인공지능 > 머신 러닝 > 딥러닝

심볼릭 AI < - 옛날 지배적인 패러다임 (체스 게임) : 잘 정의한다면 논리적인 문제해결가능

머신러닝 : 명시적인 규칙이나 프로그램 없이 데이터로부터 학습하는 능력을 갖는 것

딥러닝 : 머신러닝 중 신경망을 사용하는 것

What is Machine learning?

전통적   
데이터 -> 일반 프로그래밍(규칙) - > 답 (어느정도 output을 알고있음)

데이터 / 답 -> 머신러닝 -> 규칙 (미리 답을 정의해놓고 그 룰을 만들어냄)

머신러닝의 특징

트레이닝 과정이 필요 -> 고성능 하드웨어 / 대량의 데이터  
통계와는 다르게 대용량의 복잡한 데이터셋을 다룸

딥러닝   
엔진니어링측면에서 효율화 개선 -> 경험으로 바탕으로 아이디어를 증명하는 경향이 있음

머신러닝이 어디에 유용한가?

1. 이미지 분류
2. 뇌종양 스캔(의학)
3. 챗봇
4. 보이스
5. 추천 / 예측

Reinforcement Learning: Robots, Games, 자율주행

<딥러닝> Represation Learning

데이터로부터 층(깊이) 심층학습으로 핵심적인 특징을 새롭게 표현하는 과정 (다단계 처리학습)

목표 : 가중치의 정확한 값을 찾는 것 -> 가중치 -> 층(데이터변환) - > 예측

입력

손실함수(목적함수) -> 테스트와 예측값간의 차이를 점수로 표현   
이렇게 얻은 손실 점수를 옵티마이저 해서 가중치를 업데이트 -> 성능향상  
이러한 과정을 트레이닝 루프(Training Loop)

ML 분류

* 지도 / 비지도 / 준지도
* 강화

ML 사례(수식)

* X->Y , 함수개념
* 지도 / 비지도학습 개념

ML 사례(코드)

* Keras 설치 및 실행
* 코드리뷰(딥러닝 예제)

• 지도학습(Supervised learning)  
- x : 데이터 (미리 정답을 구축) y : 정답  
- x -> y 맵핑Function을 찾는 것  
- Classification(분류), Regression(회긔)

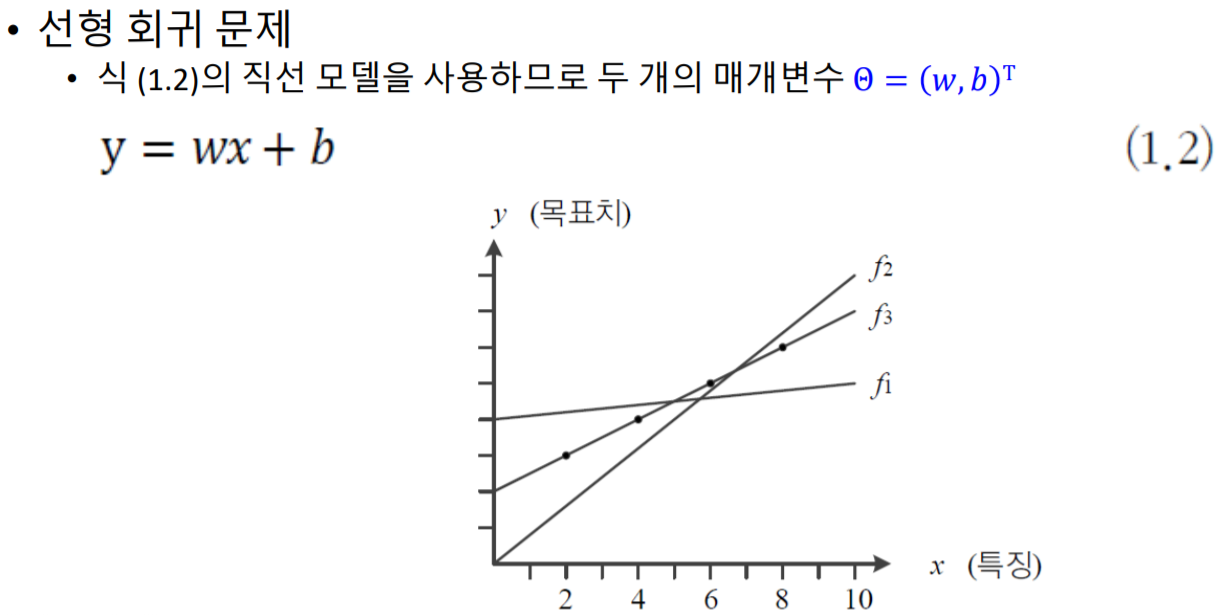
• 비지도학습(Unsupervised learning)  
- x : 데이터만 존재 (비슷한 정도만 제시, 목표값 없음)  
- 스스로 맞춤  
- 정답이 필요 없음   
- Clustering, Dimensionality reduction, Anomaly/Novelty detection

• 준지도학습(Semi-supervised learning)   
- 일부는 x와 Y를 모두 가지지만, 나머지는 X만 가진 상황  
- X의 수집은 쉽지만, Y는 수작업이 필요한 경우 유용함

• 강화학습(Reinforcement learning)  
 - 단계별 정책을 정함  
 - 묙표를 위한 단계 학습

다양한 기준에 따른 유형

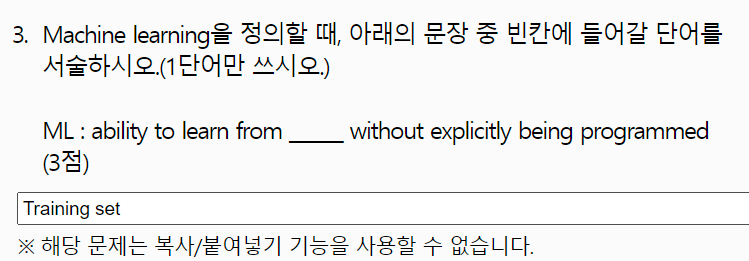
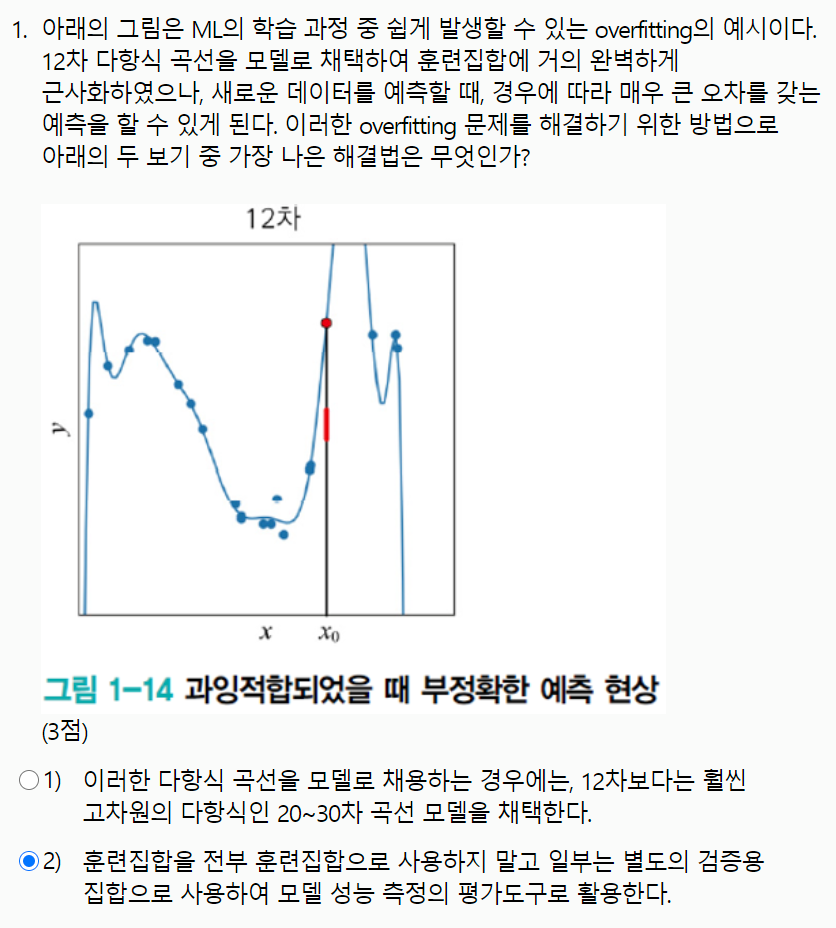
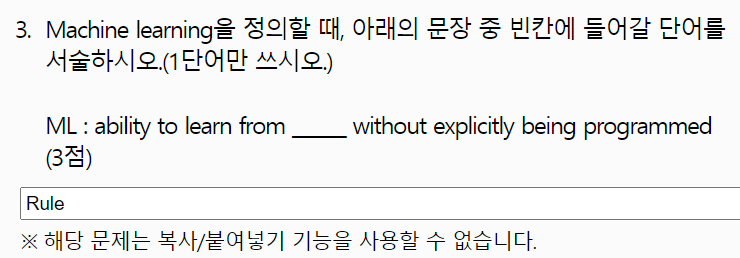
* 오프라인 학습과 온라인 학습(리얼 타임)
* 결정론적(deterministic)학습과 스토캐스틱(stochastic)학습  
  - 결정론적에서는 같은 데이터를 가지고 다시 학습하면 같은 예측기가 만들어짐  
  - 스토캐스틱 학습은 학습 과정에서 난수를 사용하므로 같은 데이터로 다시 학습하면 다른 예측기가 만들어짐.
* 분별(discriminative)모델과 생성(generative)모델  
  - 분별 모델은 분류 예측에만 관심. 즉 P(y|x)의 추정에 관심  
  - 생성 모델은 P(x) 또는 P(x|y)를 추정



기계학습에 베이스 정리  
예) Iris 데이터 분류 문제

정보 이론의 기본 원리 -> 확률이 작을수록 많은 정보

최적화 -> 주어진 데이터에서 최적의 해를 찾는 것이 (에러가 가장 낮은) 머신러닝의 핵심  
  
기계학습의 최적화  
-> 단지 훈련집합이 주어지고, 훈련집합에 따라 정해지는 목적함수의 최저점을 찾아야함  
 - 데이터로 미분하는 과정 필요 -> 오류 역전파 알고리즘  
 - 주로 스트캐스틱 경사 하강법 사용



Regression (회귀)

Y 절편인 b를 구하는 공식

B = y의 평균 –(x의 평균 x 기울기a)

Y(예측값) = ax + b

오차수정(잘못 그은 선 바로잡기)

* 일단 선을 그리고 -> 조금씩 수정해 나가기
* 언제까지?  
  - 오차가 최소가 될 때까지
* 오차?
* 선형그래프와 각 값과의 오차가 크다 <- 잘못됨

<오차>

평균 제곱 오차 (MSE)

평균 제곱 오차 = (예측값)-y(실제값))^2 의 평균

기울기 a와 오차와의 관계 : 적절한 기울기를 찾았을 때 오차가 최소화됨.

학습률:

* 어느 만큼 이동시킬지를 신중히 결정해야 하는데, 이떄) 이동 거리를 정해주는 것

경사하강법:

* 오차의 변화에 따라 이차 함수 그래프를 만들고 적절한 학습률을 설정해 미분 값이 0인 지점을 구하는 것 (Global minimum)
* Local minimum 으로 인해 기울기가 0이라서 오판할 때 있음.
* 매우 긴 구간 정체된 경우(Plateau) 오판할 때 있음

이를 위한 해결법

* Batch Gradient Descent  
  한 스탭의 배치로서 트레이닝 데이터셋을 통째로 가지고 있어야 한다.   
  트레이닝 셋이 큰 경우 느리다.
* Stochastic gradient Descent  
  배치보다 빠르다.   
  Stochastic->random   
  어느 시점에서 멈출 지 필요함  
  멈춤 시점이 최적의 코스트라는 보장은 할 수 없음. -> 러닝 레이스를 high -> low (점진적으로 줄이도록) 해결책 중 하나
* Mini-batch Gradient Descent  
  풀 배치를 계산하는 것(Batch)도 아니고 하나의 샘풀에 대해서만 계산하는 것(Stochastic)은 아니고 작은 단위의 배치로 계산  
  하드웨어가 받쳐주면 미니배치가 병렬작업으로 인해 더욱 빠르다.

일반화가 어렵다 -> 오버피팅’

희소한 모델(노이즈)/큰트렌드🡨 라소이용

Early stopping -> 트레이닝하다가 validation error가 최소화될 때 닥 스탑

규제화 정규화에 특화

Logistic Regression

50%확률로 넘나 안넘나 분류를 하는 것!(binary classfier)