목차

교육 일정

머신러닝을 활용한 나노/탄소 소재 생산공정 최적화

06 AI 기법 성능 향상 방법론

목차

AI 기법 성능 향상 방법론

EliceAcademy

01

Hyper-parameter 튜닝

파라미터와 하이퍼 파라미터

- 파라미터 (Parameter)
  - 매개변수
  - 모델 내부의 값
  - 데이터를 통해 학습 됨
  - *e.g.*, MLP의 weight

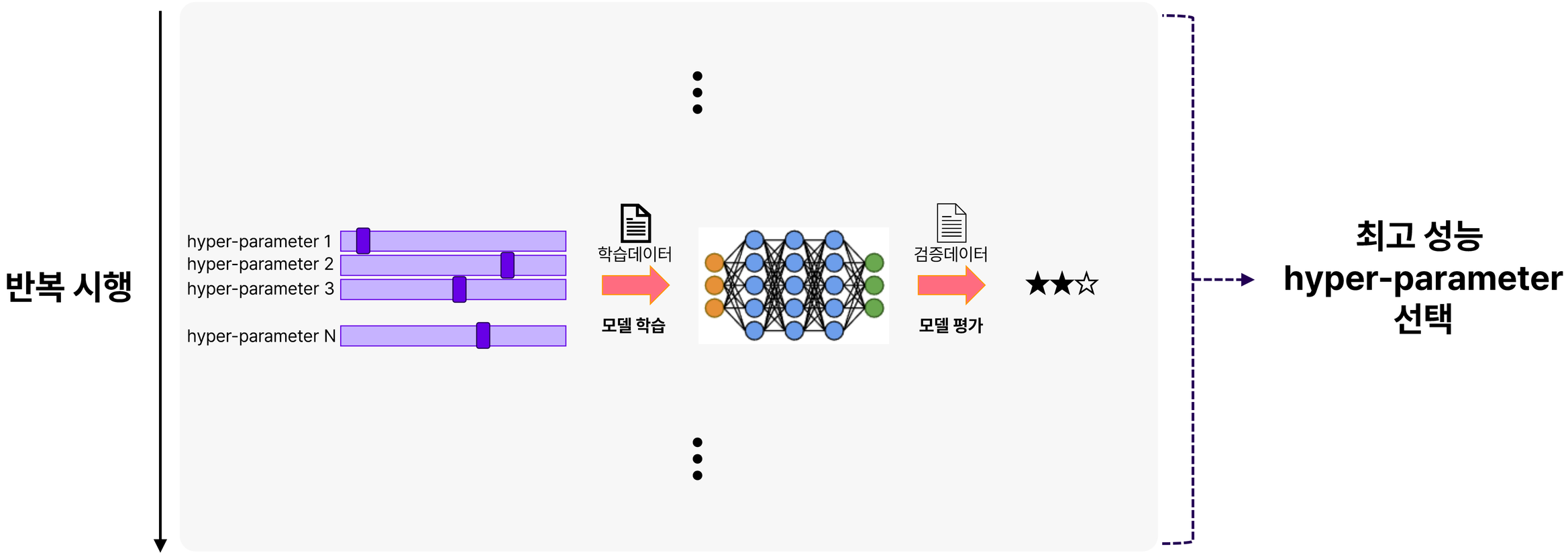
- 하이퍼 파라미터
  - 사용자가 직접 지정해주는 값
  - 데이터를 통해 학습되지 않음
  - *e.g.*, 학습률, Loss fuction, 배치사이즈, 에폭 수, 은닉층 개수, ...

Hyper-parameter 결정 방법

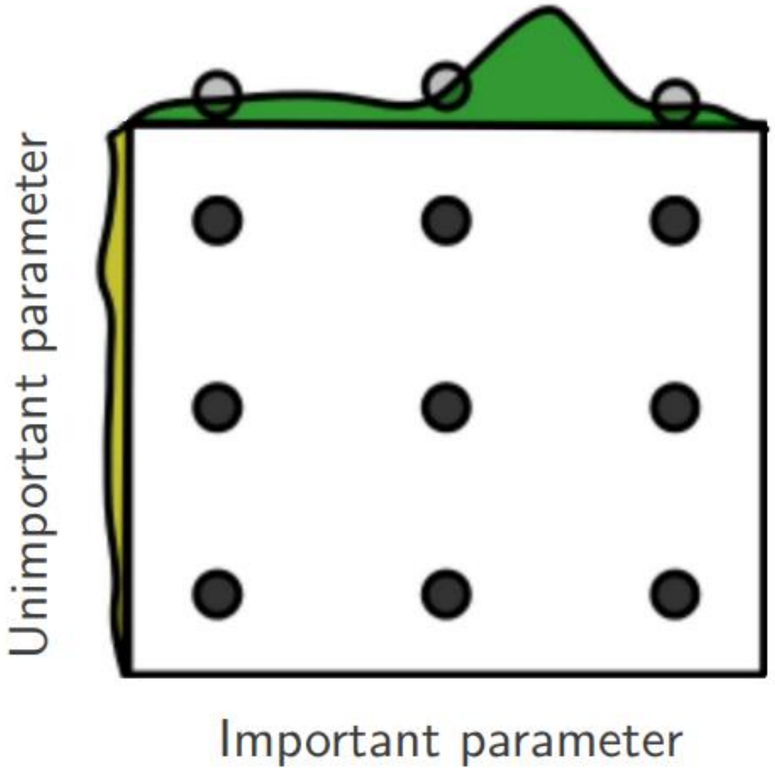
- Hyper-parameter tuning
- 경험을 통해 최적의 값을 찾음
  - 다항 회귀 분석에서 어떤 피쳐들을 사용해야 할까? ($X^2, X^3, \dots$)
  - 어떤 정규화 방법을 사용해야 할까? (Dropout, 데이터 증강, 앙상블, ...)
  - 어떤 최적화 기법을 사용해야 할까? (SGD, Adam, Adagrad, ...)
  - MLP 의 유닛수와 레이어 수를 몇으로 해야 할까?
  - 학습률을 어떻게 설정해야 할까?
- 최적의 hyper parameter 을 바로 알 수 있는 방법은 없음!
- 그럼에도, hyper parameter tuning 은 매우 중요!
  - Grid search
  - Random search

EliceAcademy

Hyper-parameter tuning



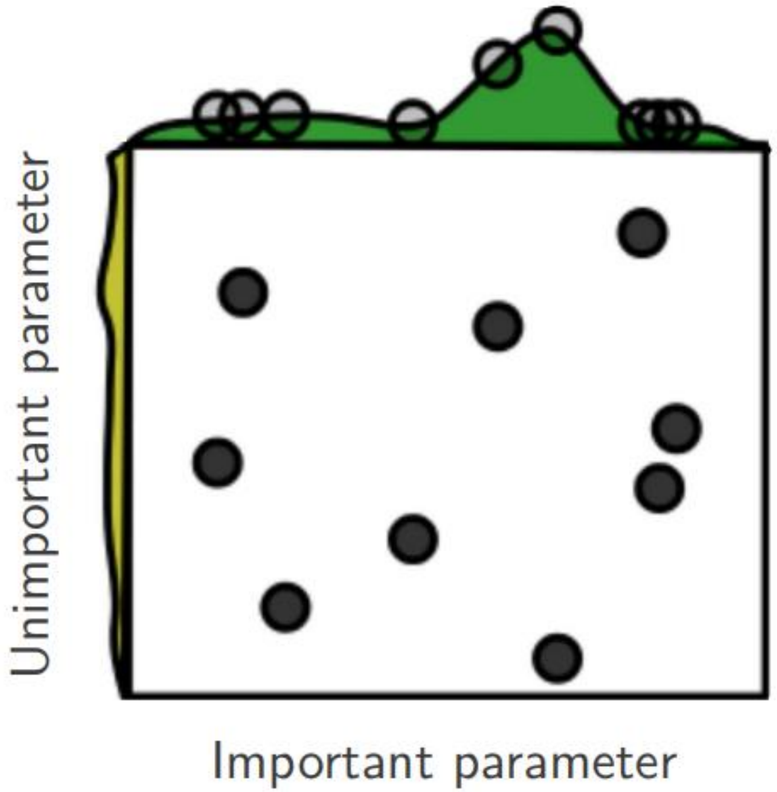
Grid search



- 각각의 hyper-parameter 들의 범위 설정
- Hyper-parameter 들을 일정한 그리드로 나누어 평가
- **비효율적**
  - 중요한 변수와 중요하지 않은 변수가 있을 수 있음
  - 각 변수 당 여러가지 값을 테스트하기 어려워 최적의 값을 찾지 못할 확률이 높음

Random Search for Hyper-Parameter Optimization, Bergstra et al.

Random search



- 각각의 hyper-parameter 들의 범위 설정
- Hyper-parameter 들을 범위 내에 임의로 샘플링
- 각 변수 당 여러 값을 테스트하여 최적의 값을 찾기 수월함

Random Search for Hyper-Parameter Optimization, Bergstra et al.

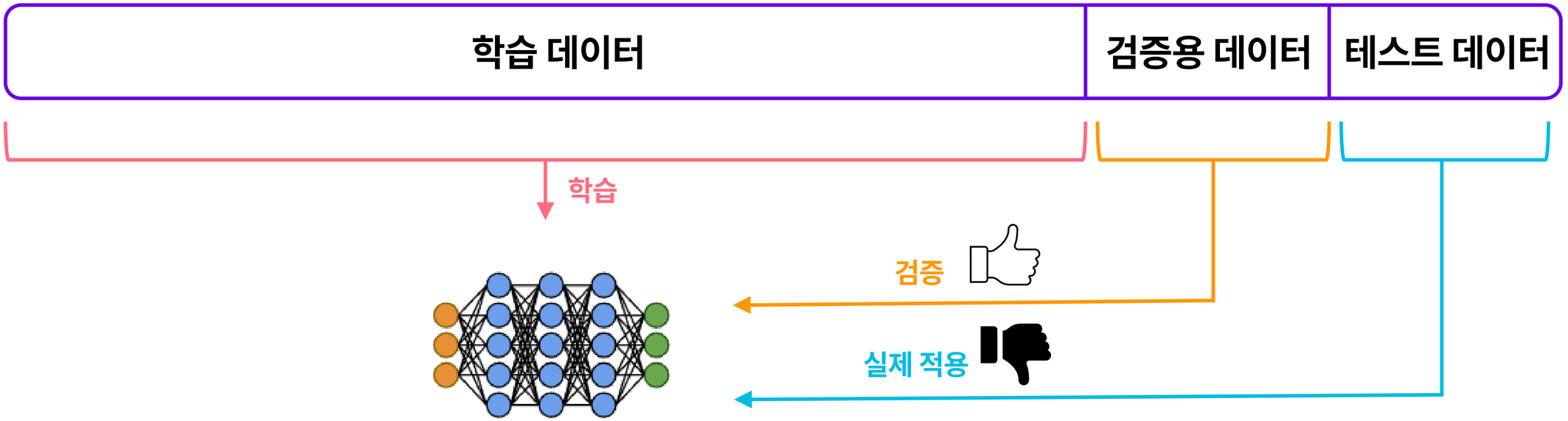
02

교차 검증

데이터셋 분할 recap

- 학습 데이터
  - 전체 데이터 중 대부분은 모델 학습에 사용
- 검증용 데이터
  - 전체 데이터 중 일부는 검증에 사용
  - 검증용 데이터에 대한 오차가 증가하면 과적합
- 테스트 데이터
  - 전체 데이터 중 일부는 테스트에 사용
  - 최종 모델 성능 평가 목적

데이터셋 분할 recap



- 검증용 데이터에 운이 좋게 잘 적용된 모델일 수도 있음!
- 즉, 검증용 데이터에 과적합 되었을 수 있음

EliceAcademy

K-fold 교차 검증 (K-fold Cross validation)

- K = 5

교차검증의 장점과 단점

- 장점
  - 모든 데이터셋을 평가에 사용함으로써, 특정 테스트 셋에 과적합 되는 것을 방지

- 단점
  - K 개의 모델을 학습하여야 하므로, 모델 학습 시간이 오래 걸림

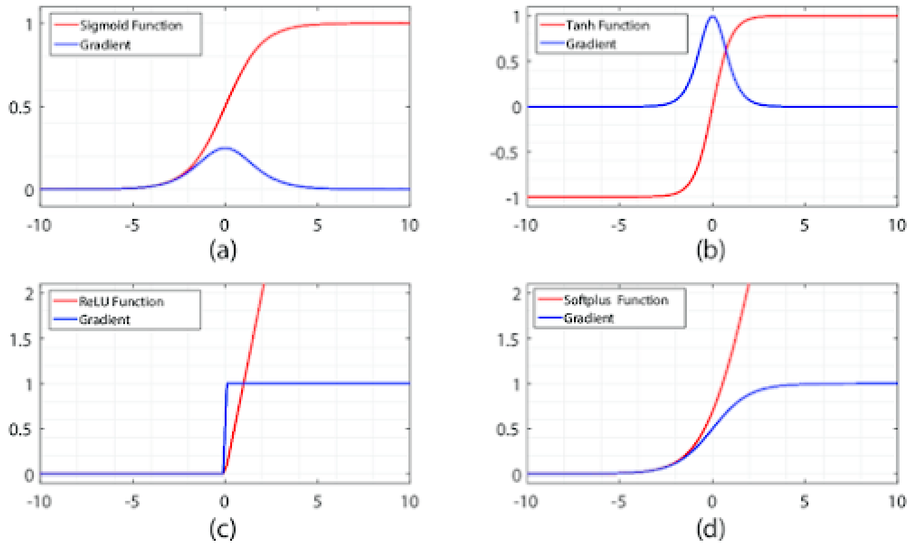
EliceAcademy

03

Residual Network

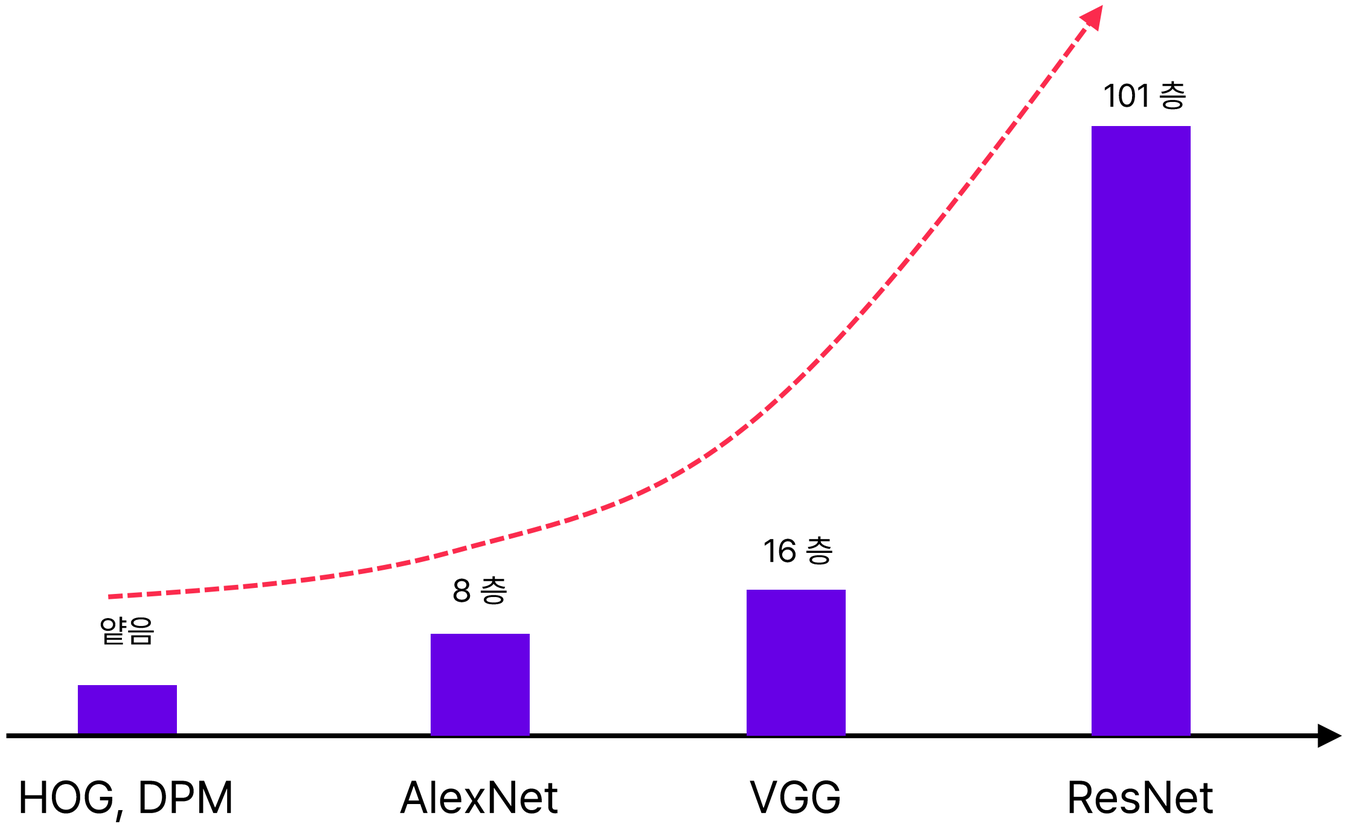
Gradient Vanishing 문제 recap

- Sigmoid 와 Tanh 함수는 입력값의 절대값이 커질수록 gradient 가 0으로 수렴
- ReLU 와 Softplus 등의 활성함수로 어느정도 완화할 수 있지만, **MLP가 깊어지면 여전히 gradient 가 0으로 수렴**

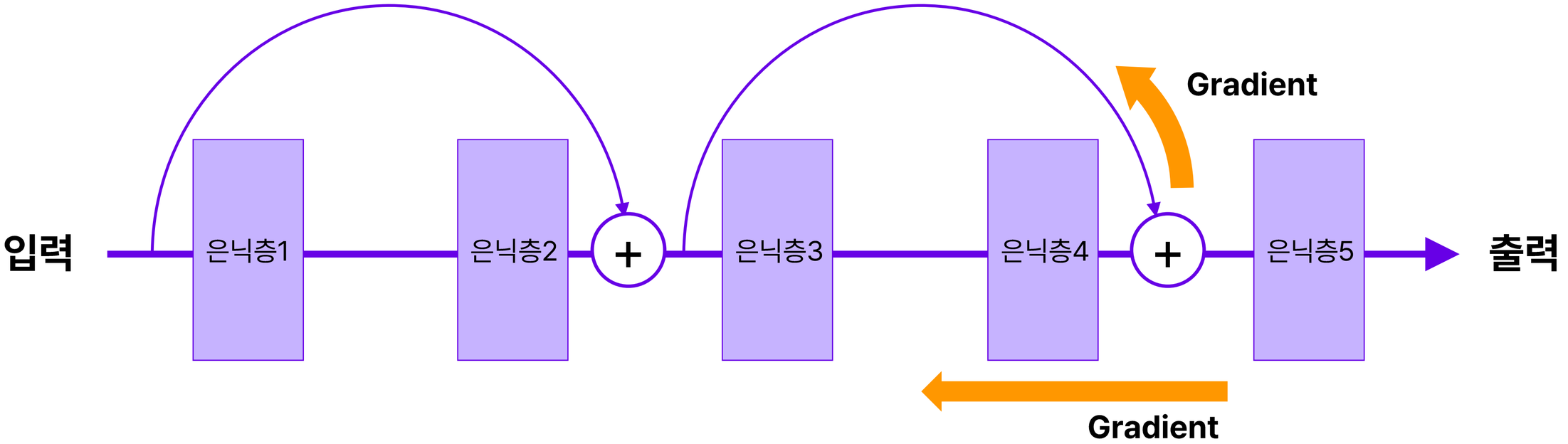


Ranking to Learn and Learning to Rank: On the Role of Ranking in Pattern Recognition Applications, 2017

딥러닝 네트워크의 깊이



Residual network



Residual network

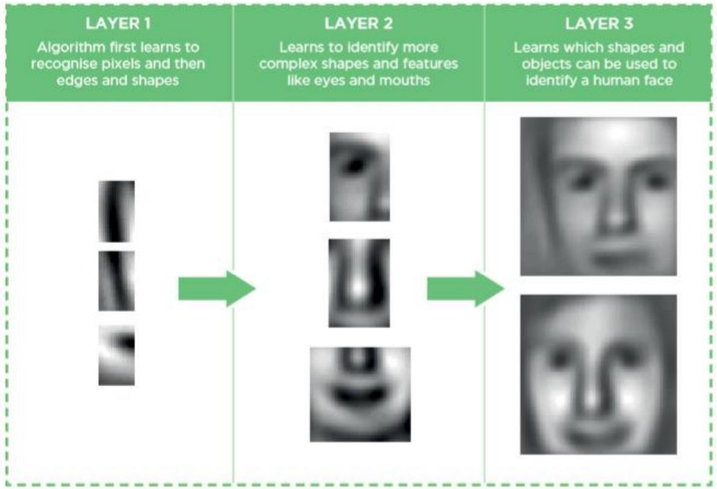
- 순전파
  - $X_{l+1} = X_l + F_l(X_l)$
  - $X_{l+2} = X_{l+1} + F_{l+1}(X_{l+1}) = X_l + F_l(X_l) + F_{l+1}(X_{l+1})$
  - ...
  - $X_L = X_l + \sum F_i(X_i)$
- 역전파
  - $\frac{\partial Loss}{\partial X_l} = \frac{\partial Loss}{\partial X_L}\frac{\partial X_L}{\partial X_l} = \frac{\partial Loss}{\partial X_L}\left(1 + \frac{\partial}{\partial X_l}\sum F_i(X_i)\right)$

04

Positional Encoding

인코딩

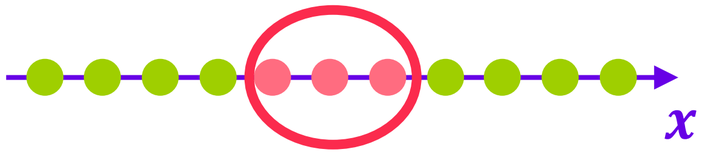
- 저차원의 데이터를 높은 차원의 데이터로 매핑하는 것
- 전통적인 머신러닝에서 적절한 피쳐를 찾는 것은 굉장히 중요
- 딥러닝 방법론의 장점은 이러한 피쳐를 스스로 찾는다는 것!



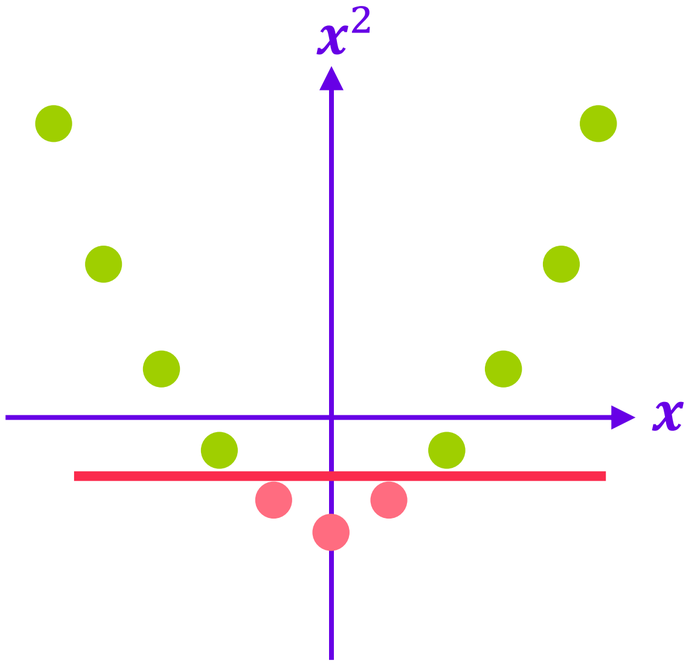
인코딩

- 입력데이터를 고차원 공간에 매핑함으로써, 이후 뉴럴넷이 효율적으로 피쳐를 찾을 수 있다.
- 인코딩의 예시
  - Kernel trick
  - One-hot encoding
  - Frequency encoding

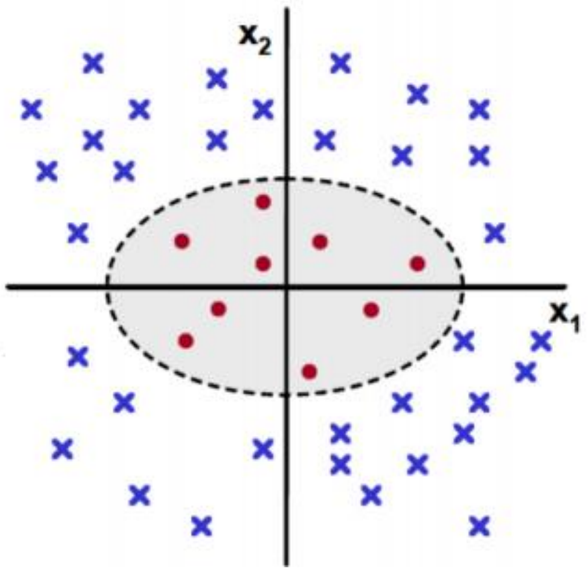
Kernel trick



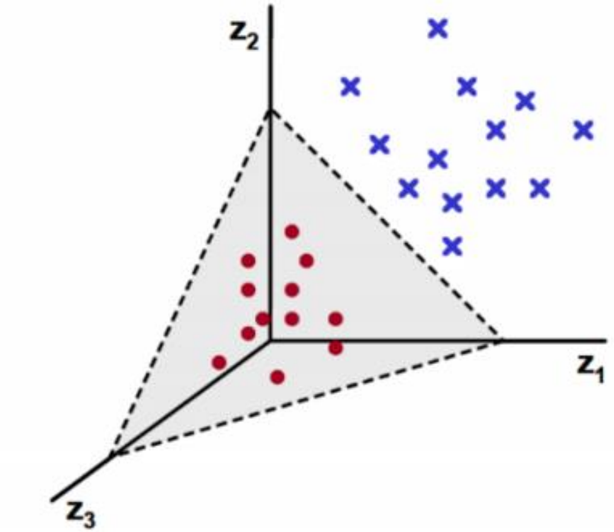
$$\Phi(x) = \{x, x^2\}$$



Kernel trick

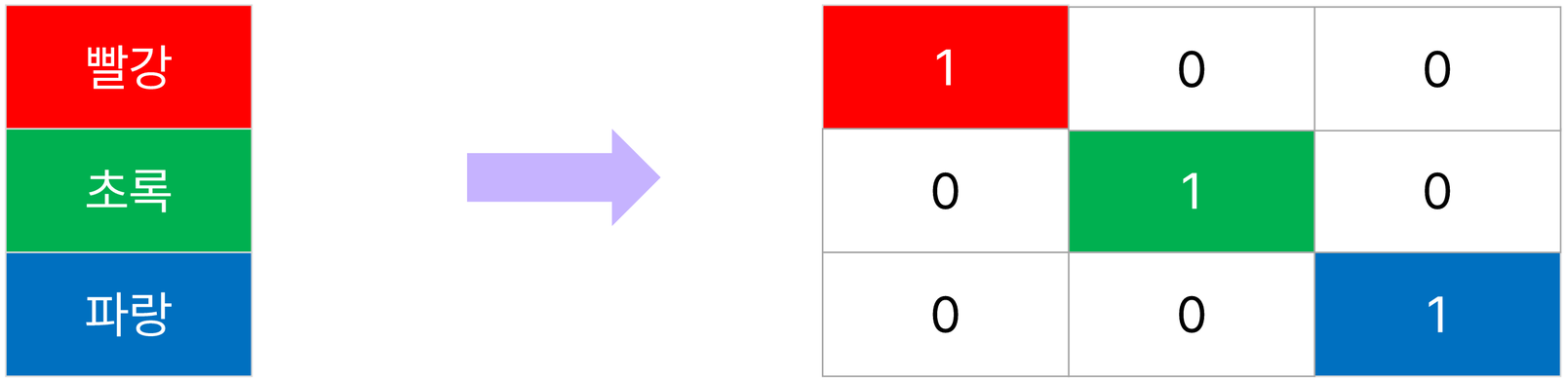


$$\Phi(x_1, x_2) = \{x_1^2, \sqrt{2}x_1x_2, x_2^2\}$$



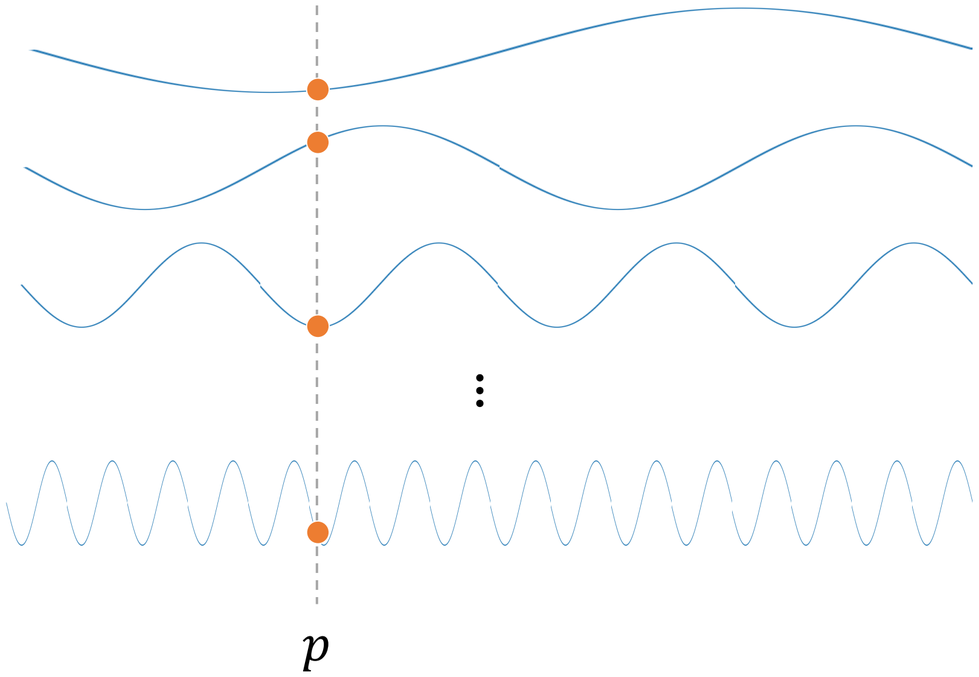
One-hot encoding

- 범주형 데이터 전처리 방법



Frequency encoding

- $\gamma(p) = (\sin(2^0\pi p), \cos(2^0\pi p), \ldots, \sin(2^{L-1}\pi p), \cos(2^{L-1}\pi p))$



EliceAcademy

05

AI 기법 성능향상 방법론

학습 = 표현 + 평가 + 최적화

- 모든 머신러닝 알고리즘은 세 가지의 조합
  - 표현
    - 데이터를 어떤 피쳐를 이용하여 표현할 것인가
    - *e.g.*, 피쳐 선택, 정규화 및 표준화, 인코딩
  - 평가
    - 모델을 어떻게 평가할 것인가
    - *e.g.*, RSS, MSE, MAE, R^2, 교차검증
  - 최적화
    - 모델을 어떻게 최적화할 것인가
    - *e.g.*, 모델 선택, 최적화 방법, 학습률

중요한 것은 일반화

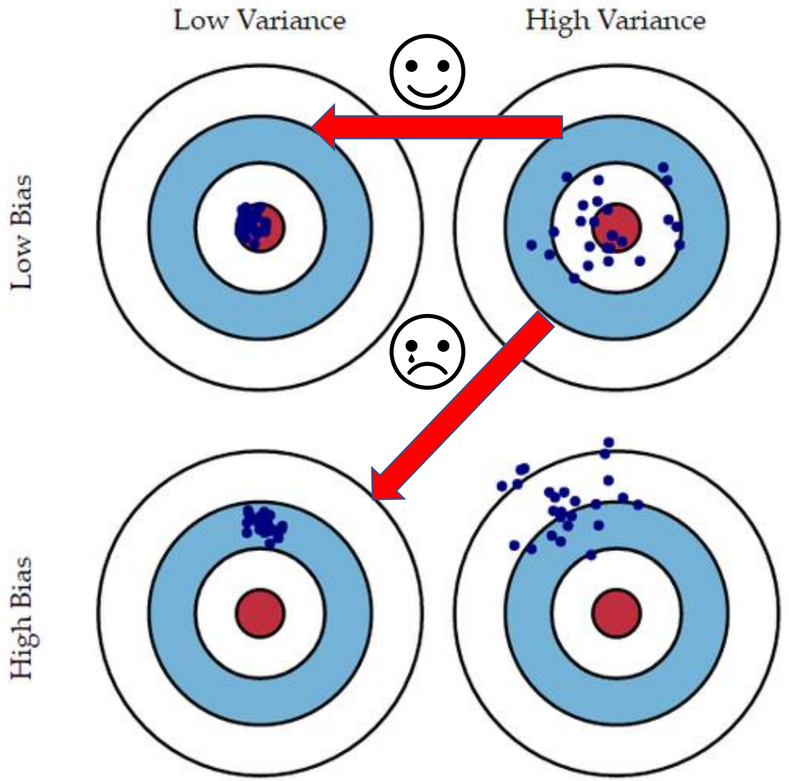
- 학습 데이터
  - 모델을 학습하기 위해 사용되는 데이터
  - *e.g.*, 오늘까지 수집한 금속분말 데이터
- 테스트 데이터
  - 모델을 실제 적용할 데이터
  - 모델이 본 적 없는 데이터
  - *e.g.*, 내일의 금속분말 데이터
- 과적합 주의!

데이터만으로는 불충분하다

- 작물은 땅이 키우지만, 농작은 농부가 한다!
- 데이터로 인해 모델이 학습되지만, 데이터 과학자의 역할도 중요하다.
- 머신러닝은 마법이 아니다.
- 도메인 지식을 잘 녹여내는 것이 중요하다.

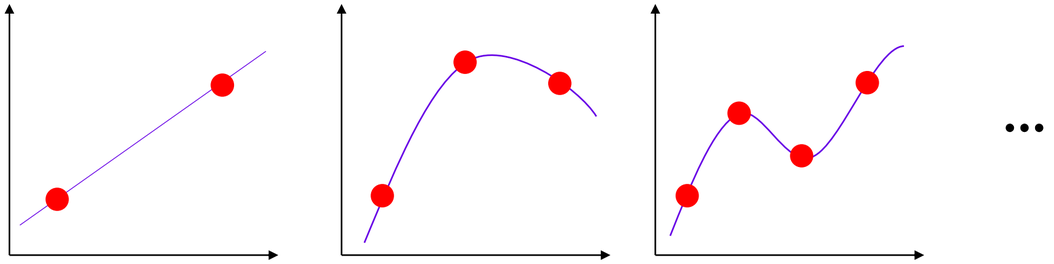
과적합 방지는 신중하게

- Overfitting
  - 학습데이터 정확도: 100%
  - 테스트 데이터 정확도: 50%
  - Bias는 낮고, Variance는 높음
- Underfitting
  - 학습 데이터 정확도: 50%
  - 테스트 데이터 정확도: 50%
  - Bias는 높고, Variance는 낮음
- Overfitting 해결
  - Underfitting 상태로 가는 것을 말하는 것이 아님!



고차원으로 갈수록 해석이 어렵다

- 과적합 다음으로 머신러닝의 주요 이슈는 **차원의 저주**
- Weight 수에 따라 필요한 데이터의 수
  - 1차 함수: 2개 점
  - 2차 함수: 3개 점
  - 3차 함수: 4개 점
  - ...
  - MLP (수많은 weight): 수많은 데이터
- Weight 수가 많아질수록 해석이 어려워진다.
- 그러나, 차원의 저주보다 더 많은 이점 (성능) 때문에 자주 쓰임



머신러닝 이론이 항상 정답은 아니다.

- 머신러닝 접근법은 기본적으로 **Induction**
- 즉, 관찰에 의거하여 이론을 정립
- 따라서, 이론만큼 **경험**이 중요
  - *e.g.*, Dropout 이 항상 일반화를 보장하지는 않는다.

Deduction	Induction
이론 ↓ 가정 ↓ 관찰 ↓ 확인	이론 ↑ 가정 ↑ 패턴 ↑ 관찰

피쳐 엔지니어링이 핵심이다

- 실제 머신러닝 프로젝트의 대부분은 **데이터 가공**에 있다.
- 데이터 가공은 한번에 끝나는 것이 아니라, 학습과 평가를 반복하며 수정해나가는 것이다.
- 피쳐 엔지니어링은 **도메인 지식**을 요구하기 때문에 가장 어렵다.

더 많은 데이터가 더 나은 알고리즘을 이긴다

- 대부분의 컴퓨터과학 알고리즘의 병목은 2가지에서 발생한다
  - 시간
  - 메모리
- 머신러닝은 한가지 요인이 더 있다.
  - **데이터**
- 더 나은 알고리즘을 고민하는 것보다, 더 **많은 양의 양질의 데이터**를 수집하는 것이 중요할 수 있다.
  - *e.g.*, 테슬라는 데이터 수집과 가공을 하는 데이터 작업자가 1000명이 넘는다.

여러 종류의 모델을 학습하라

- 최고 성능을 내는 머신러닝 모델이 모든 데이터에 대해 최고는 아니다.
- 각각의 모델은 특정 데이터에 장점을 가진다.
- 다양한 종류의, 다양한 hyper-parameter 를 가진 모델을 학습해보는 것이 필요하다.
- *e.g.*, 앙상블

“오캄의 면도날” 이 항상 옳지는 않다.

- 오캄의 면도날
  - 더 적은 수의 논리로 설명이 가능한 경우, 많은 수의 논리를 세우지 말라
- 그러나 “simplicity”에 지나치게 집착할 필요는 없다.
- 복잡한 모델이 더 좋은 성능을 가지는 경우도 분명히 존재한다!

상관관계가 인과관계를 나타내는 것은 아니다

- A 피쳐와 B 피쳐가 상관관계가 있다고 해서, A로 인해 B가 발생하는 것은 아닐 수 있다.
- 이런 경우에 A 피쳐가 모델에 입력되면, 원하는 방향으로 학습되지 않을 수 있다.
- 야구 게임 영상을 보고, 어느 팀이 이길지 예측하는 모델
  - **좋은 입력 피쳐**: 선수들의 컨디션, 구장 상태 등
  - **나쁜 입력 피쳐**: 화면에 떠있는 스코어
  - 화면에 떠있는 score을 보고 어느 팀이 이길지 예측한 모델이 내일 있을 야구 게임에서 승패를 예측하는데 쓰일 수는 없다!

Quiz

Model parameter 와 다르게 데이터로부터 학습되지 않고 사용자가 직접 지정해주어야 하는 변수를 무엇이라 부르나요?

- Hyper-parameter

Quiz

레이어가 깊어질수록 gradient 가 0에 수렴하게 되어 학습이 잘 되지 않는 문제를 무엇이라 부르나요?

① Gradient vanishing

② Overfitting

③ Local minima

④ Mode collapse

Quiz

머신러닝 모델의 성능 향상을 위해 사용할 수 있는 방법으로 올바르지 않은 것을 고르세요.

① 신중한 데이터 가공

② 도메인 지식을 활용한 모델 설계

③ 많은 모델 학습, 평가

④ 최대한 많은 피쳐를 활용