머신러닝 (machine learning)

Input + Output 제공 : 중간과정 (process)을 혼자 파악해내는 것

- 지도학습 : 레이블이 있는 데이터로 학습 → 문제의 정답과 문제 자체를 전부 제공
- 비지도학습 : 레이블이 없는 데이터에서 패턴을 파악 → 문제의 정답은 알려주지 않음 (현재 사용도 : 지도학습 > 비지도학습)
- 강화학습 : 행동에 대한 보상으로 최적의 정책을 파악 → 보상을 극대화하도록 계속 실행

지도학습 :
- 이진 분류 : 데이터를 둘 중 하나의 클래스로 분류
- 다중 분류 : " 셋 중 "
- 회귀 : 변수(input) 출력 값 (output) 간의 관계를 모델링 → ex) 키가 큰 사람과 그 사람의 몸무게의 상관관계를 파악

비지도 학습
- 군집화 → 데이터를 어떤 방법을 이용하여 묶음으로 만드는가
- 차원축소 → 고차원에서 저차원으로 바꾸어 학습에 이용 → 이 과정에서 사용하는 방법이 다양
  - 데이터를 → 차원 축소

군집화
- k-평균 → 거리함수 사용 & k개의 군집으로 나누는 알고리즘
- 계층적 군집화
- DBSCAN → 밀도 기반의 군집화
  - 밀도↑ : 군집으로 인정 / 밀도↓ : 군집으로 인정 X
- GMM → 가우시안 혼합모델로 군집화

차원 축소
- PCA : n차원을 n-k차원으로 차원 축소를 하고 군집화
- t-SNE
- UMA

* 에피소드 : 상태 - 행동 - 보상의 순서를 의미한다

강화학습
- 몬테카를로 : 에피소드를 이용한 학습
  - ↳ 이동 후에 경험에 따라 행동을 수정
- 시간차 학습 : SARSA를 이용
  - ↳ 현재 상태와 다음 상태의 가치를 비교하여 행동.
- Q-learning
  - ex) left or right : left = +10, right = +9 ∴ left
  - ↓ 결과 보상을 최적화한다.

딥러닝

인공신경망을 이용하여 패턴을 학습하고 인식

└→ 은닉층이 많은 심층신경망을 사용

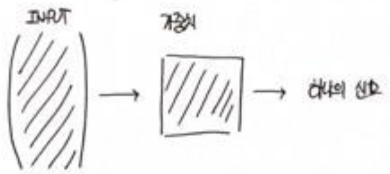
(이미지 인식
음성 인식
자연어 처리
추천시스템) 딥러닝의 예

** 신경세포

여러 신호 → 하나의 신호
IN OUT

before 출력 : 활성함수를 이용하여 내보낼 신호를 가질 수 있다

활성 함수 : INPUT을 적절한 신호로 변환



***** 인공신경망 (Artificial Neural Network , ANN)

· 데이터를 처리하고 학습하는 데 사용

· INPUT , output , 그리고 그 사이에 여러개의 layer (은닉층)이 존재

· Input & weight (가중치) ⟶ output

· layer에서 활성화함수를 이용하여 최적의 weight & Bias 결정이 목표

** 인공신경망의 단점

· 최적의 weight & Bias 결정이 어려움 .

· Overfitting : 결함 새로운 데이터를 학습하는 데 시간 ↑ & 어려워짐

*** DNN (심층신경망 Deep ")

· ANN에서 layer를 증가 : 결과 향상

· 은닉층이 2개 이상 , 스스로 분류하여 같은 의미 & 분류 레이블 지정

⟶ ** DNN의 한계

· 고해상도 이미지 ⇒ 뉴런의 개수 ↑ ⇒ 파라미터 개수↑ (INPUT)

· DNN : 목적 패턴까지의 상관관계를 파악하지 못함

** CNN (convolution " , 합성곱 신경망)

· DNN 중 일부 (하나)

* DNN vs CNN

사진 비교 → DNN

└→ 모든 부분을 확인

└→ INPUT(파라미터) 개수 ↓

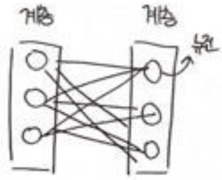
→ CNN

└→ 특정 부분의 특징을 필터로 학습한다 → input(파라미터) 개수↓

** 전결합 계층

계층 : 뉴런이 모인 단위

전결합 계층 : 모든 뉴런이 서로 연결



Bias : 가중치를 조절하여 계산하는 과정시 조절 장치

** 일반 신경망

· 입력층, 은닉층, 출력층이 전부 하나

· 은닉계층 & 출력 계층 : Full-connected

· 복잡한 문제 해결은 X but 처리 시간 & 비용 ↓

** 회귀

학습 샘플(데이터) → 규칙을 찾아 → 결과를 예측

예) 키 160? → 몸무게 예측

** 분류

입력값을 분석 → 특정 category에 배정 (결론)

└→ 이진 분류 : yes or NO

다중 분류 : 어떤 유형?

↓

One-Hot coding / Sparse Vector (희소 벡터)

(only one = 1, except = 0) (one : 1, except : 0) Just 참고 사항

** 뉴런의 수학적 표현

$x_0 \rightarrow w_0 \rightarrow$

$x_1 \rightarrow w_1 \rightarrow$ (Σ) → 출력

$x_2 \rightarrow w_2 \rightarrow$

$x_3 \rightarrow w_3 \rightarrow$

편향을 통해 더 다양한 그래프 표현 가능 (원점을 안 지나는 직선을 표현 가능)

$\therefore k \sum_{n=0}^{3} x_n w_n + b$ (b → 편향 bias)

**** 활성화 함수

- 쉽게 말하면 (파라미터로) 다음 계층으로 전달신호의 양을 결정
- 어떤 인공신경망에서 output을 받아 다음 인공신경망이 받을 수 있는 Input으로 변환

---- 활성화 함수의 종류 ----

*** 시그모이드 함수

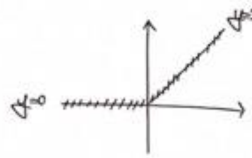
- Input을 0과 1사이의 값으로 변환
- $\sigma(x) = \frac{1}{1+e^{x}}$
- 0보다 클수록 1에 가깝게, 0보다 작을 수록 0에 가깝게
- Input의 크기가 너무 크면 학습이 어려움

*** Hyperbolic tangent (tanhx)

$-1 \le \tanh x \le 1$

- 시그모이드는 0에 수렴하는 값은 잘 학습 x
  - ↳ tanhx는 그 한계를 극복
- but 여전히 크기가 ±1에 수렴하면 학습 ↓
  also, 계산 비용 ↑

*** 렐루 (ReLU) 함수, Rectified Linear Unit



시그모이드 & tanhx 보다 처리비용 ↓, 처리속도 ↑

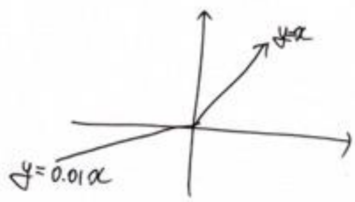
1보다 큰수 출력 가능!

- 단점 : 입력 <0 : all 0임 → 단점
  : 입력 ≈ ∞ , 발산을 초래할 수 있다.
  (폭발적 활성화)

performance 순서 : ReLU > tanhx > 시그모이드

Leaky ReLU

음수값을 아예 0으로 처리하지는 않음



출력함수 : softmax 함수

- 출력층에서 사용하는 함수
- 출력층의 값을 정규화하기 위해 Σ출력층 =1 이 되도록 조정
- 벡터 형태로 Input을 받음

40 → 0.4
60 → 0.6) Σ=1

softmax

< 인공 신경망 >

INPUT → 활성화 함수 → 결과값 → 여기에서 성능 파악

< 인공신경망 오차 구하기 >

** 이진 교차 엔트로피 (BCE)   cf) 엔트로피 = 무질서를 측정하는 개념 (or 불확실성)

- 예측을 잘하면 오차 = 0
- 예측을 잘 못하면 오차값은 매우 크게

(실제 레이블, 예측확률) → 이진 오차 엔트로피로 계산
BCE 라는 수식

** 평균 제곱오차 (MSE)

(실제값, 예측값) → MSE : MSE↑ : 오차↑ ⟹ 다시 학습해야 함
수식

< 인공신경망 오차 줄이기 >

** Back propagation (오차 역전파법)

마지막 부터 처음까지 되돌아가면서 경사하강법을 이용하여 가중치를 수정

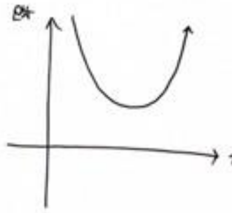
인공지능이 학습한다 : 오차를 줄여나간다 (가중치를 수정한다)

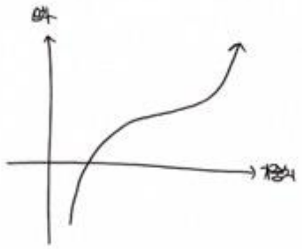
가중치를 얼마나 이동? : optimizer가 결정

< 경사 하강법 >

- 기울기를 사용하여 가중치를 변경
- 가중치가 인공지능의 성능을 결정

ex)





→ 오차가 가장 작은 위치로 이동해야함
주로 기울기 ↓ = 오차 minimum

< optimizer >

- 경사하강법을 위한 학습 Tool 이라고 볼 수 있기
- 다양한 방법으로 기울기 min을 찾아 내는

(은닉층 : ReLU
출력층 : 시그모이드 or 소프트맥스)

⇓

가장 일반적 → 은닉층 ReLU가 요즘 Best임

각 함수를 어느 계층에서 사용하냐에 따라 인공지능의 성능이 달라짐 ⌋ → 여기까지가 우선 인공지능의 과정이라고 보면 됨

***** 합성곱 신경망 (CNN)

* 이미지 데이터 (CNN이 이미지 데이터 표현에 매우 유용)

· 픽셀 : 이미지 표현 기본 단위

픽셀 하나를 하나의 칸으로 보고 " 128 × 128 × 1 " 같이 표현

⇓

마지막에 붙은 숫자는 채널을 의미함

channel : 이미지 하나가 몇 층으로 구성되어 있는지를 나타냄

* 다층 이미지 처리 구조

↳ 입력층은 일렬 형태

↓

다층 이미지 처리 구조 (MLP) 단점

(3×3 행렬 → 9×1 행렬) 과정에서 Error

⇓

비슷한 이미지도 못 알아봄

∴ (CNN : 일부를 확인, not 전체
↳ 필터) : CNN의 핵심

** CNN 구조

· 커널 : 학습의 대상이 되는 가중치

· 합성곱 : 필터를 이용하여 이미지의 특징을 추출

· 모으기 : 합성곱을 이용하여 이미지를 단순화

합성 곱 계산 절차

$$\begin{bmatrix} x_1 & x_2 & x_3 \\ x_4 & x_5 & x_6 \\ x_7 & x_8 & x_9 \end{bmatrix} \times \begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix} \rightarrow \begin{bmatrix} x_1w_1 & x_2w_2 & x_3w_3 \\ x_4w_4 & x_5w_5 & x_6w_6 \\ x_7w_7 & x_8w_8 & x_9w_9 \end{bmatrix} \rightarrow sum = y = \sum_{k=1}^{k=9} x_k w_k$$

data　　　　filter　　　　Feature Map

- 5x5 데이터 & 3x3 filter ⟹ 2x2 Feature Map
  (입력 데이터)　　　　(출력 이미지)
  (= 입력 이미지)
- 데이터와 필터의 행렬 크기에 따라 feature map의 크기가 결정됨

⇓

** but feature map (출력 이미지)는 계속 행렬의 크기가 줄어들게 돼 있음

⇓

padding! ⟷ Pooling Layer

*** padding : 입력이미지를 "0"으로 감싼다

⇓

입력이미지 & 출력이미지의 행렬 크기 유지

↳ 합성 곱 절차를 거쳐서 생긴 출력 이미지의 행렬을 줄일 때 (행렬의 크기를 작게) 사용

출력 이미지의 크기를 구하는 공식도 있다 (이번 생략)

***** Classifier (분류기)

- 추출된 이미지를 바탕으로 입력 이미지 분류
- 전결합 계층과 출력층으로 이루어져 있음

↳ 결국 입력 받은 값을 소프트맥스 함수 (0~1)로 출력

⇓

가장 높은 값 : 정답

***** 전체과정

* 1. 정사각형으로 추출
↓
2. 합성 곱 계산
↓
3. pooling
↓
4. padding & pooling ----
↓
5. classifier
6. 결과 확인

결과에 따라 filter를 수정할 수 있다

⇓

인공지능 성능 결정

Numpy 에서 사용하는 데이터 구조

- 스칼라 : 한개의 숫자 데이터
- 벡터 : 스칼라의 집합, 여러개의 숫자가 특정 순서대로 모여 있는 1차원 배열
- 행렬 : 벡터의 집합 – 1개 이상의 행과 열로 구성된 2차원 배열
- 텐서 (Tensor) : 행렬의 집합, 3차원 이상의 배열

*** axis = 0 : 세로축 (행을 따라 연산) (세로)

axis = 1 : 가로축 (열을 따라 연산) (가로)

** Import numpy as np

arrange ( ) : 배열의 요소의 범위를 지정하면 지정된 범위 내의 정수값들을 생성
→ (1,6) → 1 ~ 5

zeros (행,열) : 행x열 배열에 0으로 채우면 되는 경우 사용

ones (") : " 1 으로만 "

random. rand (") : " 랜덤으로 "

cf) 행렬 간의 사칙연산은 그냥 가능

행렬에서 사용되는 함수 : sum( ), average( ), median, max( ), var( ), std( )

sum (행렬, option) → 전체를 더 하는 것인지 / 행만 더하는 것인지 / 열만 더하는 것인지 를 결정
⇓
axis = 1, 0 으로

sum, average, median, var, std : 전부 가능 !

median : 중앙값

np.median : 행렬의 중앙값을 찾아

↓

중앙값 x → 정렬의 양쪽 평균 = 중앙값

max( )
min( ) : 행렬에서 최대 / 최소 찾아

* argmax( ) : 행렬에서 최대값의 인덱스를 알려줌

var( ) : 분산

std( ) : 표준편차

any( ) : 배열의 요소 하나라도 ~

all( ) : 배열의 요소 전부 ~

where( ) : 조건이 참일 때 실행될 값과 거짓일 때 실행될 값이 다름

↓

결과, True, False

**** Pandas → 파이썬의 라이브러리다 !

· 데이터 분석 + 전처리에 사용

* Pandas는 세가지 자료구조 제공

1. Series : 1차원 자료구조
2. Data Frame : 2차원 자료구조
3. Panel : 3차원 자료구조

** import Pandas as pd

Series : 배열 / 리스트 이용

Data Frame : 테이블 데이터 (Tabular data) 이용
(= 행과 열이 있는)

Panel :

< Series >

S = pd. Series (data)

If data = [1,3,5,7,9]

output :

0	1
1	3
2	5
3	7
4	9

→ 배열 데이터들 옆 행번호 같이 출력

< Data Frame >

df = pd. DataFrame (data)

out

col 열방향으로 출력.

0
1
2
3

row

** 크롤링 (crawling)

· 웹 상의 다양한 정보 자원을 자동화된 방법으로 수집해서 분류 및 저장하는 것

* 웹 크롤링

· html의 style 이나 특정 selector를 골라오도록 프로그래밍된 프로그램을 이용하여 웹 사이트에서 html을 기준으로 특정하게 나뉘어져 있는 부분을 긁어온다 → 특정 태그, 클래스, div, li .... 같은 html 구조에서 긁어온다

** 워드 클라우드 (= 태그 클라우드)

· 각 태그들 중 중요도나 인기도가 높은 순서대로 태그들의 크기나 색상을 조절하여 시각적으로 나타낸 것

⇓

쉽게 말하면 글자(단어)의 크기 & 사이즈 & 색상 & 모양을 조절하여 시각적으로 나타낸 것

↓

이 기준이 주로 인기도, 중요도

* sklearn

· 데이터를 학습할 때 사용 (DL)

· 데이터를 전처리할 때 사용

· 성능 평가 및 정규화 기능 (벡터화도 전처리의 일종)

In preprocessing → 다양한 형태와 종류의 정보들을 분석하기 쉽도록 전환

(· 결측치 제거 , 중복 제거 , 특수문자 제거 , 정규화 , 자연어 처리
이미지 명암화 , 벡터화 , 토큰화)

↳ 이것들이 다 preprocessing 들

문서를 예를 들면, 단어의 개수의 빈도를 전체 문서를 기준으로 비교하는 것 같은 것

궁금하면 TF - IDF 벡터화 찾아보기

* Logistic Regression

Regression 중 하나로 , 통계학이나 ML에서 이진 분류 문제를 해결하는 데 사용

Input을 두개의 category 중 하나로 예측하는 데 자주 사용

* Tensorflow

DL / ML을 위한 라이브러리

* keras

Tensorflow를 돕는 Tool (고수준의 API)

* 전이 학습 (Transfer Learning)

· 이미 학습이 된 ML / DL 에서 추가로 내가 데이터를 넣어서 부분적으로 학습시키는 것

· 정확도가 꽤 높음

* MNIST

- 필기체 분류를 위한 Database

· 성능 구리다

* Train data and Test data ·

↓ 이걸로 학습시키고

↓ 이걸로 가중치 설정하는듯?

아닐 수도

「Numpy, Pandas matplotlib」
간단히 정리하는 ver.

• 1차원 배열 출력 pd.series(data)
• 2차원 배열 출력 pd.DataFrame(data)

df.describe()
→ 기초 통계 자료 제시

cf) Teachable machine
* 이미지 → 이미지 분류 모델
→ 압축 풀어서 프로그램 사용 가능

cf) .csv 확장자
→ 이름 바꿔야 프로그램에서 열림

Pandas : 데이터분석 (정리)
Matplotlib : 데이터 시각화
< 캔버스 생성 – 색깔 선택 – 선 or 막대 결정 등 >
ex) plt.bar : 막대 그래프 생성

→ 이 코드는 주로 고정되어있음
→ (라이브러리) matplotlib에 있는 여러 모듈을 이용하여 그래프를 생성

「Wordcloud 라이브러리 정리」
문장 / 문서 / 다량의 데이터들을 빈도수나 연관된 단어들과의 관계에 따라 크기를 다르게 하여 표현

* opencv : 인공지능할 때 사용하는 모듈
• 얼굴 / 눈·코·입을 인식하는 구분 가능

* konlpy
→ korean natural language process python module.
* 한국어 전처리용 파이썬 모듈

* request & bs4 : 크롤링할 때 사용하는 모듈
• 크롤링이란 웹페이지에서 html·css의 문법을 따라 그곳에 있는 내용을 긁어오는 것
최근 사이트 – 크롤링 안되도록, 막아놓은 경우가 많다

* qrcode : QR코드 생성 모듈
* PIL : Python Image Library 파이썬 이미지 처리에 사용하는 모듈