알고리즘을 활용한 AI 학습 시뮬레이션
– 유전 알고리즘과 인공신경망 분류 속 수학

차민경
(김해제일고등학교 교사)

Ⅰ. 알고리즘의 수학적 정의와 역사적 배경

1. 알고리즘의 의미

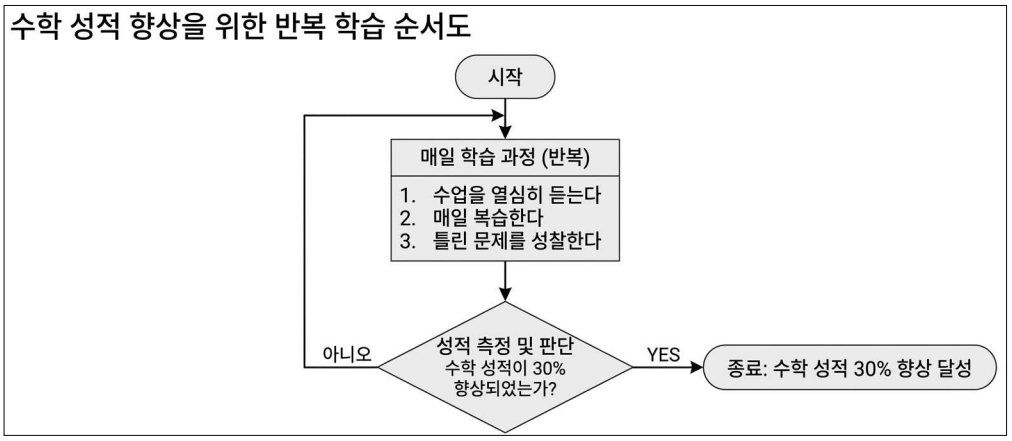
알고리즘은 주어진 문제를 해결하기 위해 설계된 논리적이고 절차적인 해결 방법을 의미한다. 알고리즘은 단순히 해결책의 나열이 아니라 특정 조건에 따라 판단하거나, 반복하는 등의 절차적 과정으로 구성된다. 순서도(flowchart)를 이용해 시각적으로 구조화하여 효과적으로 나타내기도 한다.

2. 일상적 문제를 통한 알고리즘의 도식화

예를 들어, '수학 성적을 30% 향상시켜야 한다'는 문제 상황으로부터 출발하여 해결에 이르기까지의 알고리즘을 짜고 이를 도식화하자. 구체적인 방법으로 아래와 같은 것들이 있을 것이다.

> ▫ 수업을 집중하여 듣는다.
> ▫ 그 날 배운 내용을 복습한다.
> ▫ 틀린 문제를 성찰한다.

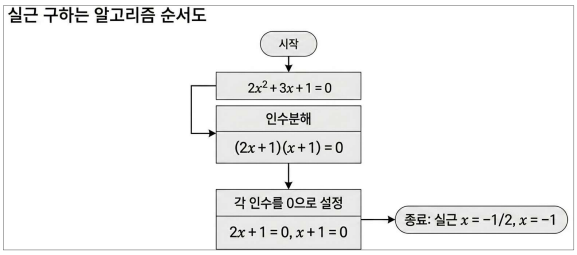
이 과정을 수학 성적이 30% 향상될 때까지 매일 반복하면 아래와 같은 순서도로 나타낼 수 있는 하나의 알고리즘이 된다.

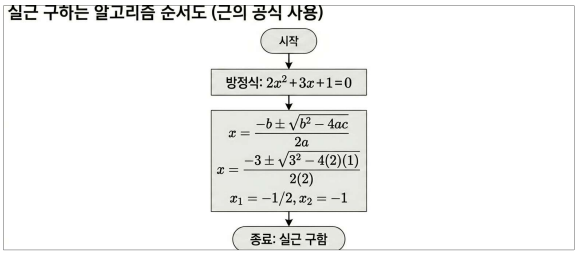


3. 방정식의 실근 구하기 알고리즘 분석

가. 이차방정식 $x^2-5x+6=0$의 해법 알고리즘

이차방정식 $x^2-5x+6=0$를 푸는 과정을 나열하면 다음과 같다. 첫째, 좌변을 $(x-2)(x-3)$으로 인수분해한다. 둘째, 각 인수를 0으로 만드는 x의 값을 구한다. 또는 근의 공식을 이용해서 한 번에 해를 구할 수도 있다. 간단한 과정이지만 이 또한 하나의 알고리즘이다.

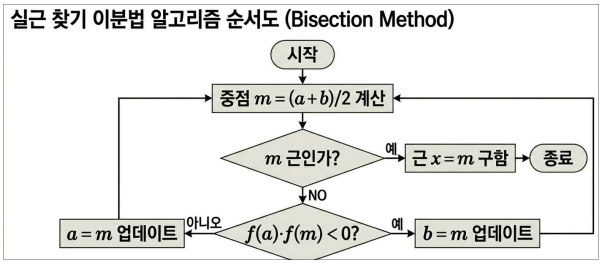




알고리즘이라는 용어를 모르는 중·고등학생들도 쉽게 떠올릴 수 있는 방법이다. 그래서 이 방정식으로는 알고리즘의 필요성에 공감하기 힘들 수도 있다. 하지만, $-0.98x^3+1.87x^2-3.01x+4.03=0$과 같은 방정식은 인수분해나 근의 공식으로 풀 수 없다. 교과서에 수록된 정제된 정수 계수의 정형화된 방정식과 달리, 실제 산업 현장이나 미세 공정에서 발생하는 고차방정식이나 계수가 복잡한 방정식은 인수분해나 근의 공식과 같은 방법으로 한 번에 해결하기는 어렵다. 이런 경우에는 다음과 같은 알고리즘이 요구된다.

나. 방정식 $-0.98x^3+1.87x^2-3.01x+4.03=0$ 의 해법 알고리즘

이분법(Bisection) 알고리즘은 해가 존재할 것으로 예상되는 구간의 중간값을 구한 뒤, 양 끝점의 함숫값 부호가 서로 다른 절반의 구간을 선택해 나가는 과정을 반복하며 실근의 어림값을 구하는 방식이다. 순서도로 아래와 같이 나타낼 수도 있으며, 프로그래밍 언어로 변환하기만 하면 어려운 계산 과정을 컴퓨터로 반복할 수도 있다. 아래는 실제로 파이썬 프로그래밍을 통해 실근의 어림값 $1.597\cdots$을 얻어내는 과정이며, 그래핑계산기로 시각화해 보면 대략 실근의 어림값을 올바르게 구했음을 검증할 수도 있다.



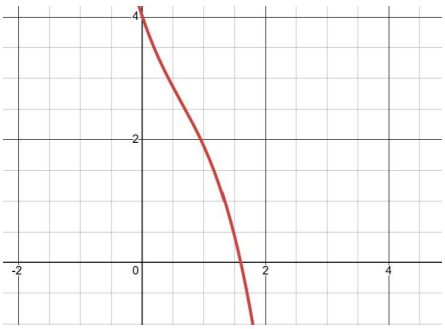

```python
a = float(input('Input a : '))
b = float(input('Input b : '))

print("- - Setting a 3-dim function px^3 + qx^2 + rx + s- - ")

p = float(input("Input p:"))
q = float(input("Input q:"))
r = float(input("Input r:"))
s = float(input("Input s:"))

print([p,q,r,s,a,b])

def f(x,_pow_) :  3+개의 사용 위치
    global p
    global q
    global r
    global s
    return p*x**3+q*x**2+r*x+s
```

```python
while (b-a) > 0.000001 :
    m = (a+b) / 2
    if f(b) == 0 :
        break
    elif f(a) * f(m) < 0 :
        b = m
    else :
        a = m
    print([a,b])

root = (a+b)/2
print(f"Final root is estimated : {root}")
```

이분법 알고리즘 파이썬 프로그래밍 코드

```
[1.59710693359375, 1.5972900390625]
[1.59710693359375, 1.597198486328125]
[1.5971527099609375, 1.597198486328125]
[1.5971527099609375, 1.5971755981445312]
[1.5971641540527344, 1.5971755981445312]
[1.5971641540527344, 1.5971698760986328]
[1.5971641540527344, 1.5971670150756836]
[1.597165584564209, 1.5971670150756836]
[1.597165584564209, 1.5971662998199463]
Final root is estimated : 1.5971659421920776

종료 코드 0(으)로 완료된 프로세스
```

파이썬 프로그래밍 결과 이분법을 반복 시행하여 어림값을 구함.



그래핑계산기로 시각화한 결과.

Ⅱ. 유전 알고리즘과 벡터 최적화

1. 유전 알고리즘(Genetic Algorithm)의 의미

유전 알고리즘은 찰스 다윈의 적자생존 및 자연선택설을 컴퓨터 알고리즘으로 설계한 것으로서, 확률적 탐색 알고리즘의 한 종류이다. 유전 알고리즘의 구성 요소 및 프로세스는 다음과 같다.

① 초기 세대(population) 생성
: 가상 환경에 다양한 형질을 가진 개체군을 무작위로 생성

↓

② 적합도 평가(Fitness Evaluation)
:각 개체가 주어진 환경에서 얼마나 우수한 성능을 보이는지 측정

↓

③ 선택(Selection)
: 우수한 적합도를 가진 개체(엘리트 개체)를 부모 개체로 선정

↓

④ 교차(Crossover)
:부모 세대의 유전 정보를 무작위로 조합하여 자식 세대를 생성

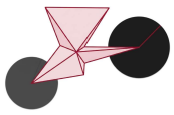
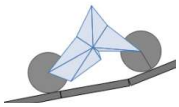
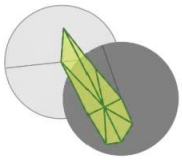
↓

⑤ 돌연변이(Mutation)
: 일정 확률로 유전 정보에 미세한 변화를 주어 지역 최적해(Local solution)에 갇히는 것을 방지

2. 자동차 진화 시뮬레이션을 통해 알아보는 유전 알고리즘 속 수학

가. 확률적 기법을 통한 유전 알고리즘

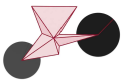
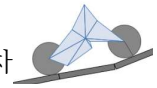
2D Car Evolution 시뮬레이션을 시작하면 모양도, 바퀴도 제멋대로인 2D 자동차 스무 대가 출발한다. 어떤 차는 출발하자마자 뒤집어지고, 어떤 차는 꽤 멀리 굴러간다. 스무 대의 차가 울퉁불퉁한 도로를 제각기 질주하고 모두 멈추면 각 자동차가 이동한 거리가 나오면서 1세대가 종료된다. 예를 들어, 1세대 자동차 주행 기록이 아래와 같다고 하자.

자동차	1등 차 	2등 차 	…	20등 차 
주행거리	100m	40m	…	10m

이 물리 시뮬레이션은 돌연변이 발생률, 엘리트 개체 수 등을 조절하여 유전 알고리즘을 통해 더 긴 거리를 주행하도록 자동차를 진화시킬 수 있는 구조로 되어 있다. 이제, 다음 세대 자동차를 만들어야 한다. 스무 대의 자동차 중 어느 자동차를 부모로 삼아야 하는가?

당연히 주행기록이 가장 높은 1,2등 차끼리만 교배시켜서 다음 세대 자동차를 만들면 될 것이라 생각하기 쉽지만 그렇게 하면 다음 세대 자동차들은 전부 1,2등차와 똑같이 생겨 작고 날렵할 것이다. 만약 다음 코스에 작은 구덩이가 나오면 이 모양의 차들은 모두 구덩이에 빠져서 전멸할 지도 모른다. 이는 생물학에서 말하는 유전적 다양성 부족 현상이다. 하지만 만약 꼴등인 녹색 차의 유전자도 유전된다면, 지금은 둔탁한 모양새답게 꼴등이지만, 그 큰 바퀴가 언젠가 장애물을 넘는 중요한 수단이 될지도 모른다. 이것이 꼴등의 유전자도 함께 유전시키는 이유이다.

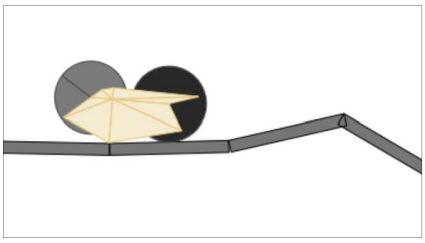
이 때, 확률적 판단이 필요하다. 아래와 같이 자동차들의 이동거리를 모두 더하고, 원판에서 각 자동차의 영역을 그 이동거리에 비례해서 나눈다.

자동차	1등 차 	2등 차 	…	20등 차 
주행거리	100m	40m	…	10m
원판 영역	100 / 150 **(약 66%)**	40 / 150 **(약 26%)**	…	10 / 150 **(약 6%)**

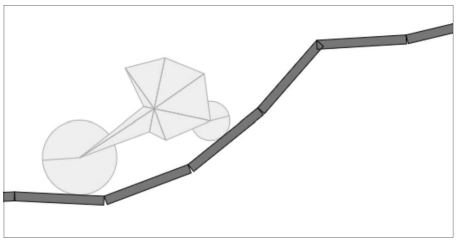
그러면 부모 자동차로 1등 차가 선택될 확률은 약 $\frac{100}{150}$ 즉, 약 66%, 20등 차가 선택될 확률은 약 6%이다. 이러한 룰렛 휠 선택 덕분에 이번 세대에 갇혀 있지 않고 다양한 형질이 유전되어 새로운 도로 여건도 뚫고 주행하는 더 나은 세대의 자동차가 탄생할 수 있다. 이처럼 우수한 차에게는 넓은 면적을 주되, 꼴등 차에게도 좁게나마 원판의 자리를 내어주어 확률적으로 부모를 뽑는 원리를 룰렛 휠 돌리기 기법이라고 한다. 1등의 우수한 유전자를 보존하면서도 꼴등의 유전자까지 포용하여 현재 세대의 최댓값(지역해, Local zero)에 갇혀 있지 않는 지혜에 확률이 쓰인다는 것은 놀라운 일이다.

나. 유전 알고리즘에서의 벡터

사람의 특징을 결정하는 변수나 유전 정보가 수만 가지 되듯이, 이 물리 시뮬레이션에서의 자동차 역시 12개의 매개변수를 갖는다. 앞바퀴 크기, 뒷바퀴 크기, 차체의 높이, 무게 중심 등이다. 유전 알고리즘에서 이 각각의 매개변수는 유전자(gene)가 되며, 이들이 모인 12개 매개변수 세트는 자동차 개체의 형질을 결정하는 염색체(Chromosome)의 역할을 한다. 처음에는 주행에 불리한 매개변수들로 이루어진 자동차들을 유전 알고리즘을 통해 진화시키고 나면 점차 해당 지형의 주행에 최적화된 구조를 갖춘 자동차가 됨을 확인할 수 있다.

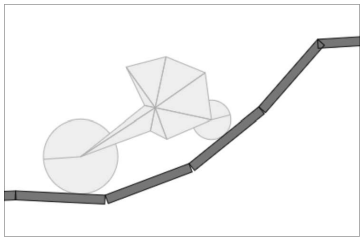


1세대 자동차. 주행거리 10여 미터.



56세대 자동차. 주행거리 320미터. 진화하면서 뒷바퀴가 커져 경사면을 쉽게 오름.

유전 알고리즘에서 매개변수를 최적화할 때, 진화의 대상이 되는 각 개체는 주로 수학적 벡터(vector)로 표현된다. 예를 들어, 앞서 언급한 12개의 매개변수를 갖는 이번 시뮬레이션의 자동차 개체는 아래와 같이 12차원 벡터로 표현할 수 있다.



앞바퀴 크기	뒷바퀴 크기	무게중심에서 각 꼭짓점까지의 거리 합	…	차체 길이
2	5	17		10

$\Leftrightarrow \vec{v} = (2,5,17, \cdots, 10)$

즉, 12개의 유전 정보(변수)를 잘 조절하여 주행거리를 최대로 만드는 최적화 문제가 된다. 변수가 한두 개 뿐이고 식을 알고 있다면 수학 시간에 배우는 이차함수의 최대·최소 문제, 미분처럼 공식을 이용해 쉽게 해결할 수 있다. 하지만 이 시뮬레이션에서는 앞바퀴 크기, 차체 높이 등 12개의 변수가 울퉁불퉁한 도로, 중력, 마찰력 같은 복잡한 환경과 얽혀 있다. 그래서 주행거리를 계산해 내는 수학 공식(함수식) 자체를 만드는 것이 아예 불가능하다. 유전 알고리즘은 이러한 상황에서 훌륭한 해결책이 된다.

Ⅲ. 분류 알고리즘

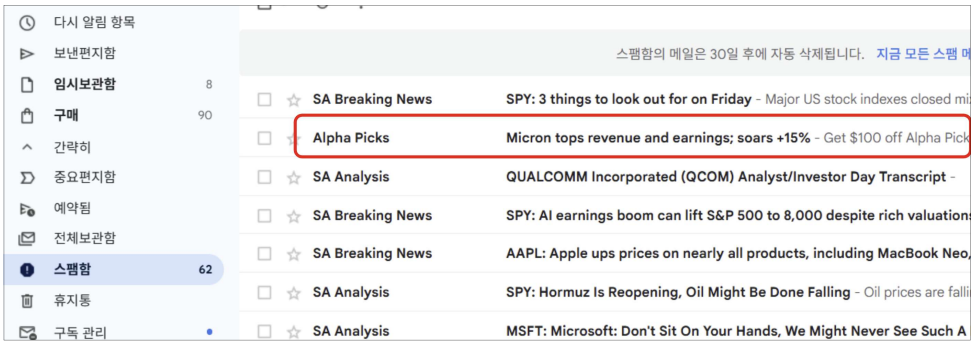
1. 실생활에서의 분류 문제

가. 스팸 메일 분류 - 단순 규칙 기반 알고리즘에서 AI 분류 알고리즘으로.

스팸 메일을 걸러 내는 장치는 인공지능이 보편화되기 전부터 있어 왔다. 대개 메일 제목에 특정 단어가 포함되거나, 메일 서버가 불건전한 곳에 위치하면 걸러내는 식의 단순 규칙에 기반한 알고리즘이었다. 이러한 단순 규칙 기반 시스템은 기피 단어에 특수문자를 추가하거나 의도적인 오타를 내는 등의 속임수로 얼마든지 피해갈 수 있었다.

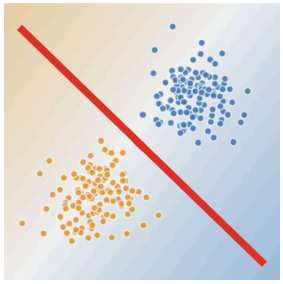
그러나, 인공지능이 보편화된 지금은 그러한 단순 규칙기반 알고리즘 대신 머신러닝 기반 분류 알고리즘을 도입하여 메일 본문의 텍스트 분포와 문맥을 파악하고 확률적으로 스팸 메일을 분류하여 차단한다. 예를 들어, 아래와 같이 겉보기에는

일반 경제 뉴스 같아 보이는 메일이 스팸 메일로 분류되어 스팸함에 들어 있는 것을 볼 수 있다. 실제로 이 메일은 경제 뉴스로 시작하여 결국 재테크 관련 구독 서비스 결제로 이어지는 광고성 메일이었으므로 AI 알고리즘이 스팸 메일을 잘 걸러낸 사례에 해당한다.



2. 인간의 분류 직관

인간은 사물을 어떻게 분류하는가. 예를 들어, 아래와 같이 좌표 평면 위에 빨간색과 파란 색 점들이 섞여 있는 데이터셋이 있다고 하자. 인간이라면 직관적으로 화면의 한가운데를 가로지르는 가상의 직선을 긋고 이 경계선을 기준으로 데이터를 두 가지로 분류할 것이다.



데이터셋과 인간의 직관적 경계선

기계에게도 이처럼 데이터와 함께 '빨간색', '파란색'이라는 정답을 함께 주어 학습시키는 것을 '지도학습(Supervised Learning)'이라고 한다. 지도학습을 통해 기계가 인간처럼 데이터 사이의 가장 올바른 경계선(기준)을 찾아내게 하고, 새로운 데이터가 들어왔을 때 그것이 어떤 그룹에 속하는지 판정하게 하는 과정을 분류(Classification)라고 한다.

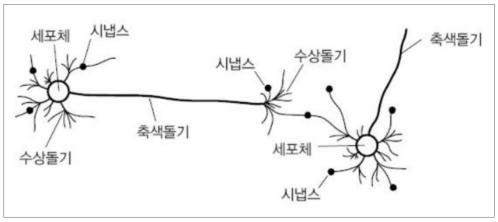
Ⅳ. 인공신경망을 통한 분류

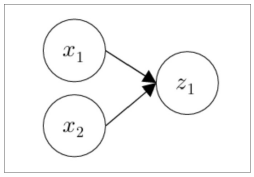
1. 인공신경망

가. 퍼셉트론과 인공 뉴런의 수학적 구조

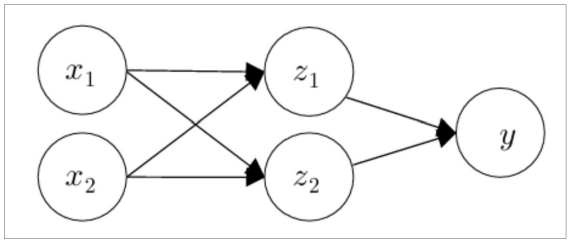
인간의 사고체계와 닮은 지능을 구현하려는 시도는 인간이 최초에 사물을 어떻게 인식하는지에서 출발하였다. 인간은 시각, 청각, 촉각 등의 다양한 감각을 신경세포를 통해 받아들인다. 이 과정에서 수많은 신경세포들은 수상돌기로부터 자극을 받아들이고 축색돌기를 통해 또 다른 신경세포에 자극을 전달하게 된다. 기계에게 이러한 작동 원리를 가르치기 위해 인간의 생물학적 신경세포를 수학적으로 모델링한 것이 바로 인공신경망(Artificial Neural Network, ANN)이다.

우리 몸의 신경계에서 자극을 전달하는 기본 단위인 신경 세포를 뉴런이라고 부르듯이, 인공신경망에서도 정보를 처리하는 최소 단위인 인공 뉴런이 존재한다. 이 인공 뉴런을 수학적으로 처음 구현해 낸 모델을 퍼셉트론(perceptron)이라 하는데, 아래는 생물학적 뉴런과 이를 모방한 퍼셉트론의 구조를 비교한 도식이다.

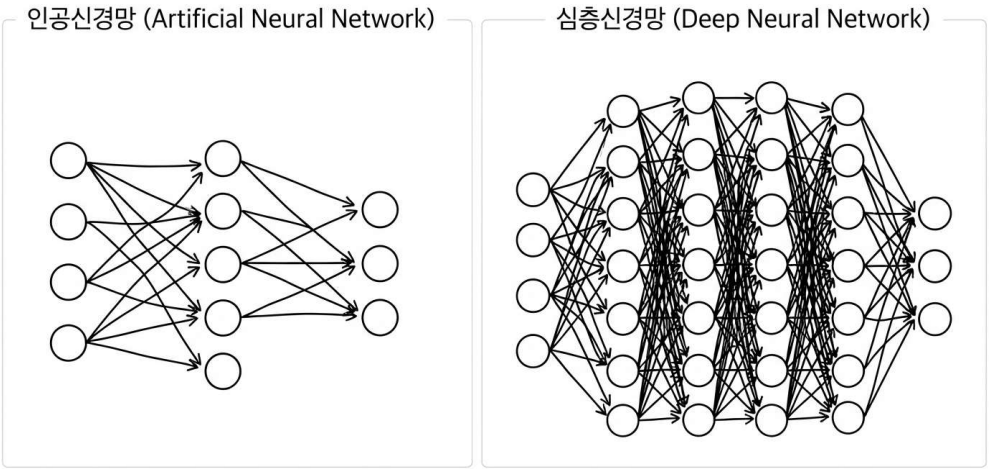




퍼셉트론은 입력층을 통해 데이터 x_1, x_2를 입력받고 출력층을 통해 결과값 z_1을 출력해 낸다. 퍼셉트론 여러개를 나열한 것을 다층 퍼셉트론(Multi-Layer Perceptron) 또는 인공신경망이라고 하며, 아래와 같이 다층 퍼셉트론은 최초로 데이터를 입력받은 퍼셉트론의 연산 결과가 다른 퍼셉트론의 입력값으로 이어지는 방식으로 조금 더 복잡한 연산을 수행할 수 있게 된다.



그리고 이 인공신경망을 여러 층으로 깊게(deep) 쌓아 올린 구조를 심층신경망(Deep Neural Network, DNN)이라고 하며, 이 심층신경망을 활용해 인공지능을 학습시키는 기술을 우리가 잘 아는 딥 러닝(Deep Learning)이라고 한다.



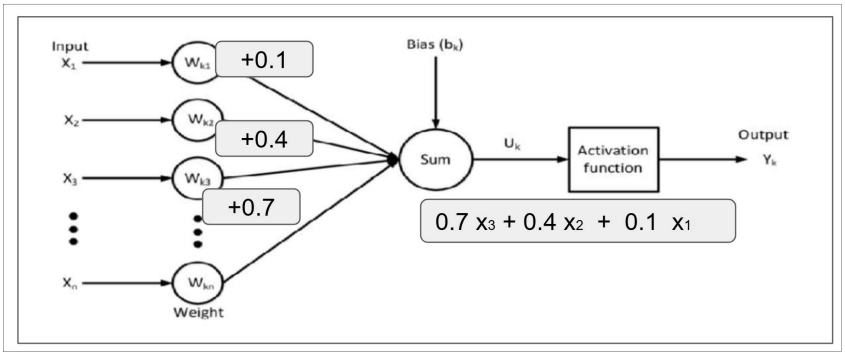
현대의 딥러닝에서는 퍼셉트론에 활성화함수를 추가한 '뉴런'을 연산의 최소 단위로 하고 있다.

나. 가중치의 수학적 의미

뉴런에 입력되는 데이터들은 각기 중요도가 다 다를 것이다. 예를 들어, 3,6,9월 모의고사 성적을 토대로 수능최저학력기준 충족 여부를 예측하는 분류 알고리즘을 인공신경망으로 구현한다고 하면, 당연히 3월 모의고사보다는 6월, 그리고 6월 모의고사보다는 9월 모의고사 점수에 더 큰 가중치를 부여하여 판단해야 할 것이다. 이처럼 각 입력데이터는 그 가중치를 반영하여 연산된다. 이 때 가중치 w_k는 곧 최적화 모델의 각 변수(파라미터)의 계수가 되기도 한다. 예를 들어, 아래와 같이 3,6,9월 모의고사 성적의 가중치가 각각 0.1, 0.4, 0.7과 같다면, 그 최적화 경계선의 방정식은

$$0.1x_1 + 0.4x_2 + 0.7x_3 + b = 0$$

꼴이 될 것이다. (x_k는 순서대로 3,6,9월 모의고사 성적. b는 오차)



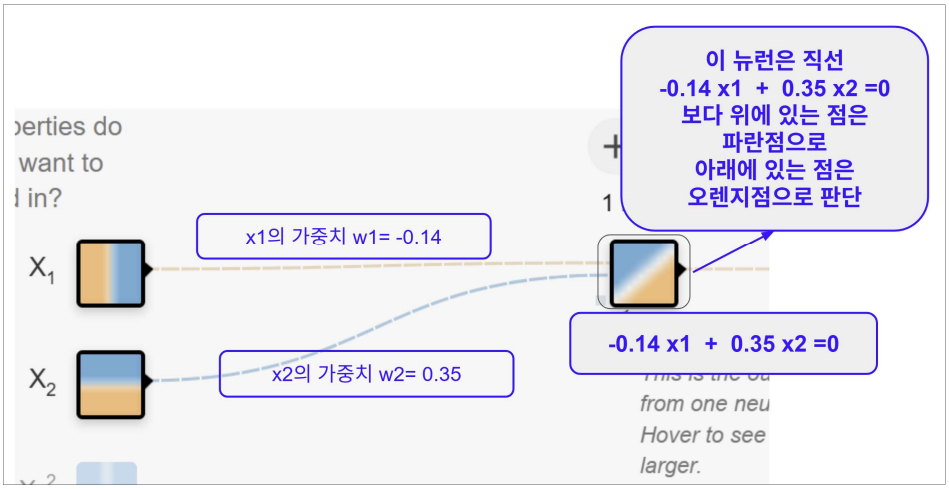
2. 인공신경망을 통한 분류 실습

가. 텐서플로우 플레이그라운드

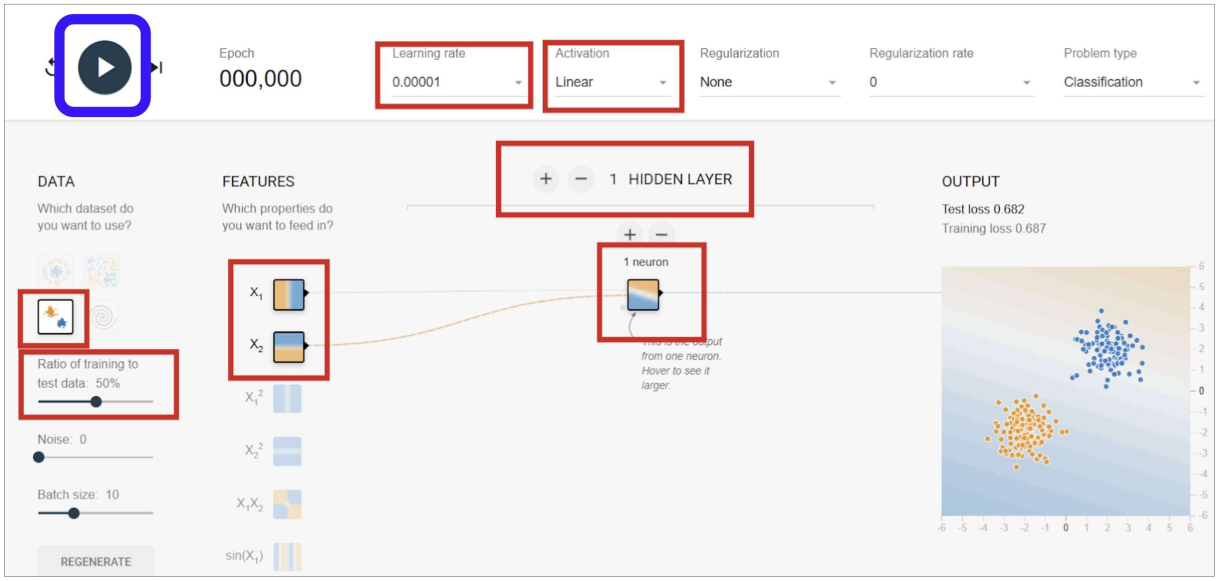
구글에서는 전문 개발자 뿐 아니라 누구나 인공지능 기술을 이해하고 활용해 볼 수 있도록 다양한 머신러닝 도구와 라이브러리를 제공해 왔다. 특히, 텐서플로우 플레이그라운드(TensorFlow Playground)는 웹 브라우저 상에서 코딩 없이 인공신경망의 동작 원리를 직관적으로 이해할 수 있도록 돕는 좋은 시각화 도구이다.

나. 가우시안 분포를 이루는 데이터셋의 분류

먼저, 가장 간단해 보이는 가우시안 분포 데이터셋을 고르고, 신경망 입력층을 통해 각 데이터(점)의 x,y좌표만 입력받고 은닉층과 그 뉴런의 개수를 1개로 하여 학습시켜 보자. 가중치의 초깃값은 랜덤으로 세팅되어 있으며 이에 따른 최초의 경계선의 방정식은 다음 그림과 같다.



플레이 버튼을 눌러 최적화를 진행하면, 손쉽게 최적의 경계선 방정식을 찾아내고 데이터셋을 올바르게 분류해 내는 것을 볼 수 있다. 이 때, 연산 과정에서 입력층에서 은닉층으로 이어지는 선들의 굵기와 색상이 끊임없이 변하는 것을 관찰할 수 있다. 이 선들이 바로 가중치(Weight)이다. 인간이 최적의 가중치를 알려주지 않아도 AI가 알아서 최적의 가중치를 찾아나가고 있다는 뜻이다. 어떻게 이런 일이 가능할까? 이는 후술할 경사하강법이라는 알고리즘을 통해서 AI가 스스로 오차를 줄여 최적의 가중치를 찾아나가는 과정이다.



다. 원형 데이터셋의 분류

한편 다음과 같이 같은 종류의 데이터들이 원형으로 분포한 데이터셋의 경우는 분류가 조금 더 어렵다.

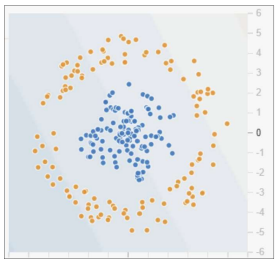


그림 27
원형의 데이터셋.

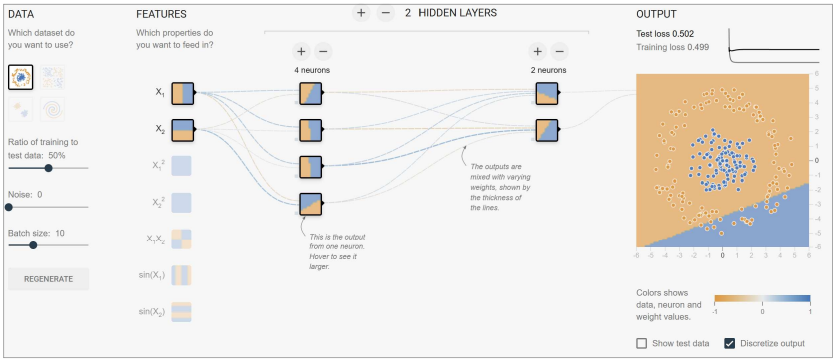
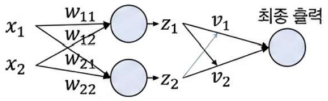


그림 28 은닉층, 뉴런의 개수를 늘려도 경계선이 직선으로 나와 데이터를 분류할 수 없다.

실제로, 가중치 w_k를 어떻게 조정해도 직선 $w_1x_1+w_2x_2=0$으로 데이터를 올바르게 분류할 수가 없으며 은닉층의 개수나, 뉴런의 개수를 늘려도 소용이 없다. 은닉층, 뉴런의 개수를 늘려도 경계선의 방정식이 계속 직선의 방정식(일차방정식)이 나오는 데는 합성함수에 관한 이해가 필요하다.



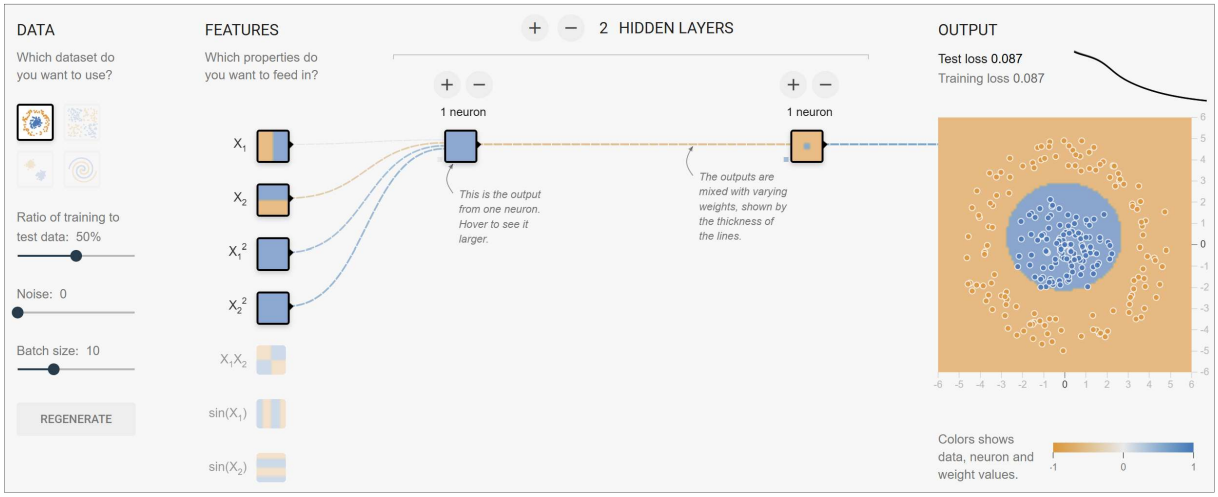
$$z_1 = w_{11}x_1 + w_{12}x_2$$

$$z_2 = w_{21}x_1 + w_{22}x_2$$

$$\therefore y = v_1 \times (w_{11}x_1 + w_{12}x_2) + v_2 \times (w_{21}x_1 + w_{22}x_2)$$

: 결국 x_1, x_2에 관한 일차식.

따라서, 입력층에서 각 데이터의 x,y좌표인 x_1, x_2뿐 아니라 ${x_1}^2, {x_2}^2$와 같은 다른 항까지 더 입력받아야 최적화가 가능하겠다는 결론을 얻는다. 실제로 아래와 같이 경계선의 방정식에 ${x_1}^2, {x_2}^2$항을 추가하면, 은닉층, 뉴런 개수를 모두 1개로 줄이고도 분류가 잘 진행됨을 볼 수 있다.

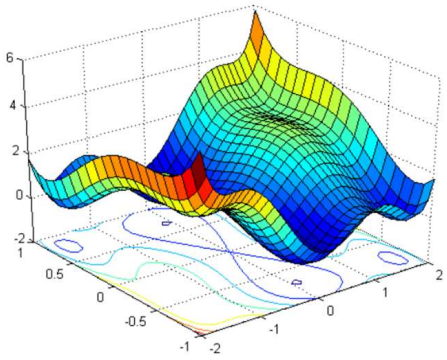


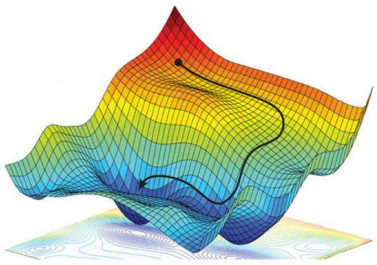
3. 경사하강법(Gradient Descent)

최초의 엉터리 가중치를 AI가 스스로 올바르게 찾아나가는 과정은 어떻게 인간의 개입 없이도 기계 스스로 해 나가는 것일까? 이는 경사하강법(Gradient Descent)이라는 알고리즘 덕분에 가능하다.

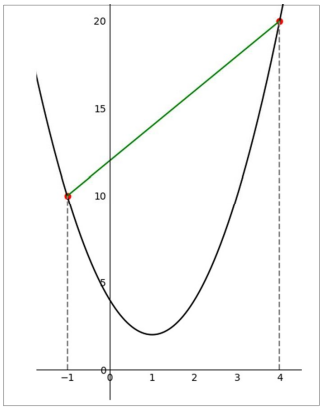
분류의 오차를 나타내는 함수를 손실함수(Loss Function)이라고 한다. 만약 변수가 한두 개 뿐인 단순한 함수라면 우리는 미분을 이용해 오차의 최솟값을 공식으로 쉽게 찾을 수 있다.

하지만 현실 속 인공지능의 손실함수는 수천, 수만 개의 변수가 얽혀 있는 아주 복잡한 식이다. 수학 공식만으로 오차가 최솟값이 되는 지점을 단번에 계산해 내는 것은 불가능에 가깝다. 바로 이럴 때, 한 번에 정답을 구하는 대신, 한 걸음씩 직접 움직이며 최솟값을 찾아가는 실용적인 알고리즘인 경사하강법이 필요하다.





←실제 다양한 손실함수

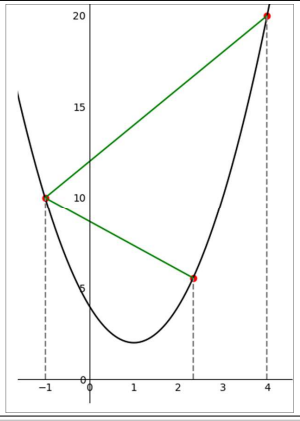


1단계) 초깃값이 $a=4$이면,

$$L'(4)=12>0$$

이므로 새로운 초깃값은 $a=4-\frac{5}{12}\times 12=-1$이다.

즉, $a=4$를 $a=4$보다 작은 $a=-1$로 이동시킨다.



2단계) 새로 설정된 초깃값 $a=-1$에 대하여,

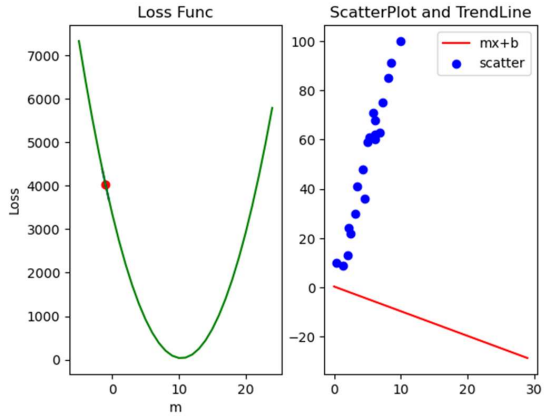
$$L'(-1)=-8<0$$

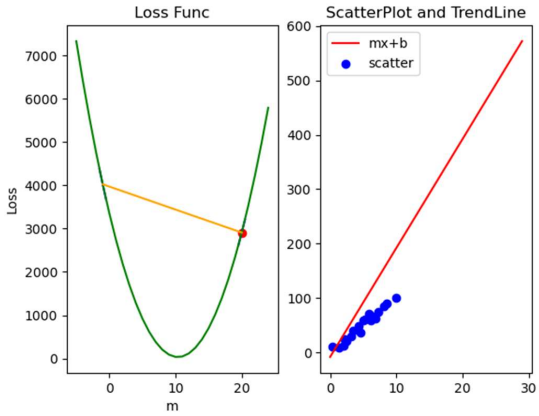
이므로 새로운 초깃값은 $a=-1-\frac{5}{12}\times(-8)=\frac{7}{3}$이다.

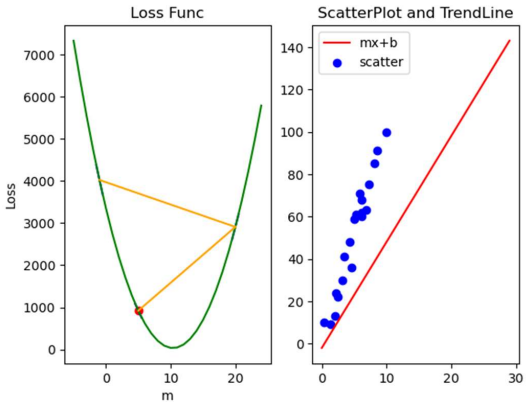
즉, $a=-1$을 $a=-1$보다 큰 $a=\frac{7}{3}$로 이동시킨다.

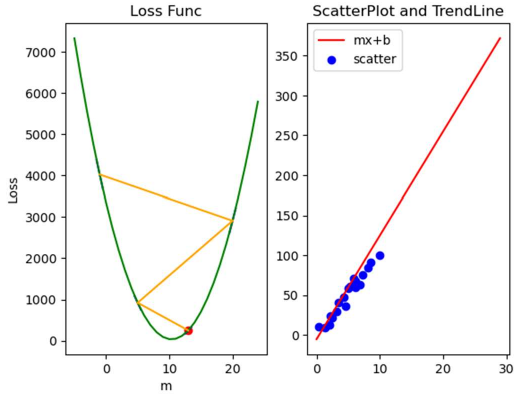
표 10 경사하강법의 단계별 안내. 초깃값(빨간 점의 x값)이 접선의 기울기 부호에 따라 이동됨을 볼 수 있다.

<경사하강법
진행에 따른
최적화 과정>









출처: https://cha-record.studio/ >> [고교수학] >> [인공지능수학] >> [4단원 교재]

Ⅴ. 결론 및 마무리

지금까지, 문제 해결 방법을 논리적, 절차적으로 제시하는 알고리즘에 대해 알아보고, AI 이전의 다양한 알고리즘과 AI 보편화 이후 특히 머신러닝에 유용하게 쓰이는 다양한 알고리즘을 알아보았다. 특히 그 중에서도 유전 알고리즘을 2D 자동차 진화시키기 시뮬레이션을 통해 알아보고 유전 알고리즘에 쓰이는 다양한 수학적 요소에 대해 알아보았다. 그리고, 인공신경망을 통한 데이터셋 분류를 텐서플로우 플레이그라운드 시뮬레이션을 통해 실습해 보고, 분류 경계 곡선이 만들어지는 수학적 원리를 알아보고 AI가 스스로 오차를 줄여 나가는 과정도 수학을 통해 이해하였다.

인공지능이라는 화려한 기술도 결국 그 껍질을 벗겨 보면 수학적 논리와 잘 짜여진 알고리즘으로 구성되어 있다. 수업 시간에 마주하는 다양한 수학적 개념과 원리는 단순히 시험을 치르기 위한 도구가 아니라, 상상 속의 아이디어를 현실로 구현해 내고 삶의 질을 향상시키는 미래 기술의 뿌리로서 가치가 있다.

차민경
(https://cha-record.studio/,
chamingyung1@gmail.com)

참고문헌

[1] 교육부, 2022 개정 수학과 교육과정, 교육부 고시 제2022-33호
[2] Rednuht, 2D Car Evolution, https://rednuht.github.io/car-html5/
[3] Google, TensorFlow Playground, https://playground.tensorflow.org/
[4] 진강규, 유전알고리즘과 그 응용, 교우사, 2000
[5] 이건길 외, 고등학교 인공지능 수학, 미래엔, 2021