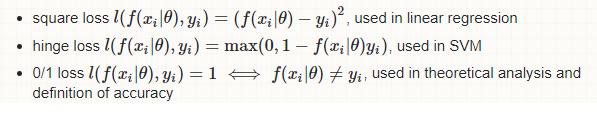
**자주 사용되는 비용 함수의 종류와 기능**

**목적: 머신러닝 알고리즘을 최적화하기 위해서는 모델 결과를 평가해야 하는데 그 평가를 위하여 사용됨. 비용함수에는 비용 함수(Cost Function), 손실함수(Loss Function), 목적함수(Objective Function)이 존재하며 이 셋은 전체적으로 큰 차이가 없다.**

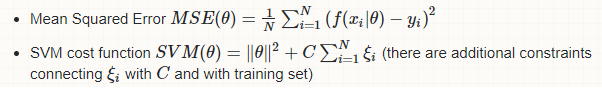
**-1) Loss Function**

**🡺 It is usually a function define on a data point, prediction and label, and measures the penalty. For example:**



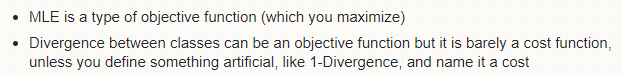
**-2) Cost Function**

**🡺 It is usually more general. It might be a sum of loss functions over your training set plus some model complexity penalty (regularization). For example:**



**-3) Objective Function**

**🡺 It is the most general term for any function that you optimize during training. For example, a probability of generating training set in maximum likelihood approach is a well defined objective function, but it is not a loss function nor cost function (however you could define an equivalent cost function). For example:**



**🡺 A loss function is a part of a cost function which is a type of an objective function**

<https://stats.stackexchange.com/questions/179026/objective-function-cost-function-loss-function-are-they-the-same-thing>

**1) L1 norm**

**L1 norm 비용함수는 절대 비용 함수라고도 한다. 대상 값과의 차이를 제곱하는 대신에 절대값을 취한다. 값 차이의 크기에 반응하는 정도가 L2 norm 보다 작기 때문에 이상치는 L1이 더 잘 처리한다. 다만 L1 norm 함수의 경우 대상 값 지점에서 꺽이는 형태를 갖고 있기 때문에 알고리즘 수렴이 잘 안 될 수도 있다.**

**TF 에서는 tf.abs(target-x\_val) 로서 구현된다.**

**2)L2 norm**

**L2 norm은 유클리드 비용함수 라고도 한다. 이 함수는 대상 값과의 거리 제곱 값을 사용한다. L2 norm은 대상 값 근처에서 기울기가 커지므로 좋은 비용 함수가 될 수 있다.**

**TF 에서는 tf.square(target-x\_vals)로서 구현된다.**

**3) Pseudo-Huber**

**Pseudo-Huber 비용함수는 후버 비용 함수를 연속적인 매끄러운 함수로 근사한 것이다. 이 함수는 대상 값 근처에서 볼록하고, 대상 값에서 먼 곳에서는 (기울기가 급하지 않은) 덜 날카로운 형태를 갖고 있어서 L2와 L1의 장점만을 취한 함수이다. 이 함수의 형태는 경사도를 결정하는 매개변수인 델타 값에 의하여 달라지게 된다.**

**TF 에서는**

**Delta1 = tf.constant(0.25)**

**Phuber1 = tf.mutiply(tf.square(delta1), tf.sqrt(1.0+tf.square((target-x\_val)/delta1))-1.0)**

**----- 분류 비용 함수**

**4. Hinge**

**힌지 비용함수는 SVM에서 주로 사용하지만, 신경망에서도 사용이 가능하다. 이 함수는 두 분류 대상인 1과 -1에 대한 비용을 계산한다. 대상값이 1인 경우라면 예측 값이 1에 가까울수록 비용 함수 값이 작아진다.**

**TF에서는**

**Hinge = tf.maimum(0. ,1. – tf.multiply(target, x\_vals))**

**5. Cross-entropy**

**교차 엔트로피 비용함수는 로지스틱 비용함수 라고도 한다. 0과 1 두 분류를 예측할 때 이 함수를 사용한다. 보통 0과 1 사이의 실수 값이 주어지는 예측 결과와 실제 분류값 (0 또는 1) 사이의 거리를 측정해야 한다. 정보이론의 교차 엔트로피 공식을 이용해서 구현한다.**

**TF에서는**

**Entropy = -tf.multiply(target, tf.log(x\_vals)) – tf.multiply(1. -target), tf.log(1. – x\_vals))**

**6. Sigmoid cross entropy**

**시그모이드 교차 엔트로피 비용함수는 X 값을 교차 엔트로피 비용 함수에 넣기 전에 시그모이드 함수로 변화하다는 점을 제외하면 앞의 교차 엔트로피 비용 함수와 매우 유사하다.**

**TF에서는**

**X\_val\_input = tf.expand\_dims(x\_vals, 1)**

**Target\_input = tf.expand\_dims(targets, 1)**

**Entropy\_sigmoide = tf.nn.sigmoid\_cross\_entropy\_with\_logits(labels=target\_input, logits=x\_val\_input)**

**Entropy\_out = sess.run(Entropy\_sigmoide) [모든 activation 에는 sess.run() 함수를 사용함]**

**7) Weighted cross entropy**

**가중 교차 엔트로피 비용 함수는 시그모이드 교차 엔트로피 비용함수에 가중치를 더한 것이다. 이 함수는 양수 대상 값에 가중치를 부여한다.**

**Weight = tf.constant(0.5)**

**Entropy\_weighted = tf.nn.weighted\_cross\_entropy\_with\_logits(targets, x\_vals, weight)**

**Entropy\_out = sess.run(Entropy\_weighted)**

**8. Softmax cross entropy**

**소프트 맥스 교차 엔트로피 비용 함수는 정규화되지 않은 출력값을 대상으로 한다. 이 함수는 여럿이 아닌 하나의 분류 대상에 대한 비용을 측정할 때 사용된다. 이 때문에 이 함수는 softmax 함수를 이용해 결과 값을 확률 분포로 변환하고, 실제 확률 분포와 비교하는 방식으로 비용을 계산한다**

**In TF**

**Unscaled\_logits = tf.constant([[1,-3,10]])**

**Target\_dist = tf.constant([[0.1, 0.02, 0.88]])**

**Softmax\_entorpy = tf.nn.softmax\_cross\_entropy\_with\_logits(logits=unscaled\_logits, labels =target\_dist)**

**9. Sparse Softmax cross entropy**

**희소 소프트 맥스 교차 엔트로피 비용함수는 앞의 소프트 맥스 교차 엔트로피 함수가 확률 분포를 대상으로 하는 것과 달리 실제 속한 분류가 어디인지를 표시한 지표를 대상으로 함. 모든 원소 값이 0이고 한 원소만 1인 대상 값 벡터를 사용하는 대신, 어떤 분류가 실제 값이지를 나타내는 지표만 전달한다.**

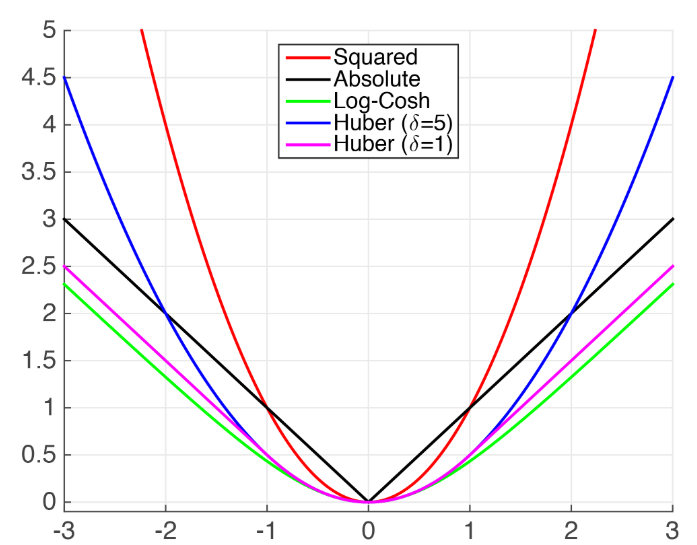
**In TF**

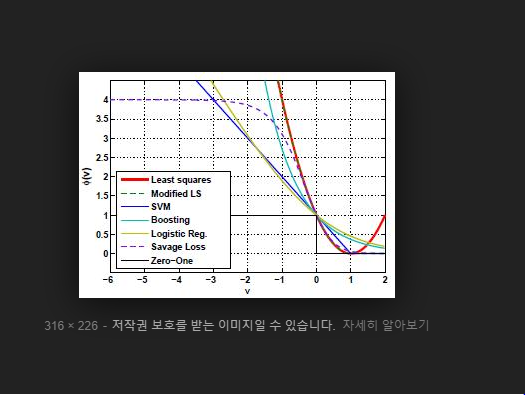
**Unscaled\_logits = tf.constant([[1,-3,10]])**

**Sparse\_target\_dist = tf.constant([2])**

**Sparse\_xentropy = tf.nn.sparse\_softmax\_cross\_entropy\_with\_logits(logits=unscaled\_logits, labels=sparse\_target\_dist)**

**\*그래프**





**\*참고**

<https://ratsgo.github.io/deep%20learning/2017/09/24/loss/> : 딥러닝 모델의 손실함수에 대하여 간단한 설명

<https://umbum.tistory.com/210>

<http://www.engear.net/wp/%EC%9D%B8%EA%B3%B5%EC%A7%80%EB%8A%A5/loss-function/>