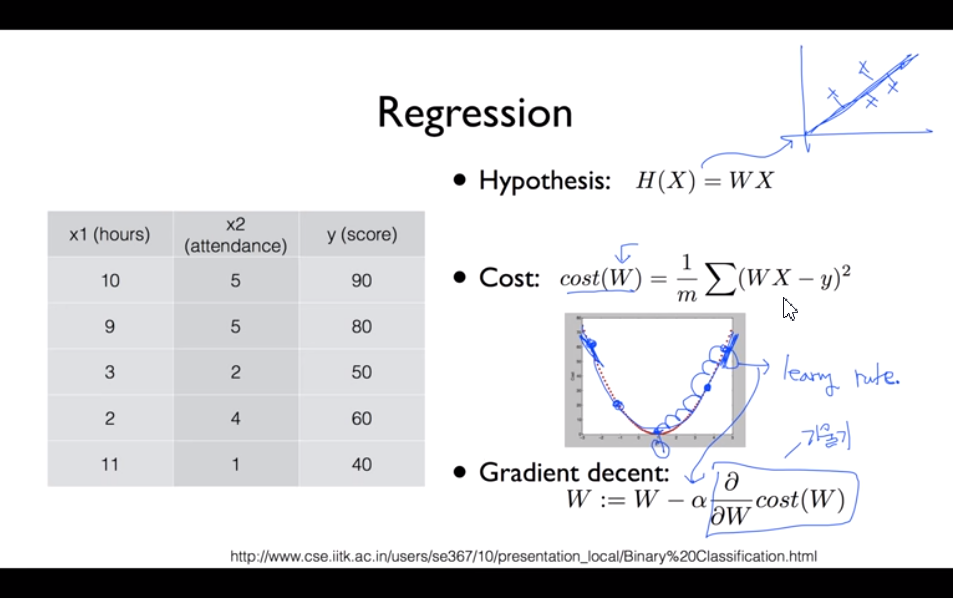
5-1, 5-2 / 8-1, 8-2

ML lec 5-1: Logistic Classification의 가설 함수 정의

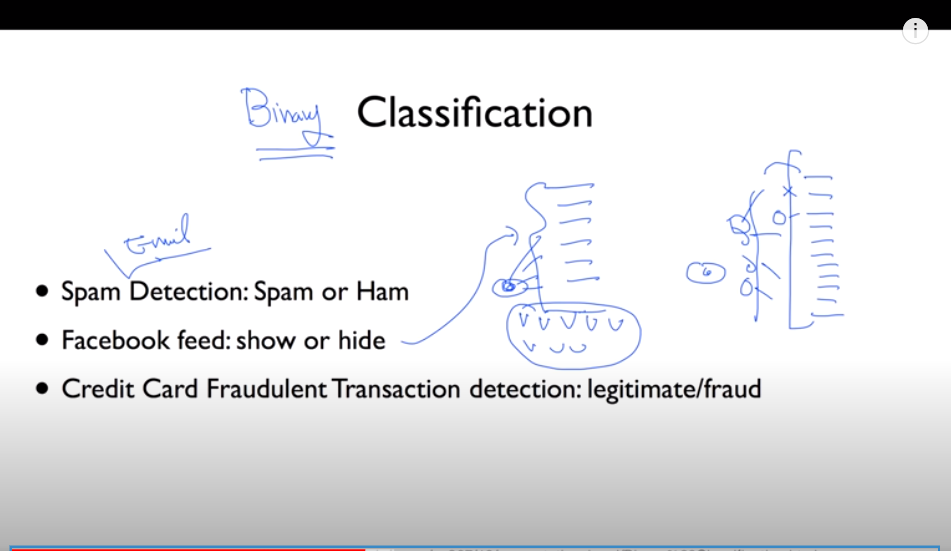
Logistic (regression) Classification

<이전>



Linear Regression (H, C, G)

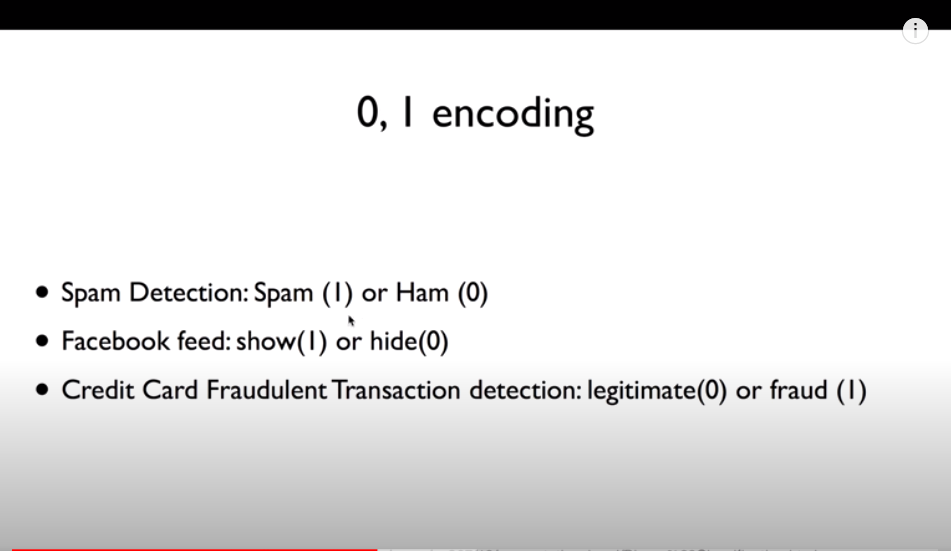
Cost를 최소화하는 W값을 찾는 것이 목표



Binary **Classification**  (1,0)

분류: 머신러닝에서 많이 활용되는 분야

<예> 요즘은 메일에서 스팸 잘 안보임 / 페이스북 피드 전체 다 보여주는게 아니라 몇 개는 숨기고 재밌는 거는 보여주기 / 신용카드 쓰던 패턴에서 벗어나면 가짜.



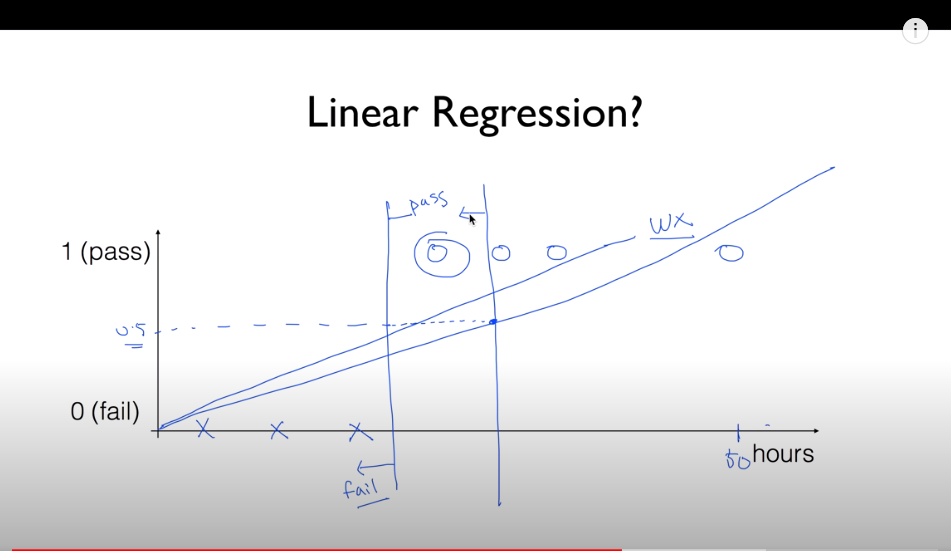
이를 기계적으로 학습하기 위해 숫자 0과1을 이용해 인코딩

최근에는 많이 쓰이고 있기 때문에 굉장히 중요한데,

다음과 같은 (예)

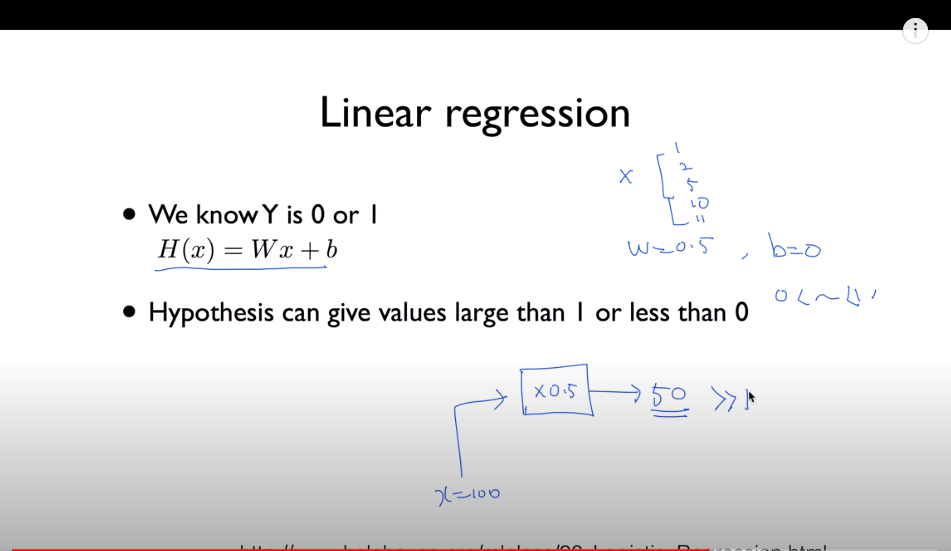
* 좋은 종양인지, 나쁜 종양인지 판별
* 주식 시장에서 이걸 내가 살까 팔까 - 이전에 있는 기록을 통해 학습할 수 있음.

ㄴ주식시장에서의 머신 러닝



[0과 1을 예측하는 모델]

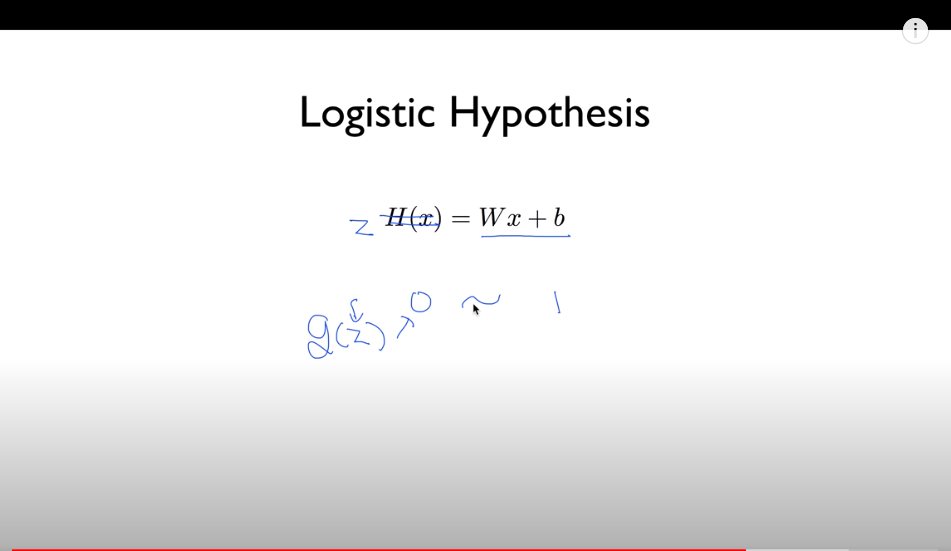
* Linear Regression 이용하기엔 한계가 있음.



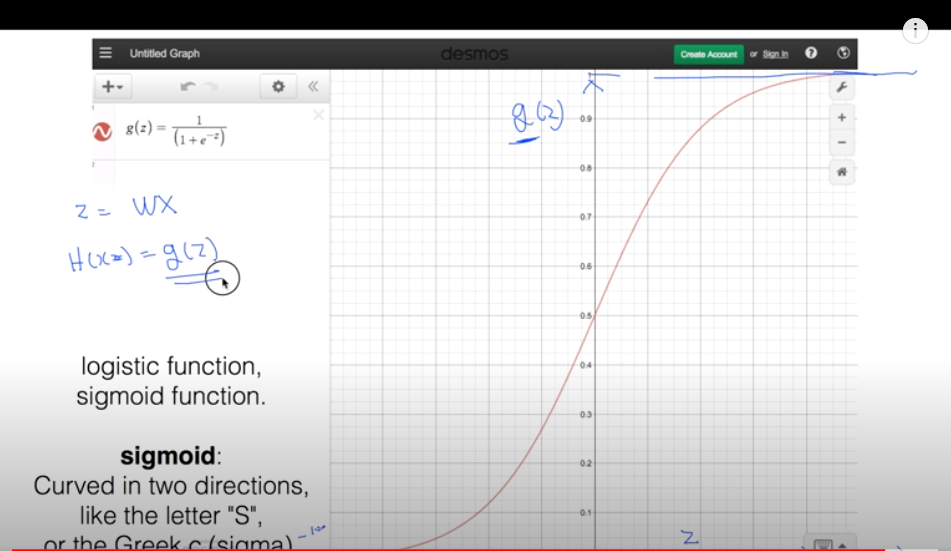
* Linear 에서 위의 식은 모델이 좋고 간단하긴 한데

이것을 0과 1사이로 압축을 시켜주는 어떤 형태의 함수가 하나 있었음 좋겠다!

* 🡪



어떤 z값이 들어오든 g(z) 값은 0과 1사이인 함수가 필요!



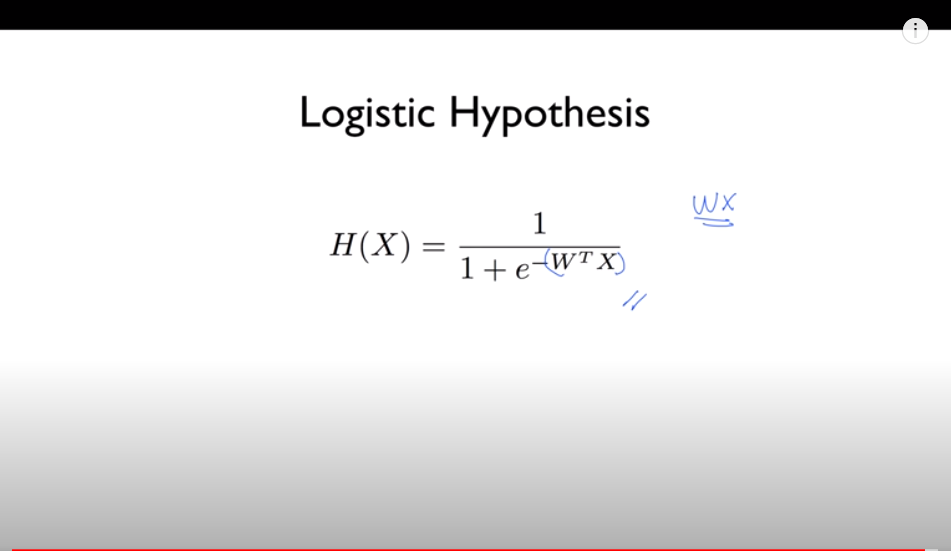
이러한 함수를 ‘sigmoid’라고 부름

Z , g(z)

Z가 커지면 ~값은 1에 가까워짐. 더 이상 커지지 않음

작아져도 0에 가까워지기만 할 뿐, 더 이상 작아지지 않음.

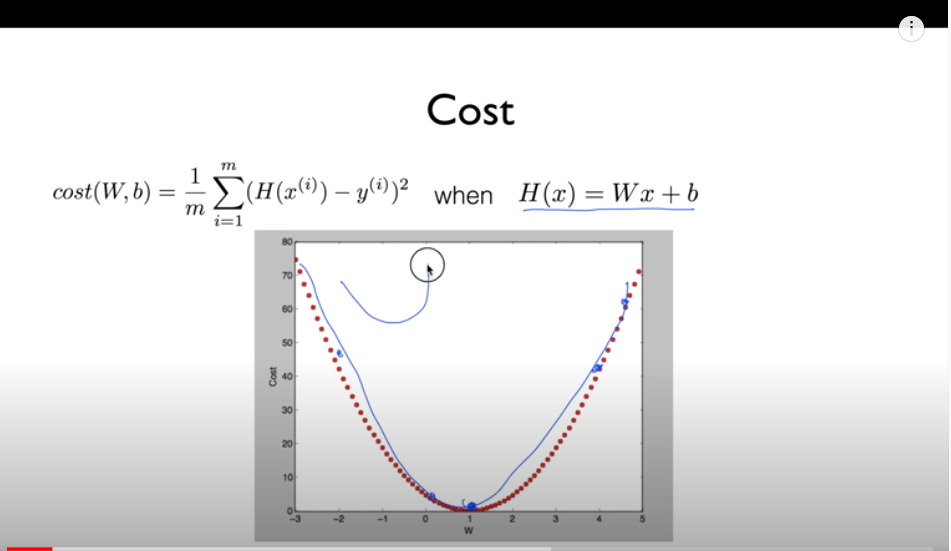
0~1사이의 값을 얻을 수 있음!



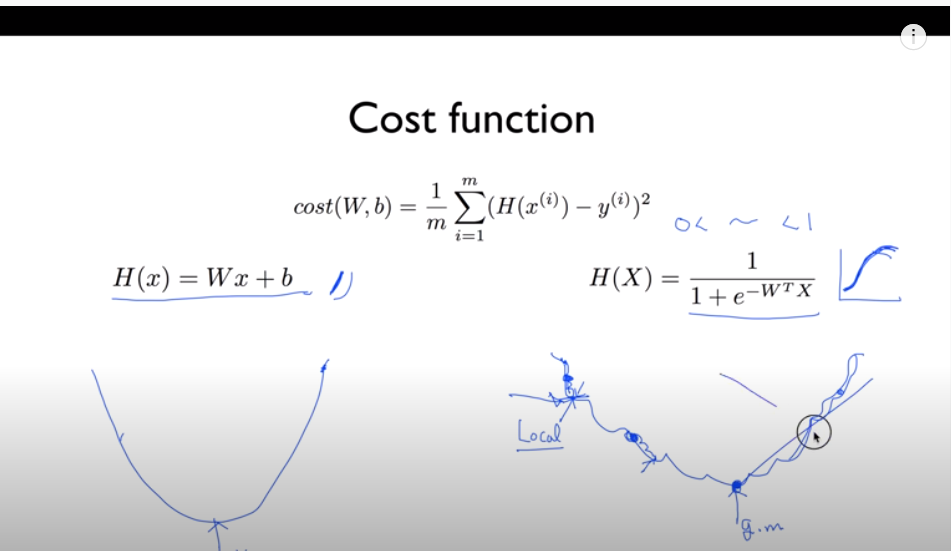
그래서 최종식! Logistic Classification Hypothesis 의 함수는 이렇게 됨.

ㅡㅡㅡㅡㅡㅡCost를 minimize시키는게 우리의 목표임!!!!!ㅡㅡㅡㅡㅡㅡ 5-2

cost함수와 cost함수를 최소화하는 알고리즘을 알아보자!



ㄴLinear.



하지만 가설이 바뀜.

1 매끄러운 밥그릇 모양 🡪 2울퉁불퉁 모양

1. 최저점에 도달할 수 있음.
2. 시작점이 어디냐에 따라 끝나는 점이 달라질 수 있음

ㄴLocal minimum, global minumum

우리는 global 미니멈 값을 찾는 것이 목표.

학습이 local 미니멈에 멈춰버리면 우리의 모델이 나쁘게 예측됨.

따라서 2 그래프는 사용할 수 없음

따라서 바꿔야함

결론적으로 우리가 사용하게 될 Cost함수!

합의 평균 /

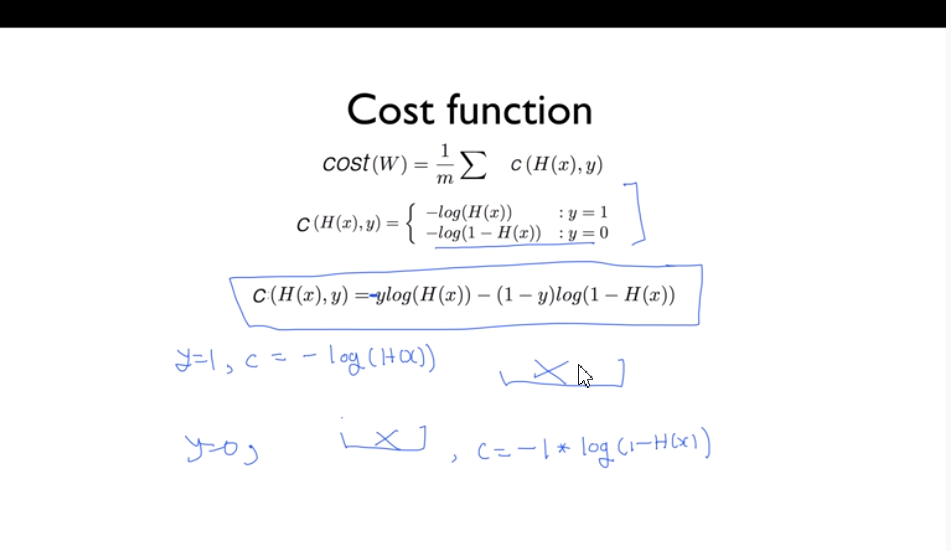
c함수는 두 값을 가지고..

cost함수란!

실제의 값과 우리가 예측한 값이 같으면 비슷하면 cost값이 작아지고!

작아지는 것이 좋음.cost값이 무한대로 커지면 모델에 벌을 주게됨. =우리가 잘못 예측했을 때.

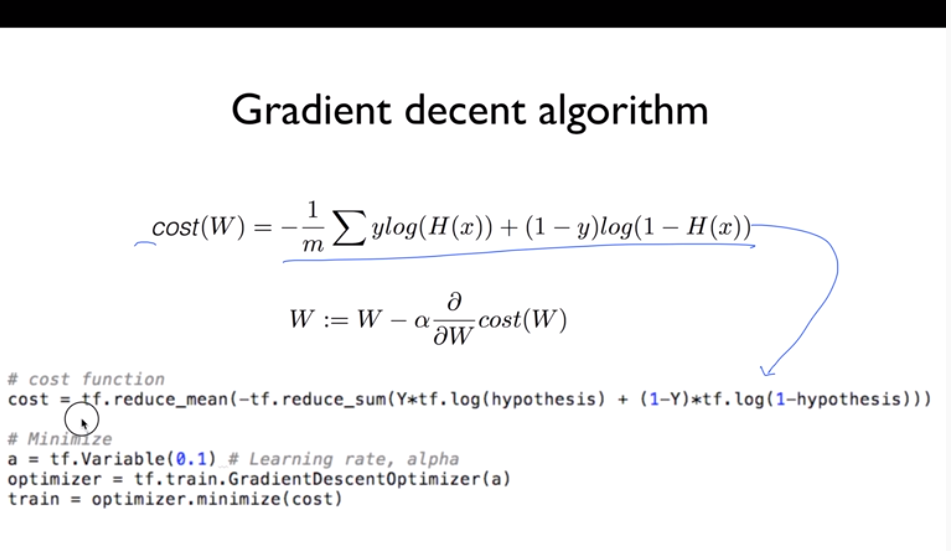
그래서 정리하자면



* cost함수는 합의 평균.
* 두개의 경우로 y=1, 0일 때로 나누어 cost 함수를 정의하면 잘 된다!

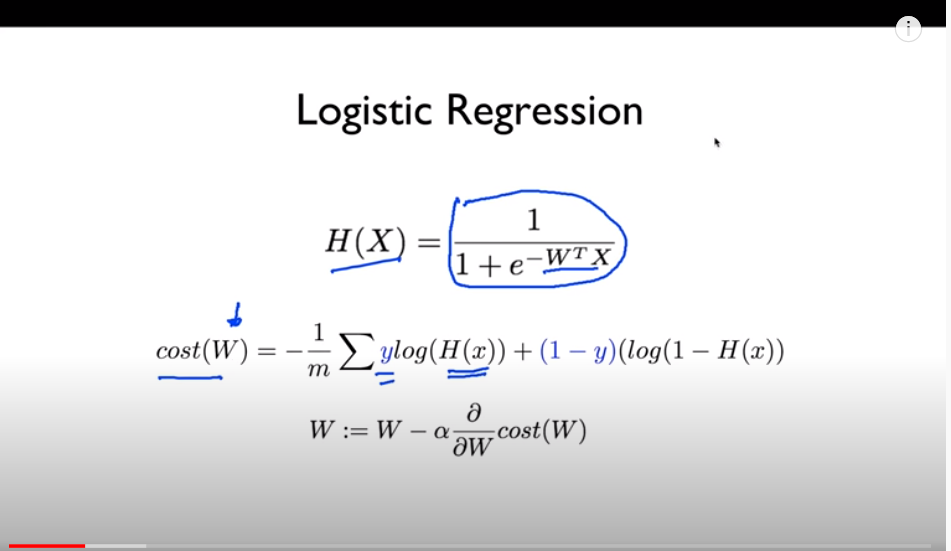
근데 텐서플로우에서 프로그래밍을 해야하니까!(너무 복잡하다)

그 두 개 식을 합쳐서 정리하자면 이렇게 된다!



GDO 라이브러리 존재. 이걸 가지고 이용!

정리



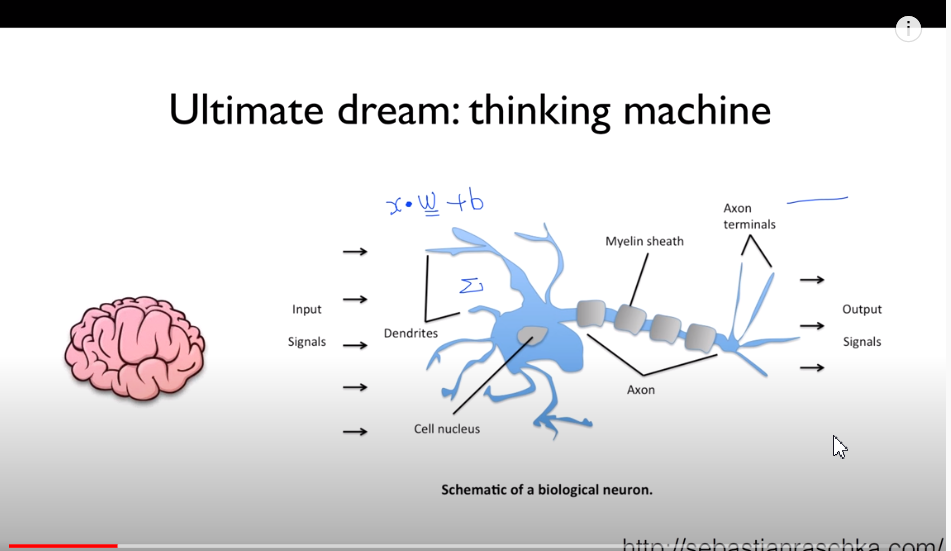
Cost를 최소화하는 w값을 구하는 것.

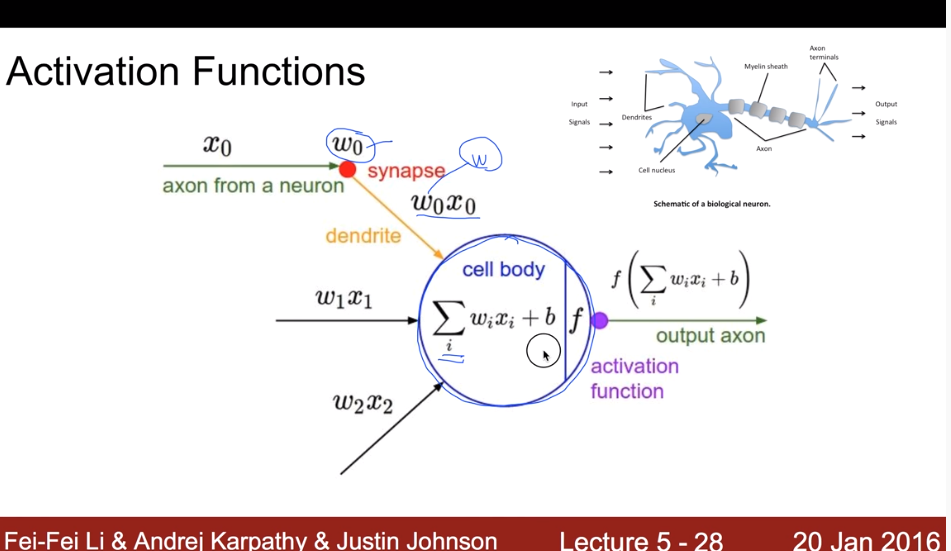
# lec 08-1: 딥러닝의 기본 개념: 시작과 XOR 문제

**Deep Neural Nets**

**딥러닝의 역사!**

이 아이디어가 어떻게 탄생했고 어떠한 문제들이 있었는지 설명하는 챕터.



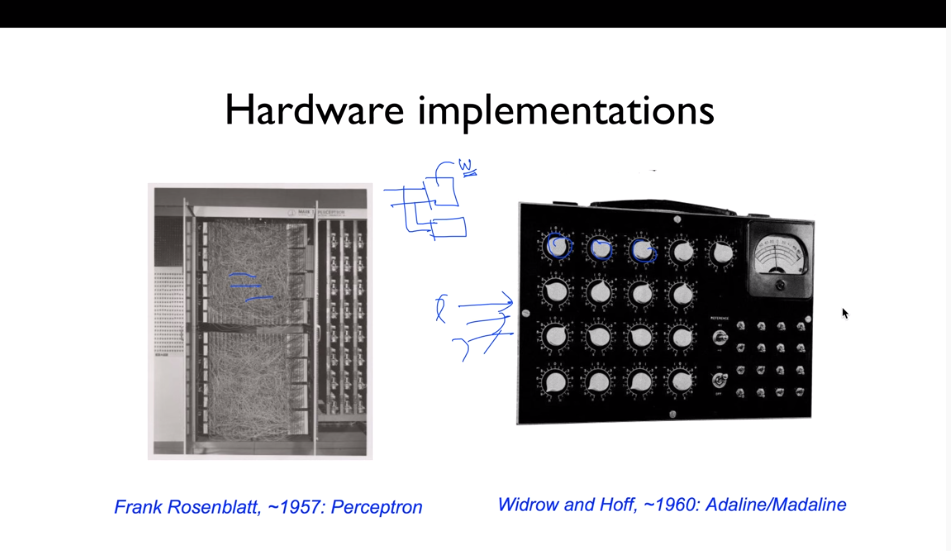


어떤 신호가 들어와 weight->sum

합한 값이 일정 값을 넘어가면 1이라는 신호를 주고, 넘어가지 않으면 신호를 주지 않는

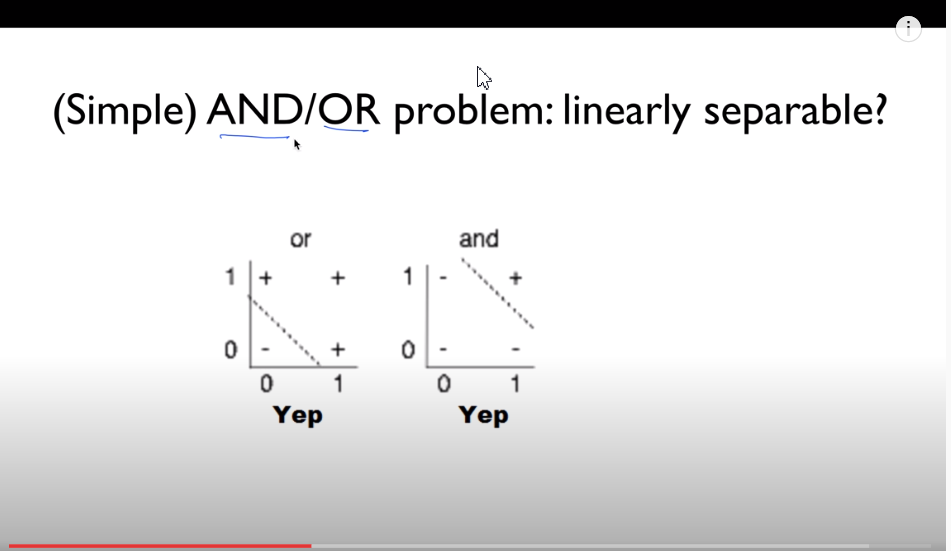
그런 단순한 형태로 할 수 있겠다!

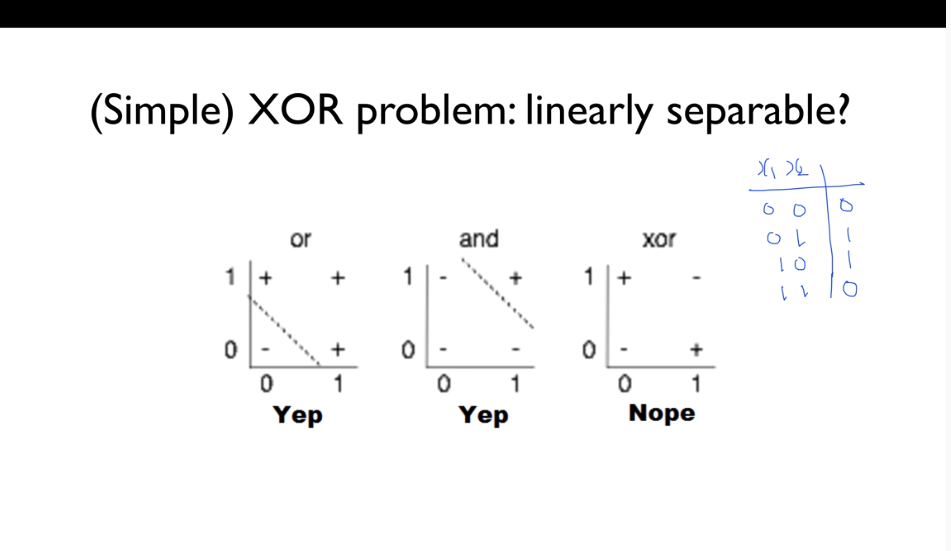
모은다면, 여러 개의 출력을 동시에 낼 수 있겠구나 ->



하드웨어로 만들 수 있음.

1. 선으로,

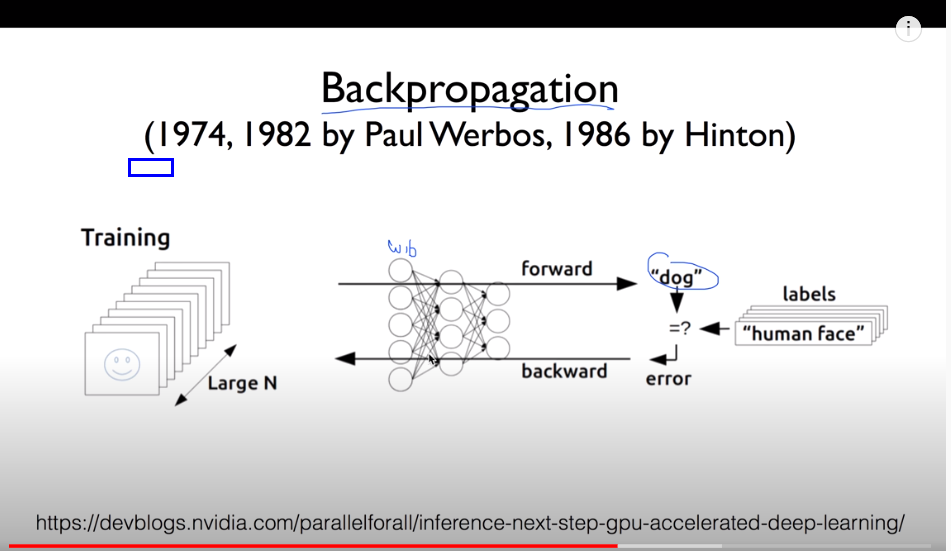




기계가 예측할 수 있도록 하는 것.

-🡪

민스키 교수: XOR는 풀 수 없다!!!!!!는 것을 발견

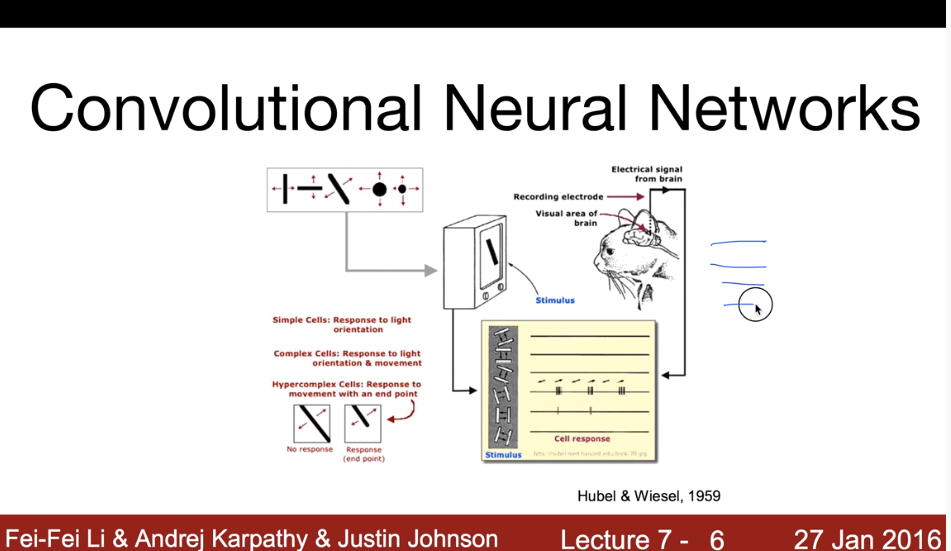


해답을 찾음

**Backpropagation 알고리즘** (무시-> 후에 재발견)

(뉴럴 네트를 학습하는 알고리즘의 하나)

‘다시 뒤로가면서 change시키면 어떨까’



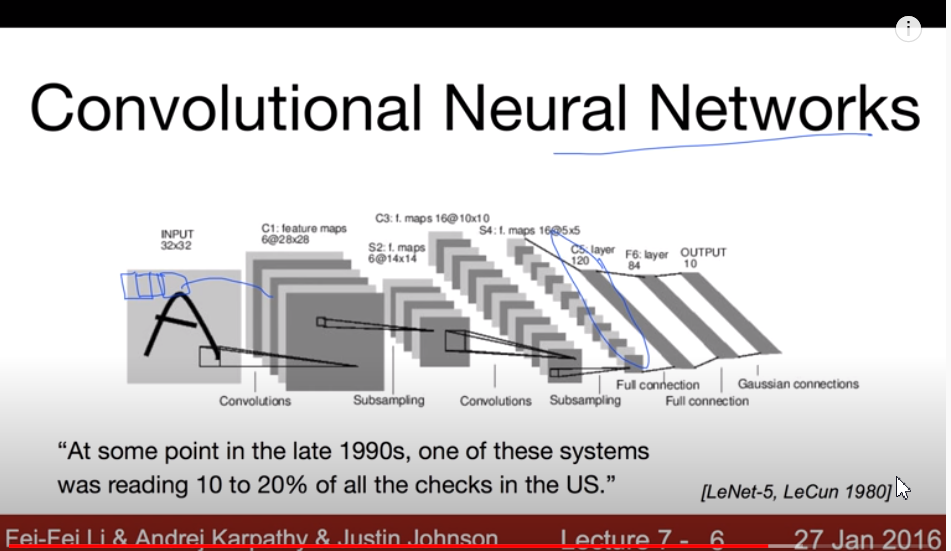
또E다른 교수님은 다른 방법으로 접근

그림을 볼 때 시신경 말고도…

여러 신경망이 모여 조합되는 것이 아닐까

* 뉴럴 네트워크

: 부분부분 잘라서 나중에 합치는. 성능 굿



뉴럴 네트워크를 이용하여

자율적으로 움직이는 자동차까지,

* Neural-net’
* 또 문제에 봉착
* Backpropagation 알고리즘(뒤에서 가는 구조)

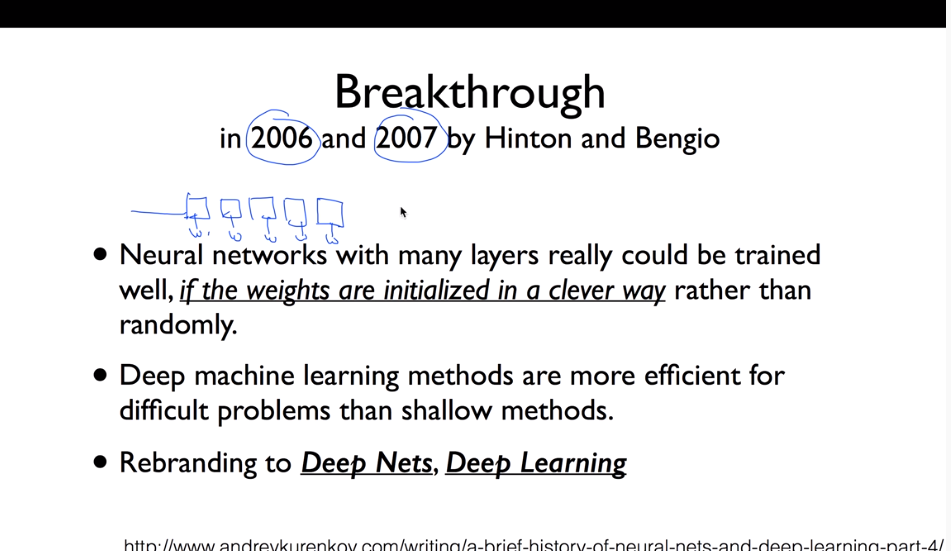
🡪 10개 이상되는 것에 적용이 불가.

많은 것을 할수록 성능이 더 떨어지게 됨.

ㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡ



단체!

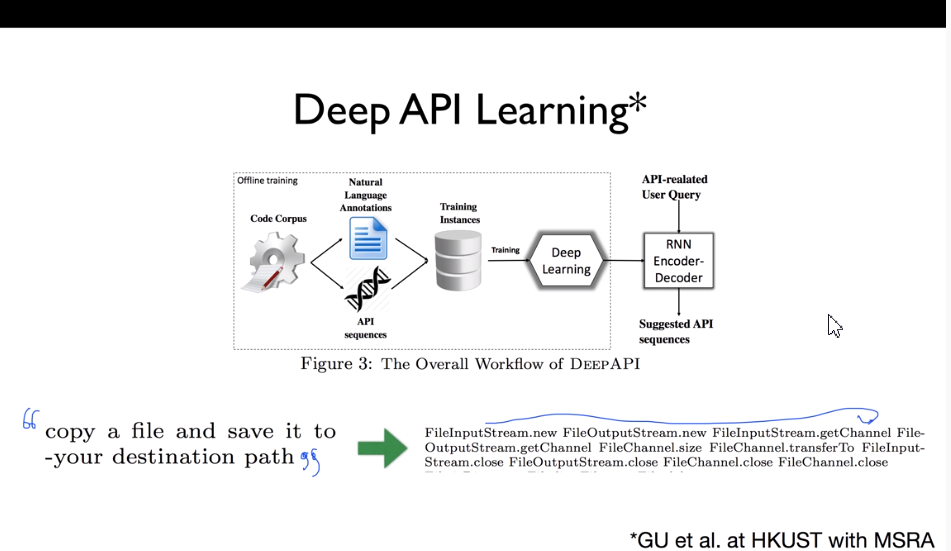


뉴럴 네트워크!

논문 두 개: 초기값을 잘 주면 된다! 깊게 신경망을 구축하면 복잡한 문제를 풀 수 잇다!

다시 주목

‘딥러닝’으로 재탄생!!!!!!!!!(신경망)

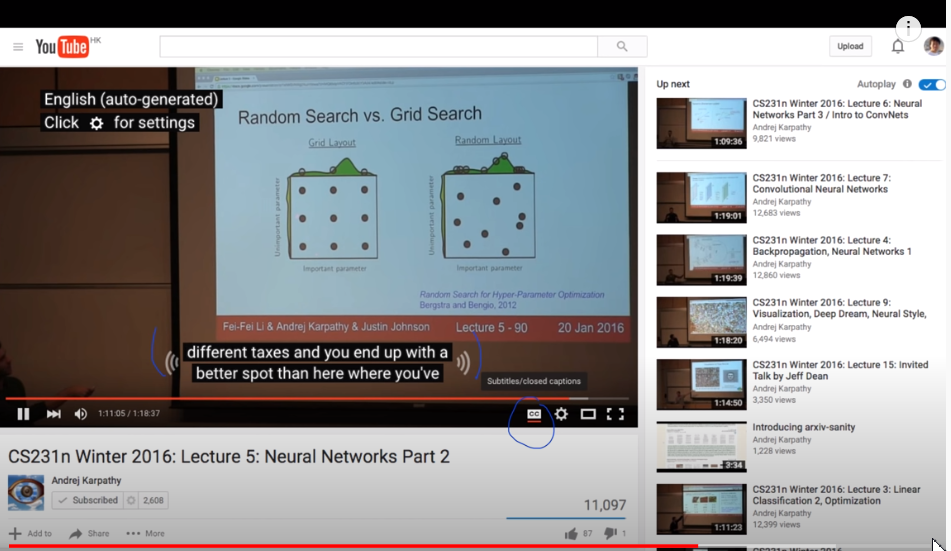


말만 하면 시스템이 알아서

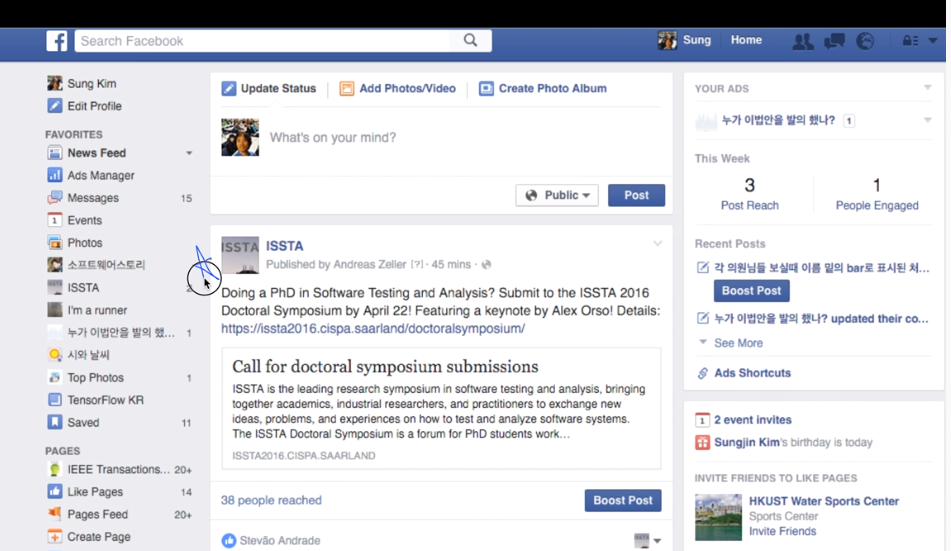
Api를 가지고 프로그램 완성시키기

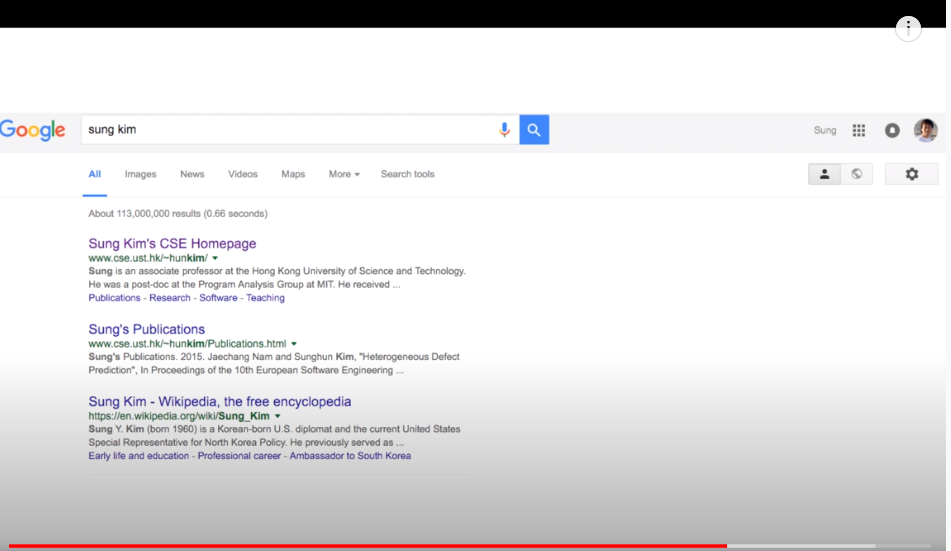
딥러닝 🡪 알파고의 승리까지

ㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡㅡ



유튜브 자막 자동 생성, 굉장히 정확





구글 검색 엔진

사용자가 클릭할만한 ..

넷플릭스 추천 시스템

아마존

추천 인공지능 머신러닝 잘 도입