신경망이 정량적으로 얼마나 나쁜지/좋은지 평가할 수 있는 기준이 필요함

* **Loss function** , 주어진 weight 평가

**Optimization** : 모든 가능한 영역(weight)중에서 최고로 좋은(= 최저로 나쁜) weight를 찾는 것

**Loss Function** : 주어진 망을 통해서 나온 값 (predict)과 실제 값을 이용하여 정량적으로 얼마나 나쁜 망인지 평가하는 함수.

-Finally, loss L는 모든 데이터셋에 대해 전체 loss의 평균을 의미

-General formulation (not only image classification but also all other machine learning tasks which has x and y)

-Loss function을 바탕으로 가능한 weight영역을 조사하여 loss를 최소화하는 weight를 찾으려고 함.

**SVM**

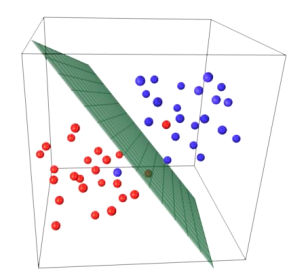
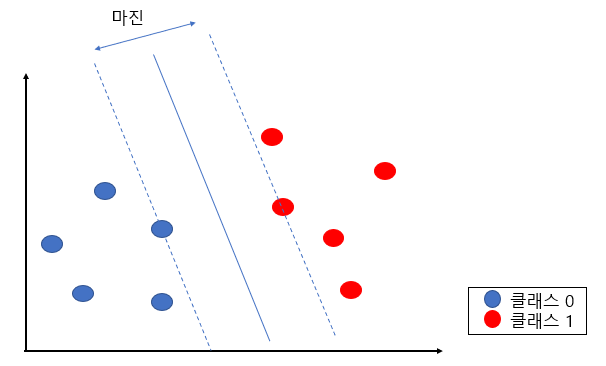
**n차원의 공간을 (n-1)차원으로 hyperplane으로 나눌 수 있다**. 이러한 hyperplane을 찾는 것!

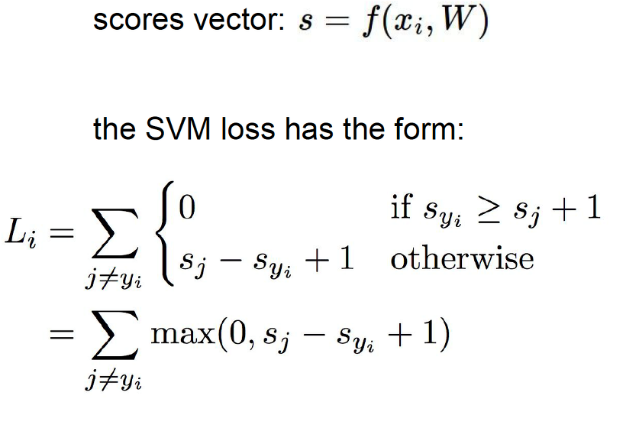
**Margine** = 클래스의 말단에 위치한 데이터들 사이의 거리, 즉 클래스들 사이의 간격

* **margine을 최대화하는 경계(hyperplane)을 찾는 것**

-이때 클래스의 말단에 위치한 데이터가 각각 클래스의 **support vector**. 이때 클래스의 말단이라는것은 hyperplane에서 가장 가까운 클래스별 데이터를 의미한다.

-Support vector인 이유는 이들에 따라 hyperplane이 달라지기 때문

1

**Multiclass SVM Loss**

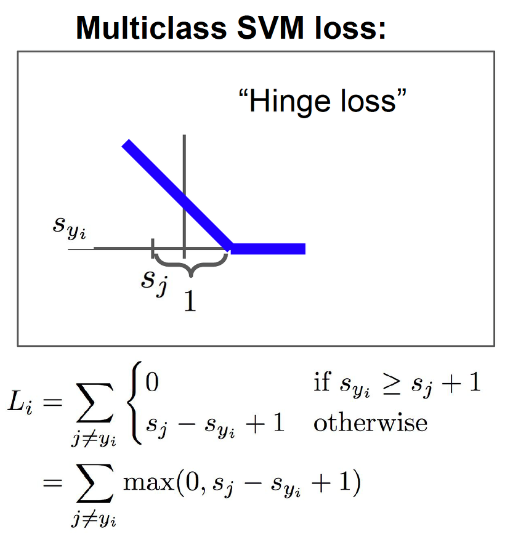
정답이 아닌 다른 label에 대해,

+ 1 <

일 경우 loss 가 0, 그렇지 않은경우

L = – + 1

이때 1 은 delta 혹은 safety margin 로 조금 더 안정적인 정답을 추구하기 위한 값.



x축은 값을 의미한다.

값이 클수록 loss가 0에 수렴

**Class 들의 절대적인 score보다 상대적인 score차이가 더 중요하다**

Q .처음에 training 을 시작할 때 모든 score가 0에 가깝다면 loss는 (#class -1)

이와같지 않다면 코드에 버그가 있는것

Q. Loss를 sum이 아닌 평균으로 구한다면?

상관없음. Loss를 rescale한 것. 실질적인 score의 점수가 궁금한 것이 아니기 때문

Q. Loss를 sum이 아닌 제곱하여 sum한다면?

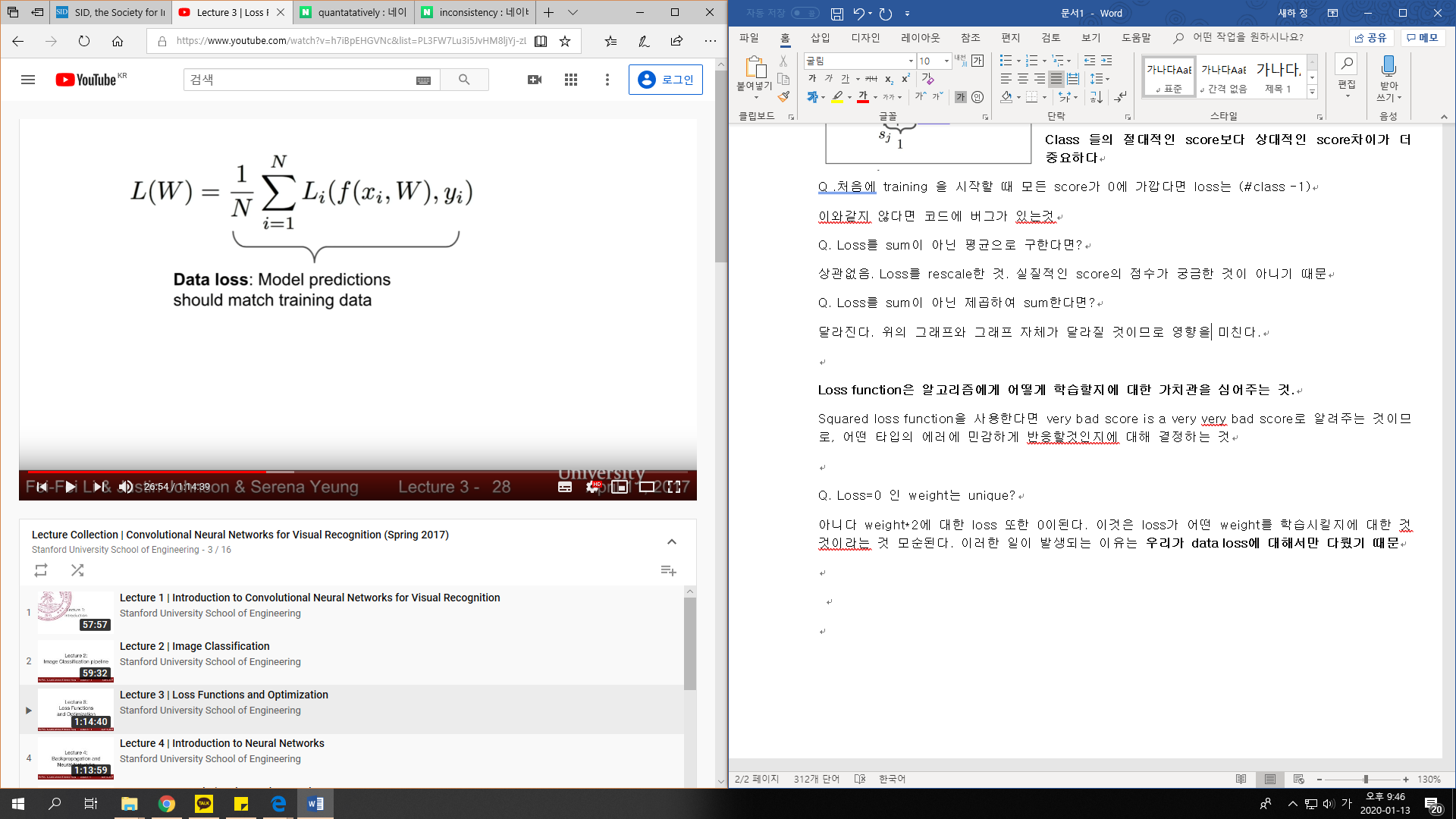
달라진다. 위의 그래프와 그래프 자체가 달라질 것이므로 영향을 미친다.

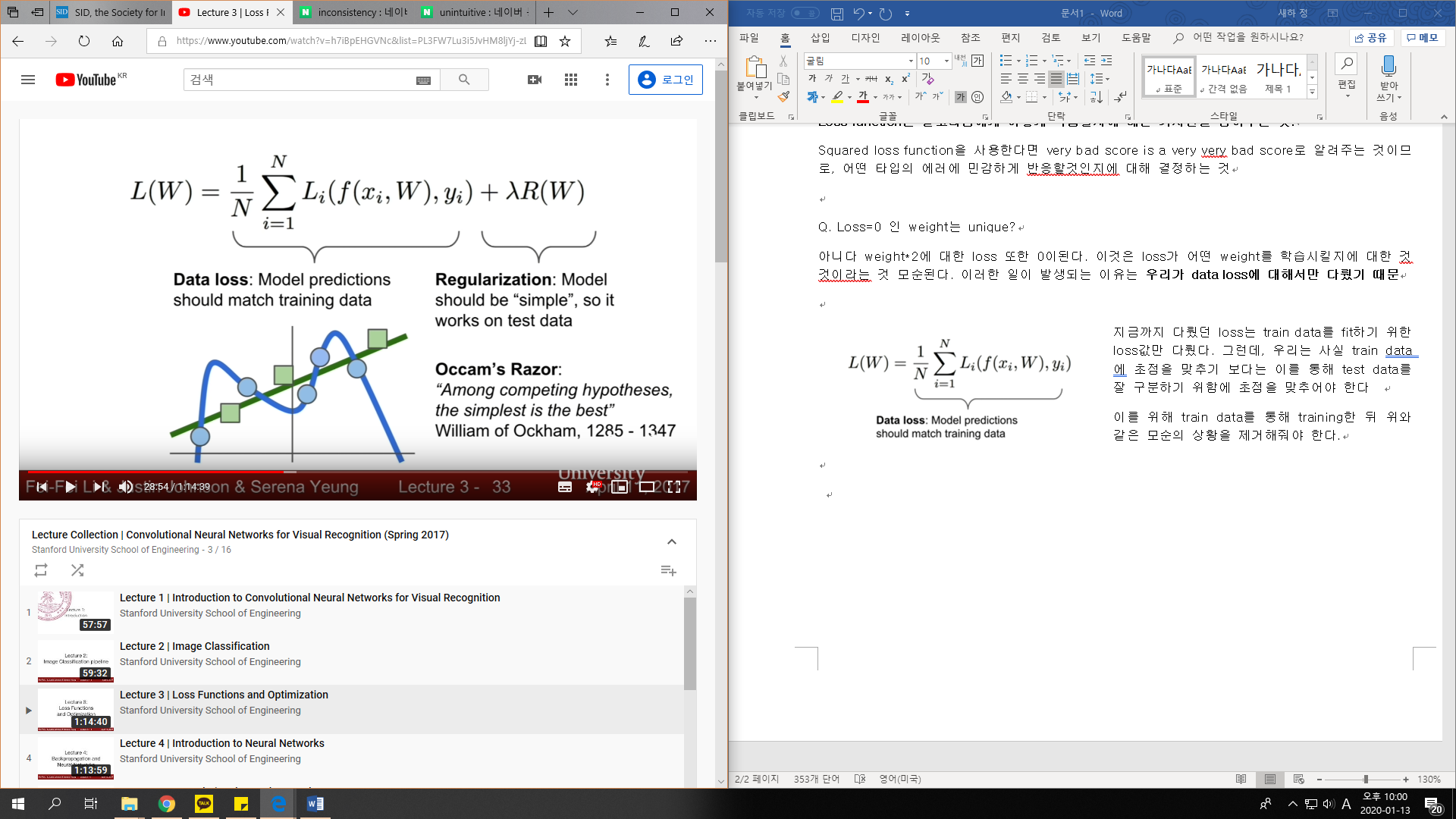
**Loss function은 알고리즘에게 어떻게 학습할지에 대한 가치관을 심어주는 것.**

Squared loss function을 사용한다면 very bad score is a very very bad score로 알려주는 것이므로, 어떤 타입의 에러에 민감하게 반응할것인지에 대해 결정하는 것

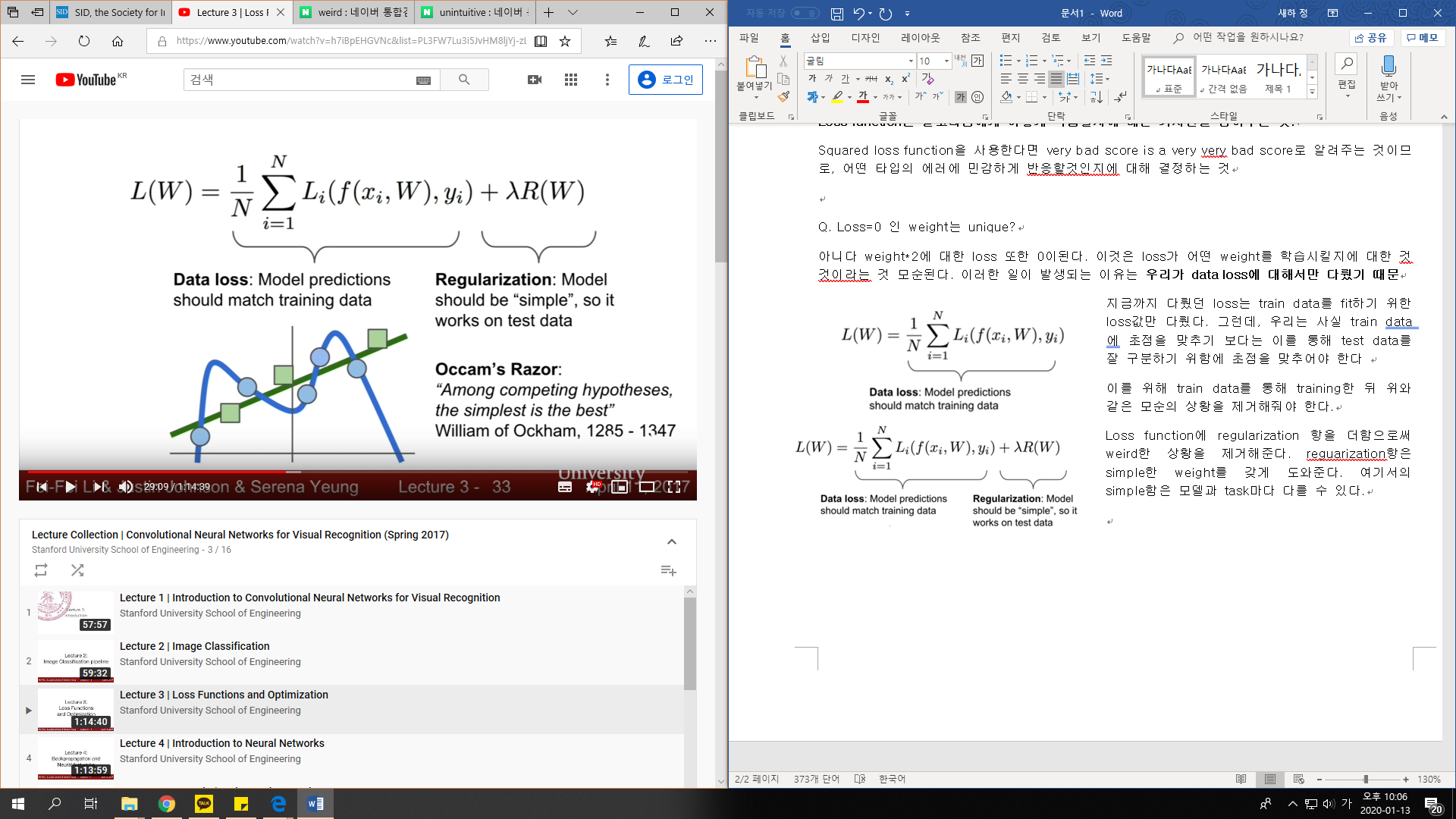
Q. Loss=0 인 weight는 unique?

아니다 weight\*2에 대한 loss 또한 0이된다. 이것은 loss가 어떤 weight를 학습시킬지에 대한 것것이라는 것 모순된다. 이러한 일이 발생되는 이유는 **우리가 data loss에 대해서만 다뤘기 때문**

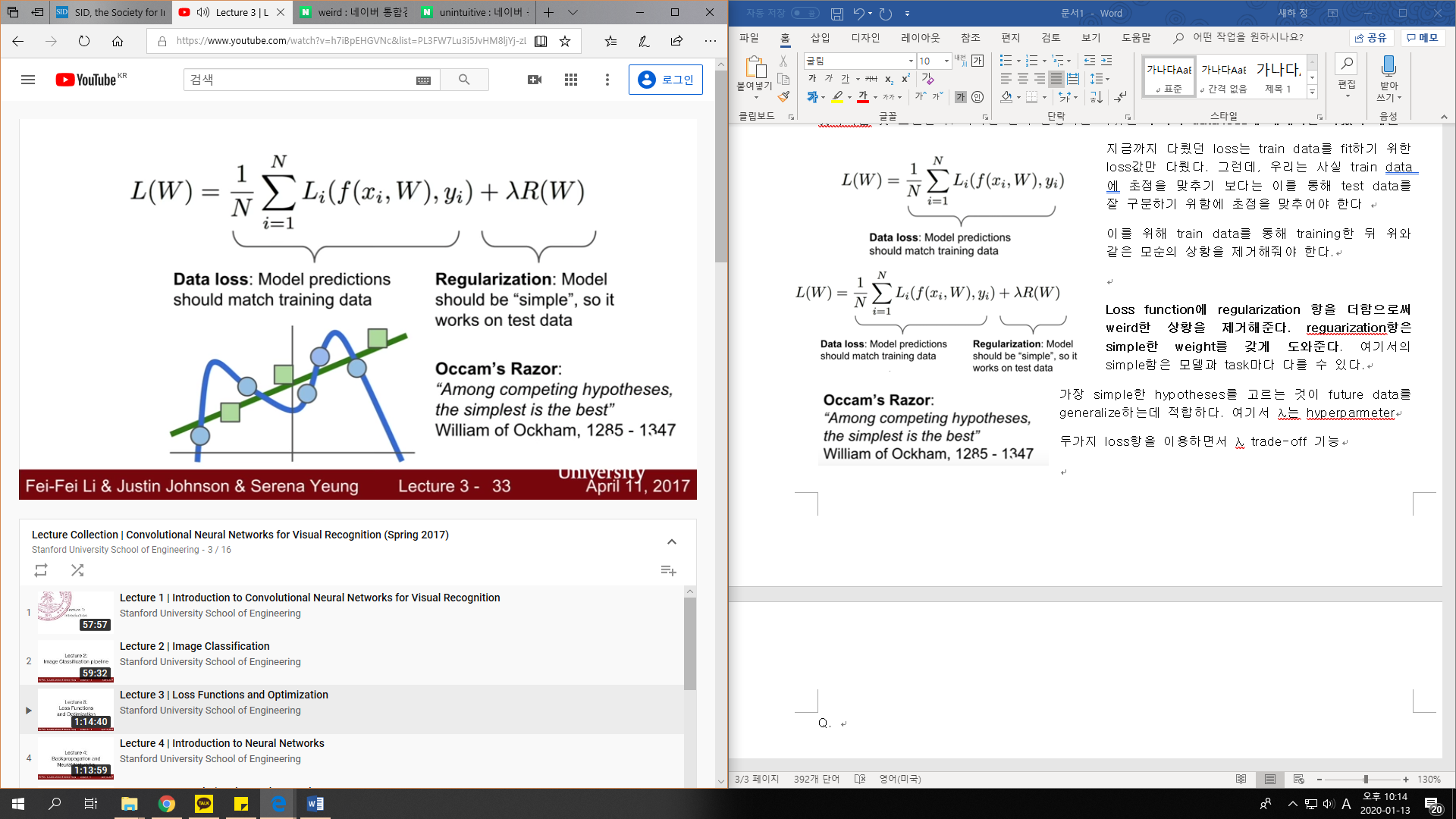
지금까지 다뤘던 loss는 train data를 fit하기 위한 loss값만 다뤘다. 그런데, 우리는 사실 train data 에 초점을 맞추기 보다는 이를 통해 test data를 잘 구분하기 위함에 초점을 맞추어야 한다

이를 위해 train data를 통해 training한 뒤 위와 같은 모순의 상황을 제거해줘야 한다.

**Loss function에 regularization 항을 더함으로써 weird한 상황을 제거해준다. reguarization항은 simple한 weight를 갖게 도와준다**. 여기서의 simple함은 모델과 task마다 다를 수 있다.

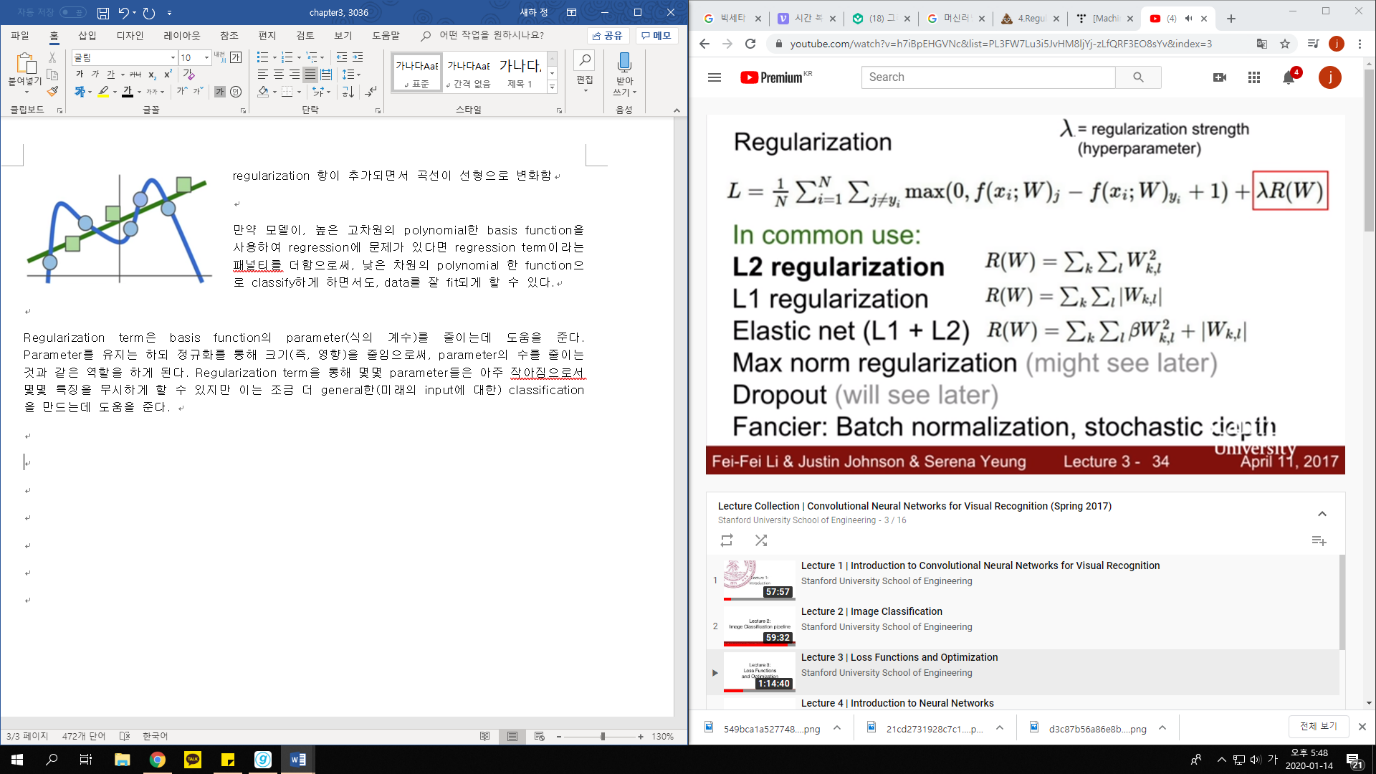
가장 simple한 hypotheses를 고르는 것이 future data를 generalize하는데 적합하다. 여기서 λ는 hyperparmeter

두가지 loss항을 이용하면서 λ trade-off 기능

regularization 항이 추가되면서 곡선이 선형으로 변화함

만약 모델이, 높은 고차원의 polynomial한 basis function을 사용하여 regression에 문제가 있다면 regression term이라는 패널티를 더함으로써, 낮은 차원의 polynomial 한 function으로 classify하게 하면서도, data를 잘 fit되게 할 수 있다.

Regularization term은 basis function의 parameter(식의 계수)를 줄이는데 도움을 준다. Parameter를 유지는 하되 정규화를 통해 크기(즉, 영향)을 줄임으로써, parameter의 수를 줄이는것과 같은 역할을 하게 된다. Regularization term을 통해 몇몇 parameter들은 아주 작아짐으로서 몇몇 특징을 무시하게 할 수 있지만 이는 조금 더 general한(미래의 input에 대한) classification을 만드는데 도움을 준다.

L 1 regularization = |W|이므로 미분하면 k 즉, 아무 의미 없는 상수가 된다. 이를 통해 weight는 0에 수렴하게 될 수도 있다.

그러나 L2 regularization을 미분하면 2W이므로 매번 2w%씩 감소하거나 증가하는것으로 볼 수 있다. 이는 2w%씩 아무리 감소해도 0이 될수는 없으므로 parameter의 strength 를 정규화 해줄뿐 수를 줄이지는 못한다. 어떻게 보면 regularization은 test set에 fit하기 위한 data loss는 train set에 fit하기 위한 줄다리기 이며 data loss와 regularization의 balance를 맞춤으로써 적절한 weight 값을 찾게 된다. 또한 이와 같은 이유로 sparse함을 상속하기 위해서는 L1 regularization이 쓰이는 경우가 있다.