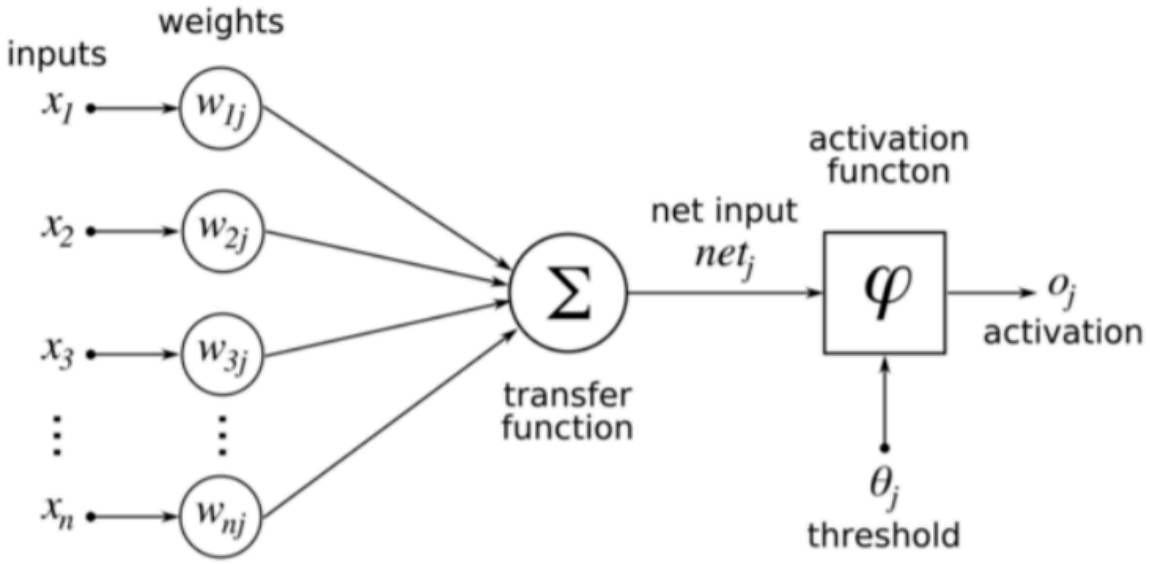
3-2-2 신경망 모형

Artificial Neural Network 요약

- 인공 신경망(ANN) 모형은 인간의 뉴런을 모방하여 알고리즘으로 구현한 방법
  - 인간의 뉴런은 시냅스를 통하여 다른 여러 뉴런으로 부터 자극을 전달받고 이를 또 다른 뉴런으로 전달시키는 과정을 거침
  - 이 인간의 뉴런을 "퍼셉트론"이라는 인공 뉴런을 구성
  - 이 퍼셉트론들이 여러 layer를 거치며 시냅스를 통한 자극의 전달과정을 모방한 것이 Neural Network
- Neural Network에서는 가장 중요한 구성 요소
  - inputs
  - weights
  - activation function
  - bias(error) 등



- Neural Network의 과정을 간략하게 말하자면,

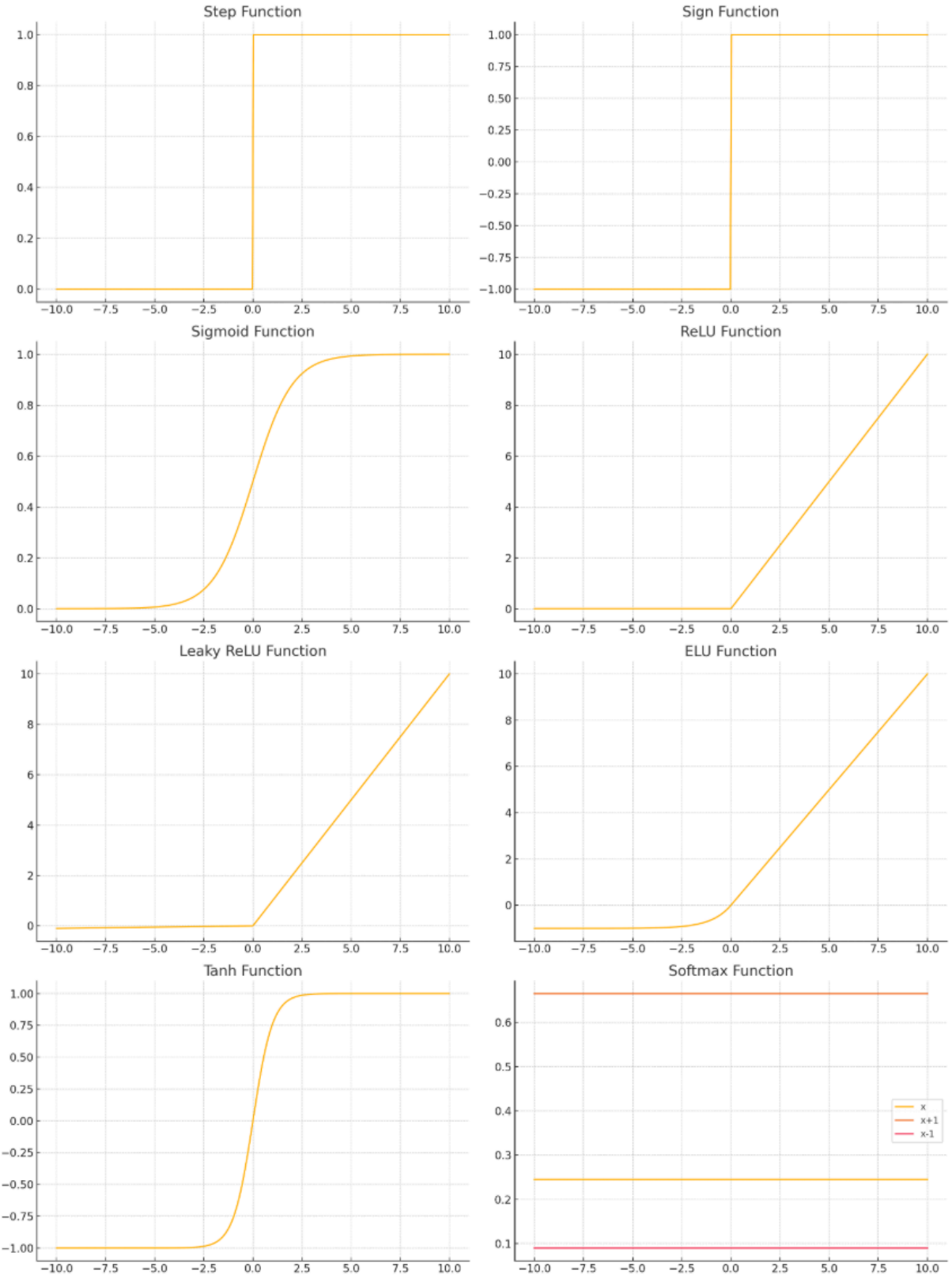
- input layer에서 inputs 값들이 여러 개의 퍼셉트론으로 이루어진 다층 퍼셉트론 층(layer), 이를 은닉층(Hidden Layer)라고 하는 층을 거치며 Weigths와 error를 활성화함수를 통해 조정하면서 학습을 진행해 출력층(Output layer)로 결과를 출력하는 과정
- 이때, activation function(활성화 함수)
    - sigmoid function / ReLU / leaky ReLU / ELU 등 여러가지 존재하지만,

1. 경사 소실(gradient vanishing) 문제
2. 중간에 Global minimum을 찾지 못하고, local minimum에서 학습을 멈추는 문제

와 같은 이유로 ReLU관련 활성화 함수를 요즘은 많이 사용

다양한 활성화 함수

이름	수식	출력 범위	주요 특징
계단 함수	$f(x) = 1$ if $x \geq 0$, else 0	{0, 1}	퍼셉트론 초기 모델에서 사용됨. 연속적이지 않아서 역전파 학습 불가
부호 함수	$f(x) = 1$ if $x > 0$, -1 if $x < 0$, else 0	{-1, 0, 1}	방향만 제공함. 연속성이 없고 딥러닝에 비효율적
시그모이드	$\sigma(x) = \frac{1}{1+e^{-x}}$	(0, 1)	출력이 확률처럼 해석됨. 그래디언트 소실 문제 존재
ReLU	$f(x) = \max(0, x)$	[0, ∞)	계산 단순, 학습 빠름. 음수 입력에서는 뉴런이 죽을 수 있음
Leaky ReLU	$f(x) = x$ if $x > 0$, else $0.01x$	(-∞, ∞)	ReLU 개선형. 음수도 약간 반응하게 해 죽은 뉴런 문제 완화
ELU	$f(x) = x$ if $x \geq 0$, else $\alpha(e^x - 1)$	(-α, ∞)	음수도 자연스럽게 반영. 출력 평균이 0에 가까워 학습 안정
tanh	$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$	(-1, 1)	시그모이드보다 중심화된 출력. 그래디언트 소실 존재
softmax	$f(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}}$	(0, 1), 합=1	출력층 전용. 벡터를 확률 분포로 정규화함


Step Function
Sign Function
Sigmoid Function
ReLU Function
Leaky ReLU Function
ELU Function
Tanh Function
Softmax Function
x
x+1
x-1

딥러닝(Deep Learning) 개요

1. 딥러닝의 정의와 배경

- **정의**: 여러 개의 은닉층(hidden layer)을 가진 인공신경망(Artificial Neural Network, ANN)을 기반으로 하는 학습 알고리즘
- **출발점**: 2006년 Hinton이 제안한 Deep Belief Network 이후, GPU의 발전과 대용량 데이터의 등장으로 본격화 됨
- **기초 원리**: 뉴런 간 가중치(weight)와 편향(bias)을 조정하여 입력 데이터로부터 출력 예측을 최적화 함

2. 딥러닝의 핵심 구조

- **층 구조**:
  - 입력층(Input Layer): 데이터를 받아들이는 층
  - 은닉층(Hidden Layers): 특징을 추출하고 변환하는 층 (2개 이상일 때 '딥'이라 부름)
  - 출력층(Output Layer): 최종 결과를 출력
- **활성화 함수**: ReLU, sigmoid, tanh 등은 비선형성을 부여해 복잡한 함수 근사를 가능하게 함
- **학습 방식**: 오차역전파(Backpropagation) + 경사하강법(Gradient Descent)

3. 주요 딥러닝 모델 유형

- **CNN(Convolutional Neural Network)**: 이미지 인식, 객체 탐지 등에 사용. 필터를 이용한 지역적 특징 학습에 탁월함.
- **RNN(Recurrent Neural Network)**: 순차적 데이터(텍스트, 음성 등)에 적합. 시간 정보를 기억함. LSTM, GRU로 발전.
- **GAN(Generative Adversarial Network)**: 생성자와 판별자의 경쟁을 통한 데이터 생성. 예술, 합성 이미지, 음성 생성 등에 활용.
- **Transformer**: 문맥 기반 자연어 처리의 혁신. BERT, GPT 시리즈로 이어지며 현재의 주류로 자리잡음.

딥러닝의 주요 종류

1. 입력 데이터 형태에 따른 분류

- **CNN (Convolutional Neural Network)**
  - 이미지, 영상과 같이 공간적 특성이 있는 2D 또는 3D 데이터에 최적화
  - 합성곱 층(conv layer), 풀링 층(pooling) 사용
  - 대표 응용: 이미지 분류, 객체 검출, 얼굴 인식
- **RNN (Recurrent Neural Network)**
  - 텍스트, 음성, 시계열 데이터처럼 순차적인 입력에 적합
  - 시점 간 정보를 순환 구조로 기억
  - 대표 응용: 번역, 음성 인식, 시계열 예측
- **Transformer**
  - RNN의 단점을 해결한 구조로, 전체 시퀀스를 한 번에 처리함
  - Self-Attention 메커니즘 사용
  - 대표 응용: 챗봇, 번역, 문서 요약 등 자연어 처리 전반

2. 학습 목적에 따른 분류

- **Discriminative Models (판별 모델)**
  - 입력을 주어진 범주로 분류하거나 값을 예측함
  - 예시: CNN 기반 이미지 분류기, RNN 기반 감성 분석기
- **Generative Models (생성 모델)**
  - 새로운 데이터를 만들어냄. 분포를 학습하여 샘플 생성 가능
  - 예시:
    - **GAN (Generative Adversarial Network)**: 진짜 같은 이미지 생성
    - **VAE (Variational Autoencoder)**: 잠재 공간에서 샘플 생성
    - **Diffusion Model**: 고품질 이미지 생성 (ex. DALL·E, Stable Diffusion)

3. 학습 방식에 따른 분류

- **지도학습(Supervised Learning)**
  - 라벨이 있는 데이터로 학습
  - 예시: 이미지에 개/고양이 라벨 붙여 학습하는 CNN
- **비지도학습(Unsupervised Learning)**
  - 라벨 없이 데이터 구조를 파악
  - 예시: Autoencoder, 클러스터링
- **강화학습(Reinforcement Learning)**
  - 보상을 극대화하는 방향으로 에이전트를 학습시킴
  - 딥러닝과 결합해 '딥 강화학습'이 등장함 (예: 알파고, DQN(Deep Q-Network))

기타 중요 구조들

- **Autoencoder**
  - 입력을 압축하고 복원함. 차원 축소, 노이즈 제거에 활용
- **Capsule Network**
  - CNN의 공간적 표현 한계를 보완하려는 구조. 아직 연구 중
- **Residual Network (ResNet)**
  - 깊은 네트워크에서 기울기 소실 방지 위해 skip connection 사용

실습

nnet 패키지를 이용해 Iris 데이터셋을 분류하는 신경망 모델을 학습하는 예제

```
## 신경망 라이브러리
install.packages("nnet")
library(nnet)
```

```
## 신경망 모형 생성
nn.iris ← nnet(Species~., data=iris, size=2, rang=0.1, decay=5e-4, maxit=2
00)
```

- Species를 목표 변수로 하고, 나머지 4개의 꽃 특성 변수들이 입력
  은닉층 1층에 뉴런 2개(size=2)를 사용하며,
- L2 정규화(decay)와 학습 반복 제한(maxit=200) 도 포함
  - L2 정규화(L2 Regularization)는 **모델 학습 과정에서 과적합(overfitting)을 방지**하기 위해 사용되는 기법 중 하나
  - L2 정규화는 모델의 가중치(weight)가 너무 커지지 않도록 제약을 주어 모델을 더 일반화하는 데 도움을 줌
  - 손실 함수(loss function)에 **가중치의 제곱합**을 추가하는 방식으로 작동하여 모델이 너무 복잡해지는 것을 방지하고, 가중치가 지나치게 커지는 것을 억제하여 과적합을 줄이는 효과
- 멀티클래스 분류(Multiclass classification) 문제로 매우 전형적인 예제

주요 파라미터 설명

- `Species~.` :

  `Species`를 예측하는 모델을 만들되, 입력 변수는 iris 데이터셋의 나머지 모든 열(`.`)로 설정

- `data=iris` :

  기본 내장된 `iris` 데이터셋을 사용

- `size=2` :

  은닉층에 뉴런 2개를 사용

- `rang=0.1` :

  가중치 초기값 범위 설정 (−0.1 ~ 0.1 사이)

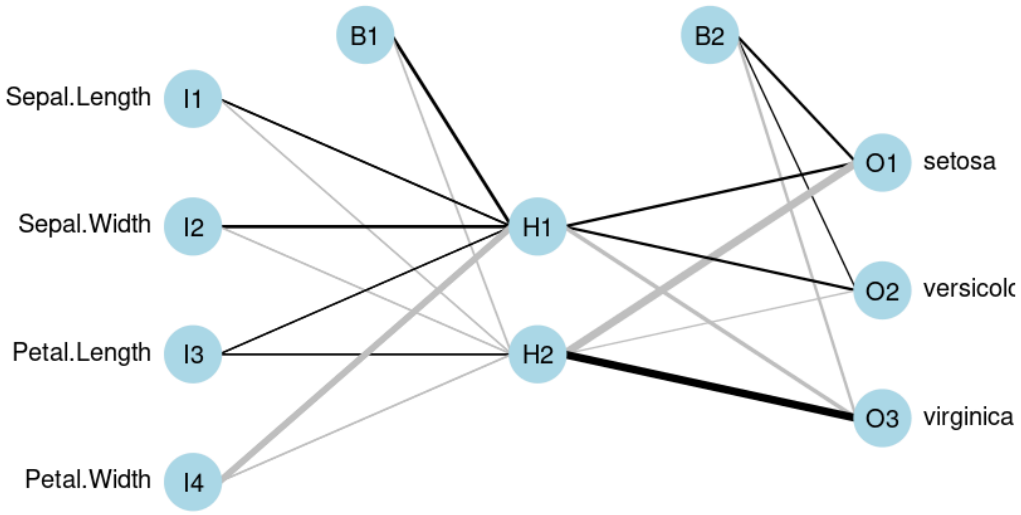
- `decay=5e-4` :

  가중치 감소(=L2 정규화). 과적합 방지용

  `decay=5e-4`는 L2 정규화의 강도를 나타내며, 가중치의 제곱합에 **0.0005**를 곱한 값을 손실 함수에 추가하여 모델의 과적합을 방지하는 역할

- `maxit=200` :

최대 학습 반복 횟수를 200으로 설정



종료