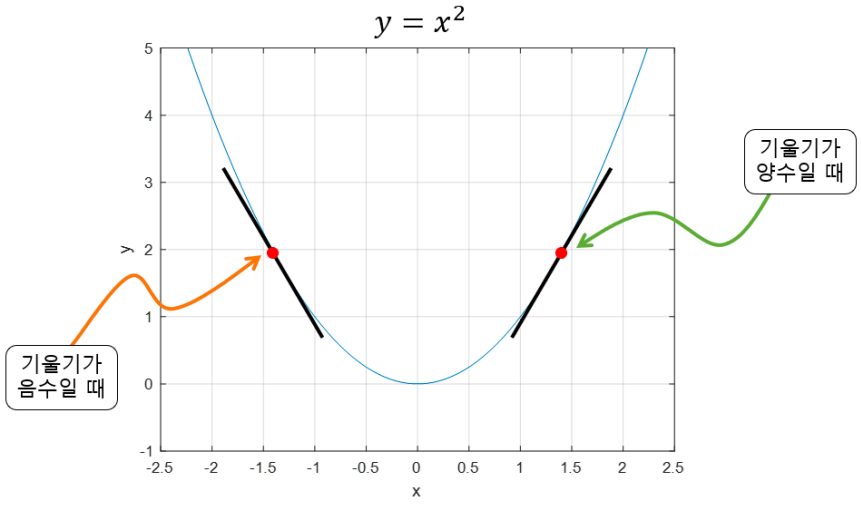
2) Tanh 활성화 함수
   (1) 쌍곡선 함수 중 하나
   - [-1, 1] 데이처 스케일링과 자주 짝을 이룸
   - 뉴런 출력값의 양수 대 음수 균형을 맞춤
   - DNN의 은닉층에서 많이 사용
   - Sigmoid 활성화 함수의 기울기 유실 문제 보완



3) Sigmoid 활성화 함수
   (1) 딥러닝에서 가장 많이 쓰이는 활성화 함수
   - Step function을 대체 : 전 구간에서 미분 가능
   - 장점 : 미분 계산이 간단
   - 단점 : 기울기 유실 문제

DNN 학습

1) 경사 하강법
   (1) Stochastic Gradient Descent(SGD)
   - 지도학습에서 가중치를 보정하는 대표적인 방법
   - 함수의 기울기를 구하여 낮은 쪽으로 이동



2) 평균 제곱 오차
   (1) Mean Square Error (MSE)
   - 예측값과 정답과의 차이의 제곱의 평균
   - 완벽한 예측의 경우 : 0
   - 오차가 커지면 MSE 커짐
   - 제곱오차들의 평균 : 큰 오차들에 더 민감

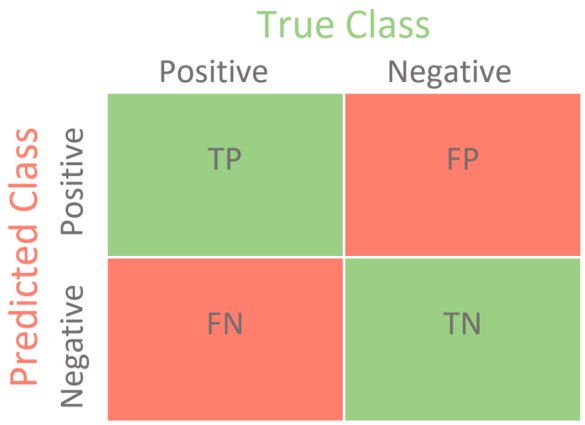
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(predict - truth)^2$$

3) Torch 텐서 VS Numpy 행렬
   (1) 둘 다 n-차원 행렬

	PyTorch 텐서	Numpy n-차원 행렬
연동 package	PyTorch	Python의 모든 package
GPU 지원	O	X
장점	속도, PyTorch와 연동	무게, 안정성, 역사
단점	미완성, 무게	속도

DNN 평가

1) 혼동 행렬 (Confusion Matrix)



		Predicted Failure: True	Predicted Failure: False
Actual Failure	True	TP := True Positive	FN := False Negative
	False	FP := False Positive	TN := True Negative

Recall := TP / (TP+FN)

Precision := TP / (TP+FP)

F1-Score := 2*Precision*Recall / (Precision+Recall)